A guide to FAIR data stewardship for health and humanitarian applications

By Joëlle Stocker, Samson Yohannes Amare, Rens Kievit, Philip Opiyo, Kai Smits, Mirjam van Reisen, Onesmus Wanjiku, Danial Zemchal.

DATA
GOVERNANCE
IN AFRICA

Table of content

Rue Stevin 115
1000 Brussels

admin@eepa.be
admin@vodan-africa.be

www.eepa.be
www.vodan-totafrica.info

Rue Stevin 115
1000 Brussels
admin@eepa.be
admin@vodan-africa.be
www.eepa.be
www.vodan-totafrica.info

Introduction

This manual serves as a comprehensive guide for implementing architectures and resources that support FAIR data stewardship, particularly in health and humanitarian contexts. For a general and basic guide to the concepts related to FAIR data stewardship, please refer to a practical guide to FAIR data stewardship.

This manual follows a specific architecture that was developed for the implementing of FAIR workflows for sensitive data, specifically related to health and humanitarian applications. The architecture contains these elements:

Common Data Model (CDM): The Common Data Model (CDM) is built upon an ontology crafted for the Humanitarian Data Space. This framework creates a shared vocabulary that outlines important concepts and their interconnections, ensuring seamless communication between various data sources. By providing a common data space for linked humanitarian and sensitive information, our ontology promotes interoperability.

Extract, Transform, Load (ETL) pipeline: At the heart of this architecture is the ETL pipeline. This process begins with a data input model and links the data to the CDM through an ontology mapper. The data is then converted into Resource Description Format (RDF), a standardized way to represent relationships between data instances. Before being stored in a triple store, the RDF syntax undergoes validation to ensure accuracy.

Monitoring dashboard: The architecture aims to empower users to run queries across multiple datasets without directly accessing the underlying data. The monitoring dashboard allows users to extract and visualize selected information, typically presented in aggregated formats. Examples include data on refugee locations or incident occurrences.

Access Control: Given the sensitive nature of the data involved, access control mechanisms are crucial. While the architecture is designed to enhance information extraction, it prioritizes the privacy and security of data subjects as well as data ownership. Data owners can manage who accesses their information, with the ability to revoke access as needed. Consent from the data owner dictates which metadata can be queried and which datasets can be accessed.

FAIR Data Points (FDPs): The architecture also incorporates FAIR Data Points (FDPs), which contain machine-actionable information about all datasets available within the Humanitarian Data Space (HDS). These catalogues outline the conditions for accessing datasets in FAIR formats and are stored on local servers, ensuring all datasets are easily findable, even if access to their content is limited.

This architecture has been tested and deployed in humanitarian organisations, as well as clinics across Africa. As a result, it has facilitated the establishment of the Humanitarian Data Space (HDS), a structured environment that encourages the integration, sharing, and management of diverse humanitarian datasets. The HDS promotes interoperability and collaboration, housing critical datasets related to sexual violence, refugee protection, human trafficking, and antenatal care.

Rue Stevin 115
1000 Brussels
admin@eepa.be
admin@vodan-africa.be
www.eepa.be
www.vodan-totafrica.info

Chapter 1: What is FAIR data?

FAIR data stands for data that is Findable, Accessible (under well-defined conditions), Interoperable and Reusable. It builds on data as Linked Data resources, defining digital data instances in semantically rich terms and transforming them in human and machine-readable formats. The aim is to recognise the value of digital data, including research data, and to ensure that these assets are available for future use. It recognises the rapid growth of data volume, and that computers are needed to create additional computationally based insights, and increase insights from diverse data sources. Semantic data or metadata increase the value of data as archived resources, and allow to maintain information on the origin of the data. FAIR also stands for Federated AI Ready, speaking to the character of FAIR Data that allows data to speak to each other while in different repositories, controlled by different researchers or data owners. FAIR data is increasingly relevant, since Machine Learning and AI benefit from high quality data and FAIR data increases the quality of the data.

FAIR Data Point is a tool to standardise the findability and access to data repositories of data generated and curated by researchers.

FAIR-OLR stipulates that all data are handled by assumption as FAIR with Ownership by data producers, Localisation of repositories and under Regulatory Compliance.

FAIRification is the process of curating digital data assets with semantic description and machine-readable formats. Semantic enrichment of the data can be defined as data that is expanded with data about the data. This can include different types of metadata, such as the metadata about the research, the researcher and the datasets (provenance and archiving metadata); machine-actionability metadata concerning how and under what conditions the data can be visited by algorithmic queries, metadata about the meaning of data instances itself and linking these to existing standards and definitions of the meaning of the data, as well as machine-readable links on the internet that prescribe the definitions and relations expressed in semantic format.

Federated data means that data can be controlled in closed repositories, and ensures that data can be fully protected by the person or group responsible for the data. This includes the computations that are carried out over the data. The strict control over data in FAIR format, ensures that data-access is conditioned on regulatory compliance and ethical considerations, particularly relevant for sensitive and personal data. The concerns that are managed in a FAIRification process of the data relate to risks such as potential reidentification, security concerns pertaining to the integrity of the data and the ability to protect the data, respecting the highest standards of legal and ethical frameworks.

Federated repositories are places where data is stored and from where data is handled. These can be FAIR-repositories, such as Linked data repositories. Another iteration of FAIR Data is Solid Data, which is particularly designed to protect personal data. The combination of FAIR and Solid gives increasing understanding of how data can be computed across different repositories.

Interdisciplinary FAIRification refers to the various needs that are associated with FAIRification in different disciplines as well as the different skills sets needed for any

Rue Stevin 115
1000 Brussels
admin@eepa.be
admin@vodan-africa.be
www.eepa.be
www.vodan-totafrica.info

FAIRification process, including data science, legislation of digital data handling, medicine, computer science, humanities, mathematics, AI, social science and any subject matter that pertains to a domain of data that is handled.

Linked Data is the methodology of creating semantic machine-actionable assets of digital data on the internet that replaces the concepts of Open Data or unstructured Big Data.

Machine-readable means that data is formatted in a way that computers can easily understand and process. This often involves structured formats (like JSON or XML) that allow software to interpret and utilize the data without human intervention.

Metadata is data that provides information about other data. It can include details like the creation date, author, file format, and description, helping users understand the context and quality of the data, and making it easier to search for and manage.

Semantic data refers to data that is enriched with meaning and context, making it clearer for both humans and machines. By using concepts and relationships defined in ontologies or vocabularies, semantic data allows for deeper understanding and automated reasoning about the information.

SOLID is a set-up of Linked Data that is used for personal data repositories.

Rue Stevin 115
1000 Brussels
admin@eepa.be
admin@vodan-africa.be
www.eepa.be
www.vodan-totafrica.info

Chapter 2: Metadata

One of the most crucial concepts to understand in-depth for FAIR data stewardship is metadata. Broadly speaking, metadata is data that describes other data. This can take shape in various ways, and understanding the different kinds of metadata that exist is essential for effective data stewardship. Metadata is characterised in many different ways, but it is generally grouped into three primary categories that relate to a dataset or data fragment: content metadata, syntactic metadata, and semantic metadata[1]. Each category serves a unique purpose and contributes distinctly to the overarching principles of Findability, Accessibility, Interoperability, and Reusability (FAIR).

Content metadata provides information about the actual content of the data. This includes the title, description, and types of data contained within the dataset. It is fundamental for users to ascertain the relevance and scope of the data for their specific needs. This type of metadata enhances the Findability and Reusability of data, as it allows potential users to evaluate whether the information contained is pertinent to their objectives.

Syntactic metadata refers to the structure of the data, including the format, language, and any encodings used. This type of metadata is crucial for ensuring that data can be processed and retrieved efficiently, thus playing a vital role in Accessibility and Interoperability. By providing clarity on data formats and structures, syntactic metadata facilitates better integration across diverse systems and platforms.

Semantic metadata empowers users to understand the meaning of the data within its context. It involves the relationships and concepts underlying the data and supports users in navigating the complexities of data interpretation. This category of metadata is integral for Interoperability, as it allows datasets to be linked and compared meaningfully, fostering a richer understanding of the data's implications and utility.

Metadata about catalogues, distributions and datasets

It is important to distinguish metadata that describes datasets as opposed to those describing data fragments. The former facilitates users in locating datasets that meet their needs, understanding the context of the data, and recognising how to access it. Effective metadata related to datasets includes various attributes such as ownership, creation date, and version history, which are imperative for ensuring transparency and traceability.

In the context of catalogues, metadata helps in the compilation and management of multiple datasets, providing an overarching structure that enables users to navigate through large quantities of data efficiently. This includes metadata elements like:

- Catalogue title and description, offering a summary of the collection and types of datasets contained within.
- Access rights clearly defining who can use the data, under what conditions, and any associated licensing information.

[1] Movva, S., Ramachandran, R., Li, X., Khaire, S., Keiser, K., Conover, H., & Graves, S. (2005). Syntactic and semantic metadata integration for science data use. *Computers & geosciences*, 31(9), 1126-1134. https://doi.org/10.1016/j.cageo.2004.12.011

Rue Stevin 115
1000 Brussels
admin@eepa.be
admin@vodan-africa.be
www.eepa.be
www.vodan-totafrica.info

- Data distribution format specifying the formats available for data download or access, assisting users in identifying suitable options based on their technical requirements.
- The language of the dataset.

Such metadata not only enhances the Findability of data but also plays a pivotal role in ensuring that data is easily accessible and reusable. By standardising these attributes across catalogues, organisations can promote clarity and consistency while fostering trust among data users.

Metadata about data fragments

When delving into the specifics of individual data fragments within datasets, metadata assumes an equally critical role. These fragments represent the smallest units of data (such as individual entries, records, or observations) that collectively form a dataset. Metadata pertaining to these fragments enriches understanding by providing context and details necessary for accurate interpretation and application.

Key attributes of metadata associated with data fragments may include:

- A unique identifier that distinguishes each fragment, facilitating easy retrieval and citation.
- Detailed specifications that describe the format of each fragment, such as numeric, textual, or categorical. This serves as a guide for proper data handling.
- Temporal information indicating the time period during which the data was collected or derived.
- Spatial information providing details about geographical relevance when applicable, enabling better integration with geospatial analyses.

This granular level of metadata enhances the Accessibility and Interoperability of datasets by allowing users to understand the precise characteristics of each data fragment. Moreover, it fosters Reusability by enabling researchers to assess the applicability of individual fragments to their inquiries without needing to examine the entire dataset. By ensuring that each data fragment is well-documented with appropriate metadata, organisations can significantly improve the efficiency and effectiveness of their data use in health and humanitarian contexts.

Metadata within the data space

Metadata are key elements in a data space, ensuring FAIRness and linked data. Within a data space, the three types of metadata have diverging functions.

Content metadata provides descriptive information about the actual contents of datasets. Its contributions to a data space include:

- Enhancing Discoverability: By providing titles, descriptions, and keywords, content metadata makes datasets easier to locate during searches.
- Facilitating Contextual Understanding: Content metadata informs users about the nature, scope, and intended use of the data.
- Supporting Data Governance: By identifying data sources, ownership, and rights information, content metadata ensures that proper ethical and legal standards are maintained, fostering trust among users and stakeholders in the data space.

Rue Stevin 115
1000 Brussels
admin@eepa.be
admin@vodan-africa.be
www.eepa.be
www.vodan-totafrica.info

- Promoting Collaboration: Clear content metadata can facilitate collaboration by providing sufficient context and information, enabling diverse teams to work together more effectively on data-driven projects.

Syntactic metadata refers to the structural and technical details regarding data formats and organization. Its contributions to a data space include:

- Ensuring Accessibility: Syntactic metadata includes information on file formats, encoding types, and data structure.
- Facilitating Interoperability: By providing technical standards and schema, syntactic metadata allows different systems and applications to interoperate seamlessly. This is crucial for integrating data from multiple sources, enabling richer analyses.
- Improving Data Integration: Syntactic metadata helps identify the data types, constraints, and relationships within datasets. By understanding these structural aspects, users can integrate datasets more effectively, leading to more comprehensive analytical outcomes.
- Streamlining Data Processing: Metadata that clearly delineates schema and format enables automated processes, enhancing the efficiency of data ingestion and processing within the data space.

Semantic metadata enhances the richness of understanding by providing context and meaning beyond mere data representation. This type of metadata helps to interpret the relationships between individual data fragments and situates them within broader conceptual frameworks or workflows, which is particularly vital in complex fields such as health and humanitarian applications. Key aspects of semantic metadata for data fragments include:

- Contextual Relationships: This describes how a data fragment relates to other fragments within the dataset or to external datasets. For instance, if a data fragment represents patient health records, its semantic metadata could explain its relationship to treatment records, demographic data, or clinical outcomes. Such relationships facilitate more comprehensive analyses and enable users to draw meaningful conclusions.
- Ontology References: Using well-defined ontologies or controlled vocabularies can standardise the interpretation of data fragments. By linking data entries to established terminologies (e.g., SNOMED CT for medical terms), semantic metadata ensures that data is understood consistently, which is crucial for both interoperability and collaboration across different platforms and disciplines.
- Data Provenance: Acknowledging the lineage of each fragment, including its origin, transformations, and the methods used for its generation, plays a key role in validating the data's reliability. This semantic metadata contributes to a deeper understanding of the data's context and supports its responsible use.

To establish a data space, agreements need to be set on how the different kinds of metadata are created, published and how their longevity is preserved.

Rue Stevin 115
1000 Brussels
admin@eepa.be
admin@vodan-africa.be
www.eepa.be
www.vodan-totafrica.info

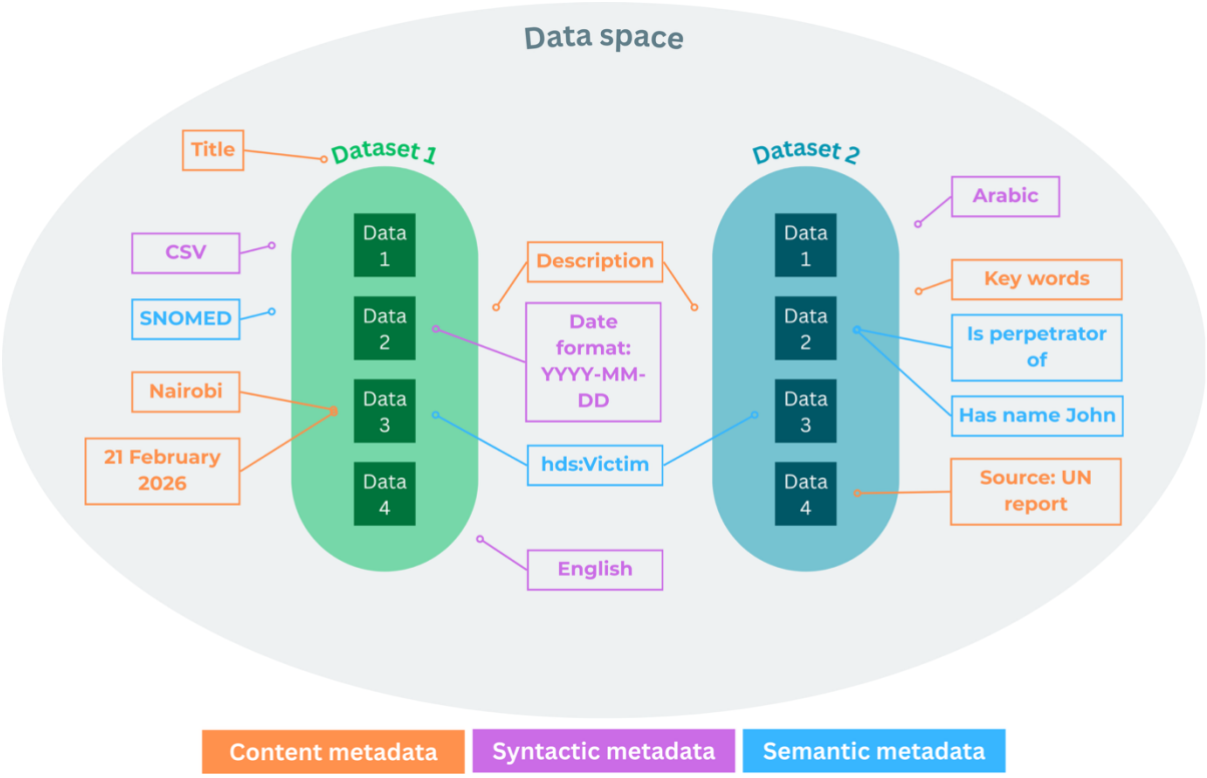


Figure 2.1. How metadata describes datasets and data instances within a data space.

Rue Stevin 115
1000 Brussels
admin@eepa.be
admin@vodan-africa.be
www.eepa.be
www.vodan-totafrica.info

Chapter 3: Common Data Models

Common Data Models (CDM) are standardised frameworks that facilitate the exchange of information and data across various applications and systems. By establishing these standardised models, information and data from different sources can be linked effectively. This linking is accomplished through the use of standardised terminology that consistently describes the nature and attributes of the data across different platforms.

Definitions of key terms

Knowledge graph are structured representations of interconnected information that captures relationships between entities, such as people, places, concepts, and events, along with their attributes. This format allows for easy retrieval and analysis of data, enabling more effective understanding and insights.

Ontologies serve as a formal representation of a set of concepts within a particular domain, outlining both the terms used and the relationships between those terms.

Metadata

CDMs are used to standardise metadata – metadata about datasets and about source data. Metadata related to datasets is important for findability, whereas metadata about the source data enables the creation of Knowledge Graphs, structured representations of interconnected information that captures relationships between entities along with their attributes.

Building blocks of a CDM

Classes are the fundamental building blocks of a Common Data Model. They represent categories or types of entities within the data structure. Each class defines a specific entity type, such as "Person," "Location," or "Event." They can be organised in a hierarchical structure, where a parent class may lead to more specialised subclasses. For example, a parent class "Animal" may have subclasses like "Mammal" and "Bird." Classes ensure that entities across different datasets are categorised consistently, which aids in data integration and prevents ambiguity.

Properties are attributes or characteristics of the classes, providing essential information about each entity. Each property describes specific aspects of a class. For example, a "Person" class might have properties like "Name," "Age," "Gender," and "Occupation." Properties are associated with defined data types (e.g., string, integer, date), which dictate what kind of information can be stored. For instance, the "Age" property would typically be an integer. Properties can also define relationships between classes, facilitating connections within the data model. For example, a property like "Lives_In" might connect a "Person" class to a "Location" class, indicating where the person resides.

Properties are defined by their **domain** and **range**. The domain of a property specifies the class or classes to which this property applies. It indicates the type of entity that can have the property. For example, if we have a property called “hasAge”, the domain might be the class “Person”. This means that only instances of the class “Person” can have the property “hasAge” associated with them. The range of a property defines the potential values of classes that the property can take. It specifies the type of data or entity that the property can hold. For instance, the range of the property “hasAge” might be defined as “Integer”. This indicates that the value of “hasAge” must be an integer representing

Rue Stevin 115
1000 Brussels
admin@eepa.be
admin@vodan-africa.be
www.eepa.be
www.vodan-totafrica.info

person's age. The range of a property can also be defined as another class, if the relevant representation is an individual from that class.

Relationships link different classes and establish how they interact within the data model. They represent associations among classes, providing important context. For example, a "Student" class may have a relationship with a "Course" class through an "Enrolled_In" property. Relationships can have cardinality constraints, indicating how many instances of one class can relate to another, such as one-to-one, one-to-many, or many-to-many relationships.

Common Data Models are often developed in alignment with established ontologies, promoting semantic consistency by using widely accepted terminologies, which facilitate better understanding and integration across systems. CDMs should be designed to be flexible and scalable enabling new classes and properties to be added as needed to accommodate evolving data requirements, and ensuring it remains relevant and effective for diverse applications.

Steps in developing a CDM

Developing a CDM starts by understanding the data that you want to structure. Domain experts should be closely involved as they understand the data and concepts the best.

The first step in developing a CDM is to **define the use case and scope**. This is also where you will map relevant domain experts, data stewards and end-users to involve in the project. The use case and scope determine the kind of information that you want to extract from the dataset and therefore how the data should be structured. From this, and based on existing datasets, a **data input schema** should be created. This schema should identify the different elements of the datasets that are of important, and allows the identification of concepts to include in the CDM. At this stage, these concepts should be clearly defined, keeping in mind the use case and scope, and with the input of domain experts.

In the second step, **a rudimentary vocabulary is established** based on the concepts that have been identified and defined. To ensure interoperability, these concepts should be linked to existing standard vocabularies and ontologies. It is best to use as widely used and accepted standards and resources as possible.

Example of commonly used vocabularies

- Schema.org: excellent for general-purpose descriptions of people, places, events, products.
- Dublin Core (DC): general-purpose vocabulary for describing resources.
- FOAF: for describing people and organisations.
- SKOS: for representing taxonomies and thesauri.
- PROV-O: for describing provenance information.
- DCAT: data catalogues.

There are a number of platforms that can be used to search for ontologies and entities defined within these ontologies:

- Linked Open Vocabulary (LOV)
- Ontobee
- Ontology Lookup Service (OLS)
- FAIR sharing

Rue Stevin 115
1000 Brussels
admin@eepa.be
admin@vodan-africa.be
www.eepa.be
www.vodan-totafrica.info

- Domain-specific portals, such as bioportal. You can find an overview of all portals on ontoportal.

Where no suitable vocabulary exists in your findings, feel free to define your own. But make sure to Use a standard namespace (e.g., https://www.myorg.org/ontology/).

⇒ **Classes:** myorg:Patient, myorg:ClinicalTrial

⇒ **Properties:** myorg:hasDiagnosis, myorg:participatesInTrial

Make sure to **document the Model** by creating human- machine-readable documentation for your ontology. Tools like **Protégé** can be used to design and export the model in RDF/S (Turtle or RDF/XML). Models and vocabularies should be uploaded to findable resources (e.g. bioportal) to ensure that it is FAIR itself.

From this a community vocabulary can be created, containing the concepts from the data schema, and their linked terminologies. Elements in the vocabulary should be distinguished between classes and properties, in the case of the latter, the domain and range should also be specified. Whether a concept should be a class or a property is not always straightforward, and a concept may even function as both. Various factors influence this decision, including the desired complexity of your CDM. As a rule of thumb, having fewer classes increases the simplicity of your CDM, making implementation easier, but it also limits the level of detail in data structuring.

The final step is to **create the model by translating it into a machine-actionable language suitable for data transformation**. It is essential to write the model in the appropriate syntax to ensure seamless integration into existing data pipelines. For instance, turtle (TTL) is a syntax used for expressing RDF (Resource Description Framework) data, allowing for the serialization of RDF graphs in a more human-readable format. Once the model is created, it should undergo thorough testing and validation to identify any inconsistencies or issues. This validation phase ensures that the model accurately represents the intended semantics and can effectively support the required data transformation tasks. After resolving any identified issues, the model should be adapted based on feedback and testing outcomes, refining it to better meet the project goals.

Upon finalisation, the model, along with relevant documentation, should be published on collaborative platforms such as GitHub. This not only facilitates version control but also allows for community contributions and enhancements. Additionally, hosting the model on a searchable resource like BioPortal increases its visibility and accessibility to other users and researchers in the field.

Rue Stevin 115
1000 Brussels
admin@eepa.be
admin@vodan-africa.be
www.eepa.be
www.vodan-totafrica.info

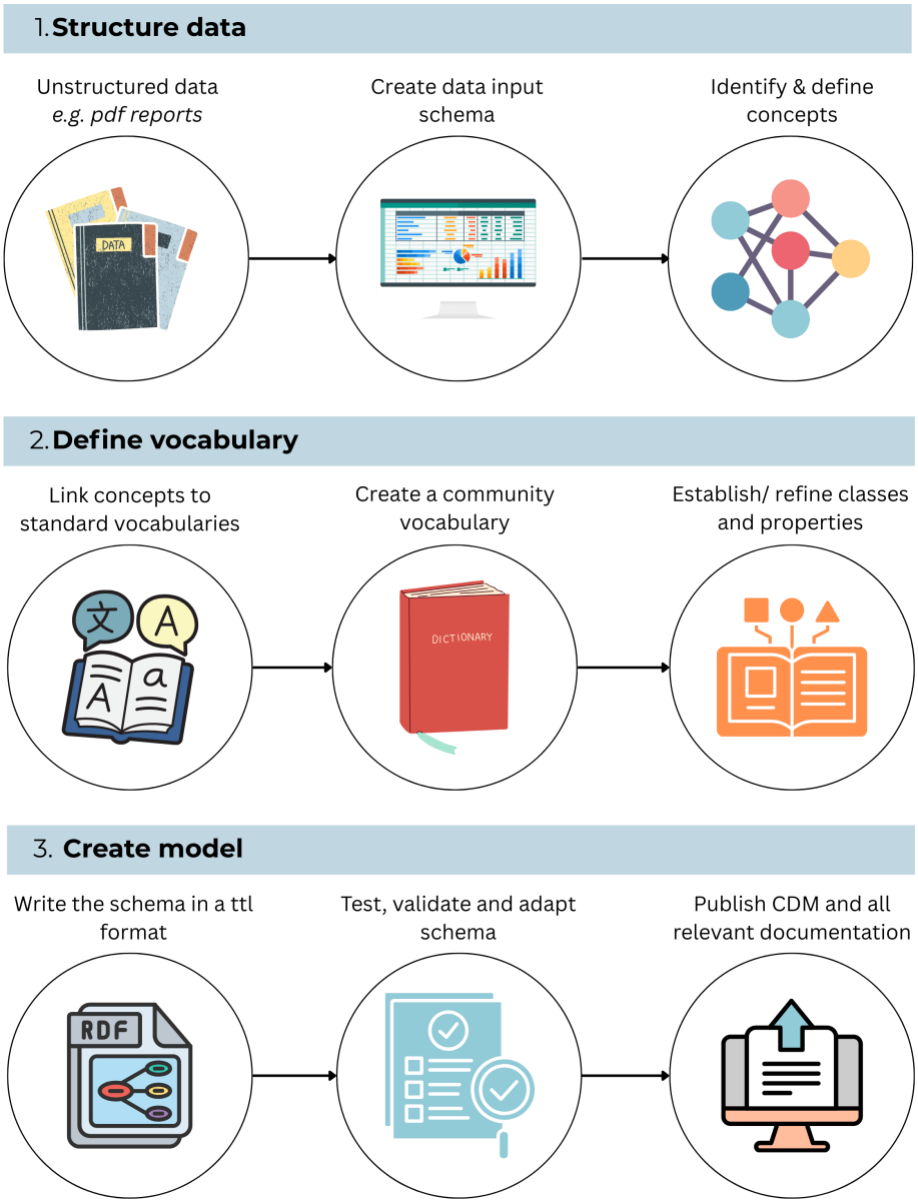


Figure 3.1. Steps for creating a CDM.

Concrete example: structuring incidents of sexual violence

The starting point is always the **data schema** based on real datasets. For the purpose of this exercise, we have created a mock dataset based on a data schema which was created with real incidents of sexual violence data. The data schema with mock data can be accessed here (warning: the mock data contains explicit language on sexual violence).

From the data schema, classes and properties can be defined. In our example data schema, we can see some obvious classes: incident, victim, perpetrator and record. For this exercise, we have mapped the classes and properties as in *figure 3.2*. At this stage, it is also important to define each concept and property with clear definitions relating to the original data and data schema. For instance, in our example the class of Incident can be defined as: "an incident of sexual violence" and victim as: "the victim of an incident of sexual violence". Properties should also be clearly defined; we define medical impact as "Medical issues sustained by the incident" and country "the country where the incident took place". At this stage, the format of the information to be filled in should be defined,

Rue Stevin 115
1000 Brussels
admin@eepa.be
admin@vodan-africa.be
www.eepa.be
www.vodan-totafrica.info

for instance dates should be inputted with the dateTime format, ages should be integers and texts such as names and descriptions should be strings. Here we already start linking to controlled vocabularies, by defining that these formats are conform to the *xsd language*.

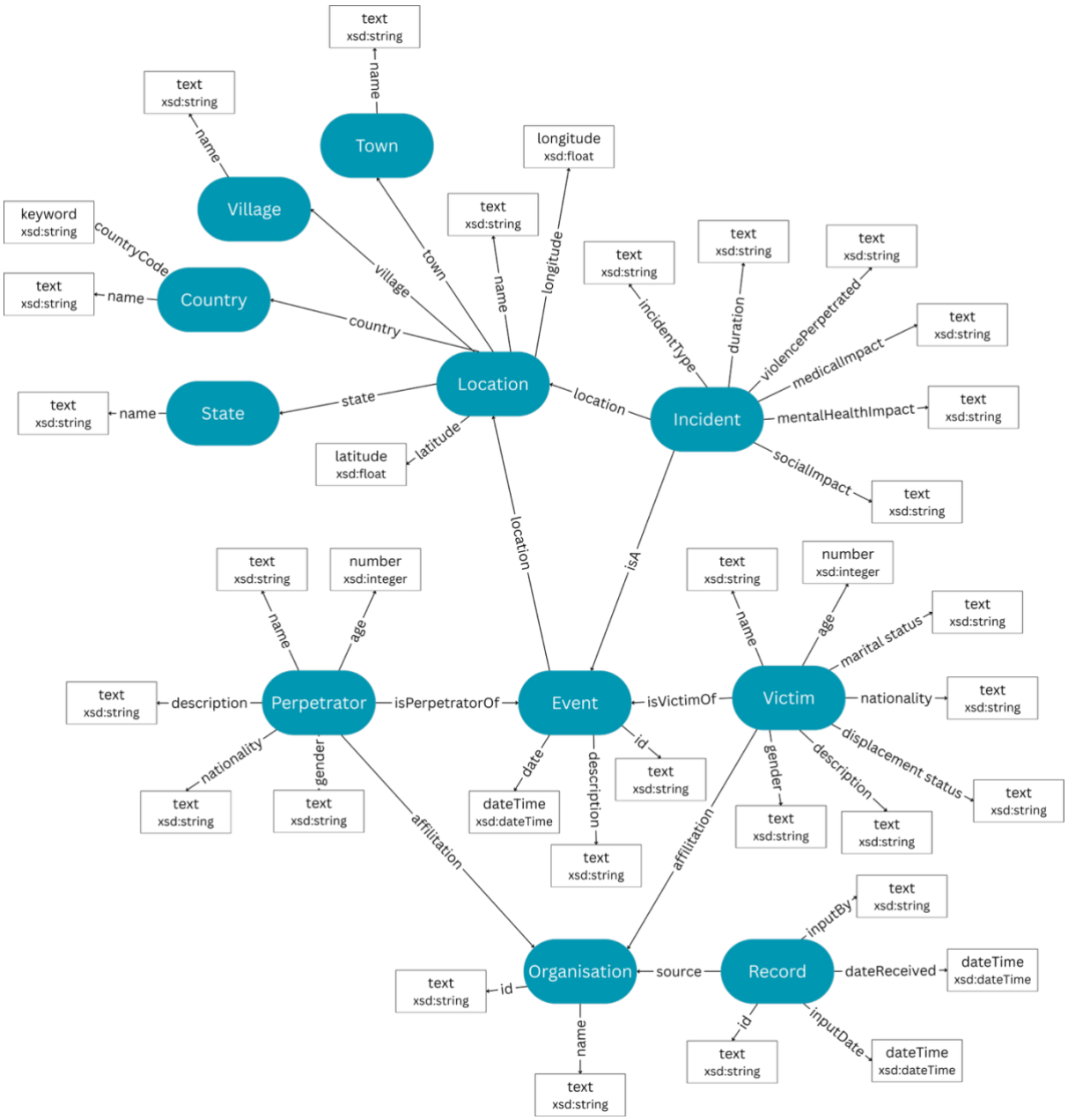


Figure 3.2. visualisation of knowledge graph for the inputting of incidents of sexual violence.

Examples and syntax

The following examples are in ttl.

The model begins with listing all the existing vocabularies and resources that you will use, with links to these resources.

```
@prefix hds: <http://example.org/hds#> .
@prefix schema: <http://schema.org/> .
```


Rue Stevin 115
1000 Brussels
admin@eepa.be
admin@vodan-africa.be
www.eepa.be
www.vodan-totafrica.info

```
@prefix dct: <http://purl.org/dc/terms/> .
@prefix prov: <http://www.w3.org/ns/prov#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix gsso: <http://purl.org/gsso/> .
```

Following this, classes can be defined. In the example below, two classes are defined: "Perpetrator" and "Victim", and the preferred label is indicated together with the definition under rdfs:comment.

```
hds:Victim a owl:Class ;

    rdfs:label "Victim" ;

    rdfs:comment "Person harmed in an incident." ;

hds:Perpetrator a owl:Class ;

    rdfs:label "Perpetrator" ;

    rdfs:comment "A person responsible for an action
causing harm" ;
```

Properties are defined in a similar way. For properties, it is important to add the *domain* and *range*, which define which classes are defined by the property. In the example below, the property is the affiliation of victims and perpetrators (the domain) to organisations (the range).

```
hds:affiliation a owl:ObjectProperty ;

    owl:equivalentProperty schema:affiliation ;

    rdfs:domain [ owl:unionOf (hds:Victim hds:Perpetrator)
] ;

    rdfs:range hds:Organisation .
```

Additional resources

- The ontology for the Humanitarian Data Space

Rue Stevin 115
1000 Brussels
admin@eepa.be
admin@vodan-africa.be
www.eepa.be
www.vodan-totafrica.info

Chapter 4: Extract, Transform, Load (ETL) Pipelines

An ETL pipeline is a framework used for the process of Extracting, Transforming, and Loading data from various sources into a destination storage system, such as a database or data warehouse. This process is essential for data integration and ensures that data is organised, clean, and ready for analysis.

The ETL pipeline consists of the following components:

Extract: During this phase, data is collected from multiple sources. These sources can include databases, ERP systems, CRM software, flat files, and APIs. The aim is to gather all relevant data needed for analysis.

Transform: Once the data is extracted, it goes through several transformations to standardise, clean, and enrich the data. This may involve actions like data cleaning (removing duplicates, correcting errors), data normalization (standardising formats), aggregation (summarising data), and filtering (removing irrelevant data). Transformation ensures that the data is in a suitable format for analysis and meets any specific requirements.

Load: The final step involves loading the transformed data into a target system, such as a database or data warehouse. This is where the data becomes accessible for analysis and reporting. The loading process can be done in real-time or in batch mode, depending on the use case.

Overview of system architecture

The characteristics of the ETL pipeline described in this guide are that it is FAIR-compliant, it allows for seamless integration with existing ontologies and other systems, and it enables ontology-driven semantic interoperability for data sharing and analytics.

The steps below map out how to develop a fully integrated software system that:

- Accepts user data from a CSV upload, an API ingestion, or using an interactive data entry form
- Automatically maps data to a predefined ontology that is stored in an FDP, BioPortal, on an official website, GitHub, or other public domain.
- Generates ontology-rich RDF triples containing the RDF Schema Classes as well as data instances converted to RDF by attaching the right ontology to the item.
- Validates data using SHACL shapes.
- Stores triples in a triple store (e.g., AllegroGraph, Fuseki) with full semantic context.
- Exposes a SPARQL endpoint for queries and analytics

The diagram below indicates the steps that need to be followed, starting from raw data to a semantically rich graph database that is available for local and remote queries.

Rue Stevin 115
1000 Brussels
admin@eepa.be
admin@vodan-africa.be
www.eepa.be
www.vodan-totafrica.info

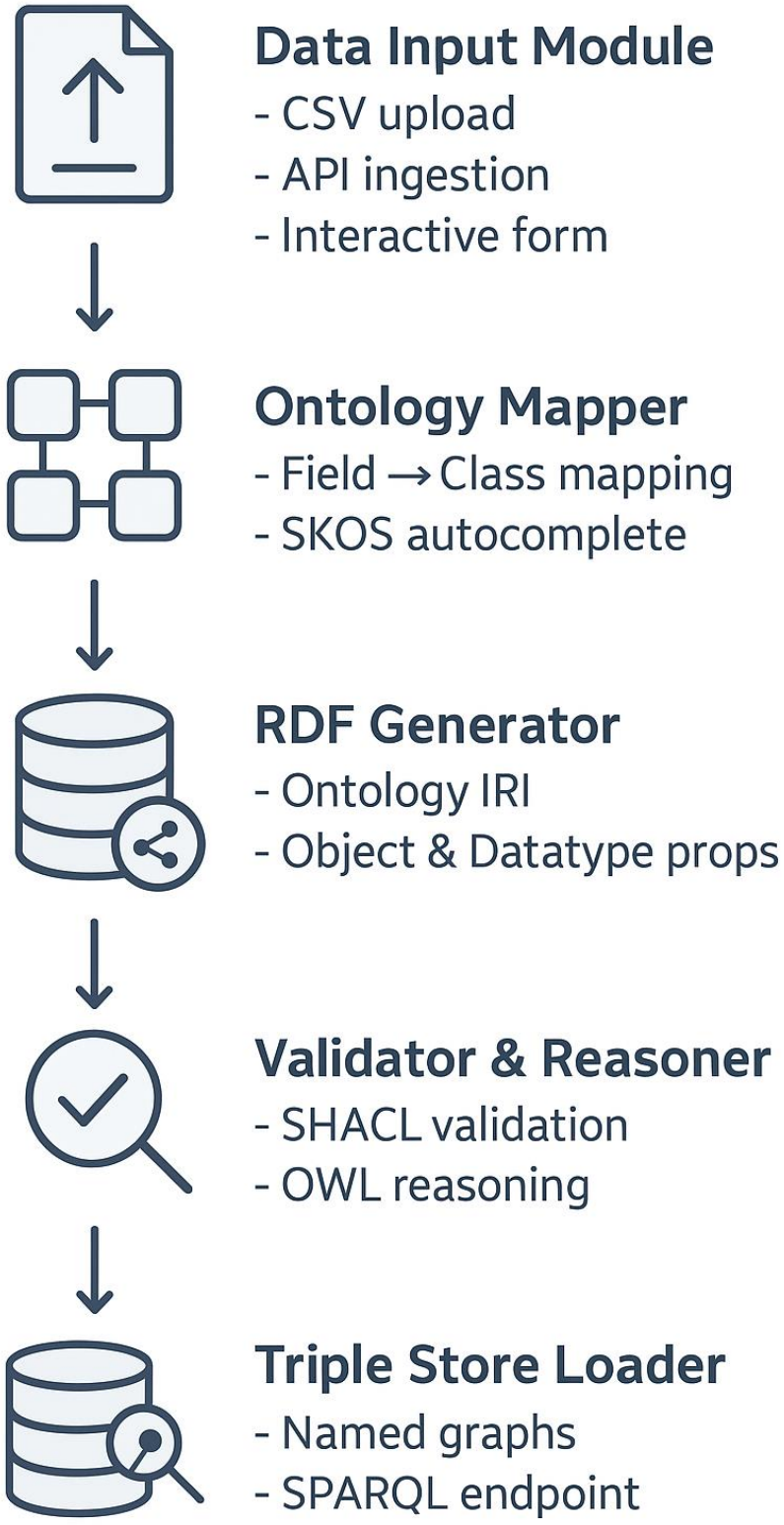


Figure 4.1. Steps of the ETL pipeline.

Rue Stevin 115
1000 Brussels
admin@eepa.be
admin@vodan-africa.be
www.eepa.be
www.vodan-totafrica.info

Technical components and tools

The following are some of the recommended tools to be used in the process indicated above. This is in no way an exhaustive list of the tools. Feel free to choose tools as you see fit.

Component	Technology / Library	Notes
Programming Language	Python 3.10+	Core system logic
RDF & Ontology	RDFLib	Triple creation & serialization
Validation	pySHACL	SHACL-based validation of triples
Reasoning	RDFLib reasoning / OWL-RL	Infer class membership & consistency
Triple Store	AllegroGraph / Fuseki / GraphDB	Storage & SPARQL endpoint
Web / Data Entry Form	Flask / FastAPI / HTML+JS	Interactive user input, autocomplete from SKOS via BioPortal
API Ingestion	REST endpoints	Accept JSON or CSV payloads

Implementation Steps

Step 1: Environment Setup

Install Python 3.10+ and required packages:

```
pip install rdflib pyshacl flask owlrl requests
```

Deploy **triple store** locally or on server (AllegroGraph/Fuseki/GraphDB).

Step 2: Ontology Preparation

1. Load the ontology (Turtle format) into the system.
2. Validate ontology syntax in Protégé.
3. Define **ontology IRI** for data mapping and named graphs.

Step 3: Data Input Module

CSV Input:

- Parse CSV files using Python pandas.
- Validate column names against ontology mapping config.

API Input:

- Define REST endpoints accepting JSON/CSV payload.
- Map fields to ontology classes/properties automatically.

Rue Stevin 115
1000 Brussels
admin@eepa.be
admin@vodan-africa.be
www.eepa.be
www.vodan-totafrica.info

Interactive Form:

- Browser-based or CLI form.
- Autocomplete for fields, for instance, in ANC may include variables like Diagnosis, Lab Test, and Location from ontology SKOS terms.

Ontology mapping

⇒ Map input fields to ontology classes and properties.
⇒ Create object and datatype properties in RDF.
⇒ Apply controlled vocabulary for relevant fields (SKOS).

Example mapping:

Input Field	*Ontology Class*	*Ontology Property*
PatientID	anc:Patient	anc:id
Age	anc:Patient	anc:age
VisitDate	anc:ANCVisit	anc:date
Hemoglobin	anc:LabTest	anc:hasLabResult
Diagnosis	anc:Diagnosis	skos:exactMatch
Diagnosis	anc:Patient	anc:hasLabResult

RDF Generation

⇒ Generate ontology-rich triples using RDFLib.
⇒ Include ontology IRI as graph context.
⇒ Serialize RDF in Turtle and optionally RDF/XML or JSON-LD.

Validation & Reasoning

⇒ SHACL Validation: Ensure data complies with ontology constraints.
⇒ OWL Reasoning: Infer additional properties and class memberships.

Triple Store Loader

⇒ Push RDF triples into a triple store under a named graph for the dataset.
⇒ Ensure ontology IRI linkage for semantic context.
⇒ Expose SPARQL endpoint for queries.

API & Query Layer

⇒ Provide SPARQL API for querying stored triples.
⇒ Optional endpoints for graph export (Turtle, JSON-LD).
⇒ Enable automated batch updates from CSV/API.

Monitoring & Logging

⇒ Track ETL pipeline execution and errors.
⇒ Log SHACL validation errors.

Rue Stevin 115
1000 Brussels
admin@eepa.be
admin@vodan-africa.be
www.eepa.be
www.vodan-totafrica.info

⇒ Log triples successfully loaded into the triple store.

Deliverables

One-stop ETL application (Python-based, CLI and/or web UI).

- RDF generation module with ontology attachment.
- SHACL validation module for compliance checking.
- Triple store integration with SPARQL endpoint.
- Configuration file for field-to-ontology mapping.
- Documentation, including installation, usage, and extension instructions.

Opportunities for Improvement

- Standardising of vocabularies: Add equivalence links to controlled clinical classification (or terminology classification system such as ICD-10 : ICD-11.
- Ontology Enrichment: add ontologies in the medical domain such as SNOMED CT; HL7-FHIR, LOINC).
- FAIR Metadata: Add prov:wasGeneratedBy, dct:creator, dct:modified.
- Visualization: Integrate ontology graph visualization (VOWL, WebVOWL).
- Incremental ETL: Detect changes in CSV/API and update the triple store automatically.

Rue Stevin 115
1000 Brussels
admin@eepa.be
admin@vodan-africa.be
www.eepa.be
www.vodan-totafrica.info

The architecture is completed with a SPARQL endpoint, which is an HTTP API, and which allows knowledge generation by querying within and over Graph repositories. While the SPARQL endpoint can in principle execute queries in any of the repositories in the community, access of queries to the repositories is controlled.

Access control is a fundamental concept in data management that determines who can view or interact with specific data within a system. It ensures that sensitive data is protected while allowing authorised users to perform necessary actions. In the context of data stewardship, especially within federated data architectures, implementing granular access control is vital for maintaining both the security and privacy of sensitive information.

Granular Access Control in Federated Data Architectures

Federated data architectures comprise interconnected systems that allow for real-time interaction and sharing of data. *Granular access control* refers to the capability to assign specific permissions to various parts of the data, enabling nuanced restrictions and authorisations. This approach is essential for facilitating multi-directional orchestration and reuse of data analytics while ensuring the privacy and security of sensitive information.

With the principles of FAIR data, organisations can handle granular permission settings effectively. This enables the generation of insights while safeguarding sensitive information. The implementation of various access-control options creates an *intelligence layer* within the data space, acting as a secure, privacy-preserving framework for controlled computations.

Figure 5.1 depicts how access control could look like in a federated FAIR-compliant architecture, where datasets are published in FDPs.

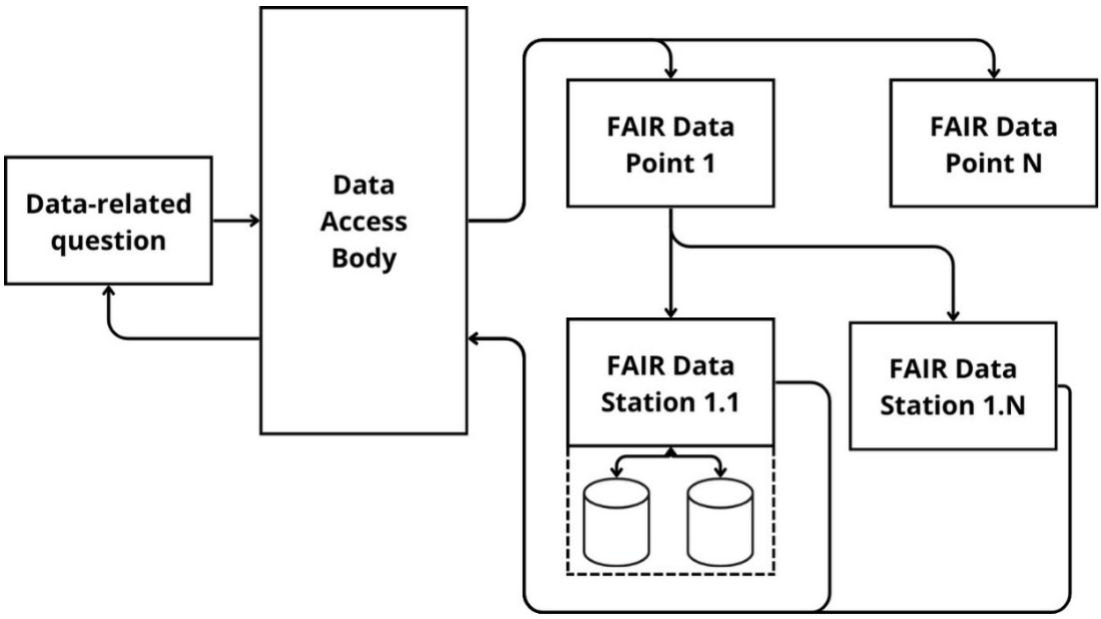


Figure 5.1. Investigating of Access-Control in Federated multidirectional reuse scenarios of FAIR data

Rue Stevin 115
1000 Brussels
admin@eepa.be
admin@vodan-africa.be
www.eepa.be
www.vodan-totafrica.info

Practical Implementation

Clinical Research Applications: One practical example of granular access control can be seen in clinical research settings that use routine patient data. By creating tools that accommodate multi-variate access control permissions, organisations can adhere to strict regulatory and ethical guidelines while working with mature datasets.

Real-Time Data Handling: As data spaces evolve, it becomes necessary to develop systems that manage access rights dynamically. This process involves FAIRification at the instance level, allowing for real-time integration into various FAIR Data Points (FDPs) to evaluate and test different access scenarios.

Challenges

Despite its benefits, implementing granular access control in data architectures comes with challenges. Key concerns include:

Reidentification Risks: The potential for de-anonymising sensitive data raises significant privacy issues, necessitating robust methods for data anonymisation.

Security Integrity: Ensuring that data remains secure from breaches is crucial, as any compromise could lead to unauthorised access and misuse.

Compliance with Legal and Ethical Standards: Organisations must navigate complex regulations and ethical frameworks to protect data subjects' rights while utilizing data for research and analytics.

By establishing a robust access control framework, organisations can ensure the secure and ethical use of sensitive data while maximising the potential of data analytics in health and humanitarian applications.

Rue Stevin 115
1000 Brussels
admin@eepa.be
admin@vodan-africa.be
www.eepa.be
www.vodan-totafrica.info

Chapter 6: Setting up FAIR Data Points

FAIR Data Points (FDP) serve as a framework for publishing and sharing metadata aligned with the FAIR principles, enhancing the management and accessibility of data resources across diverse fields. By employing standardised and machine-readable metadata, FDPs significantly improve the discoverability of datasets, making them easier to locate through search engines and data catalogues. Additionally, they facilitate collaboration among researchers, institutions, and organisations by offering a consistent framework for metadata. This standardisation enables seamless integration of metadata from various sources, promoting broader cooperation and utilisation of data.

FDPs use a RESTful API, which allows programmatic access to metadata, enabling users and applications to interact in a consistent and automated manner. They also typically include a user-friendly web-based interface (GUI) that allows users to create, visualise, and manage metadata without requiring extensive technical expertise. The metadata content in FDPs is generally generated semi-automatically, which helps maintain consistency and adherence to the FAIR principles. Furthermore, FDPs leverage widely accepted metadata standards to ensure interoperability across systems.

These data points are specifically designed to ensure compliance with the FAIR principles for both metadata and the underlying data. Moreover, FDPs can be integrated with various data repositories and databases, thus enhancing existing data management systems through FAIR-compliant metadata practices.

Creating an FDP renders data catalogues and datasets findable. The data resides in a secure container, accessible for computation only upon explicit permission. This approach will enable researchers and organisations to publish their data securely and transparently, allowing for machine learning and artificial intelligence queries, contingent on granted access. Third parties may discover the data and perform analytics if permission is provided.

There are several examples of FAIR Data Points:

- https://fdp.tangaza.ac.ke/
- https://fairdp.eepa.be/

Other FAIR Data Points can be visited here.

Challenges remain, for instance, while the possibility of algorithmic specific data visiting is possible, this needs to be made more user friendly. Other challenges pertain particularly to the automation of data visiting permission, taking into account reidentification risks and cybersecurity concerns and other challenges associated with the data access.

For specific information and documentation on setting up FAIR Data Points, please visit the following github page.

Rue Stevin 115
1000 Brussels
admin@eepa.be
admin@vodan-africa.be
www.eepa.be
www.vodan-totafrica.info

Additional resources

- https://www.fairdatapoint.org/
- FAIR Data Point software specification document
- Instructions on setting up FAIR Data Points on Ubuntu

Rue Stevin 115
1000 Brussels
admin@eepa.be
admin@vodan-africa.be
www.eepa.be
www.vodan-totafrica.info

Chapter 7: Data Spaces

Data spaces are conceptual framework that allow data to be managed, shared, and used more effectively, particularly across different domains and systems. They are increasingly important in today's data-driven environment, enabling collaboration among various stakeholders. Data spaces enable better decision-making as the integration and sharing of data across platforms, enable organisations to gain new insights and make informed decisions. It also fosters innovation by allowing different entities to collaboratively develop new solutions and services based on shared data.

Key characteristics of data spaces are:

1. Interoperability: Data spaces are designed to enable different systems and platforms to communicate and work together seamlessly. This is crucial for integrating data from various sources.
2. Data Sovereignty: Organisations retain control over their data, allowing them to share only what is necessary while maintaining compliance with regulations and privacy standards.
3. Standardised protocols: Data spaces often employ standardised methods and protocols for data exchange, ensuring consistency and reliability.
4. Collaboration: Data spaces facilitate collaboration among different organisations, sectors, or departments by providing a shared environment for data sharing and analysis.

The Health Data Space[2]

Research by VODAN on creating the African Health Data Space has focused on data sovereignty, keeping data with full ownership in localised residence under regulatory compliance in jurisdiction. The leapfrogging of an African Health Data Space tests the understanding of options for the creation of the European Health Data Space. The work involves collaboration across countries in different jurisdictions with clearly set principles, and with regulators such as the European Commission and the African Union.[3,4]

One of the major challenges for the creation of a Health Data Space is the need for awareness of the potential of such systems, training and regulatory arrangements supporting such architectures. The architecture of the Health Data Space is represented in *figure 7.1*, consisting of a data layer in which machine actionable data is created, a services layer and a user interface.

[2] https://africahealthdataspace.com/
[3] https://aun.mu.edu.et/ahds/conferences/
[4] https://pap.au.int/en/news/press-releases/2025-07-25/pan-african-parliament-champions-africas-quest-data-sovereignty-and

Rue Stevin 115
1000 Brussels
admin@eepa.be
admin@vodan-africa.be
www.eepa.be
www.vodan-totafrica.info

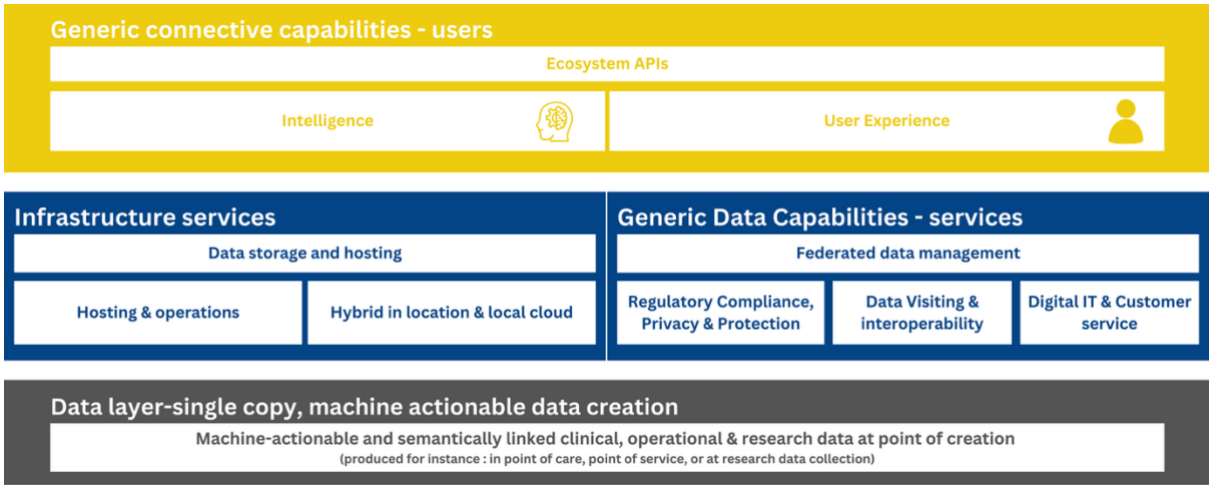


Figure 7.1. Architecture of the Health Data Space.

The Humanitarian Data Space[5]

The Humanitarian Data Space has been set up to enable data from the ground – provided by humanitarian and refugee organisations, as well as trusted sources, to connect to data users for better advocacy, interventions, research, and innovation.

At the moment, the Humanitarian Data Space consists of data related to sexual violence, human trafficking, and refugee protection from organisations working with survivors of gender-based violence, providing humanitarian assistance in refugee camps, and supporting refugees and survivors of human trafficking. The data space also consists of research data related to human trafficking.

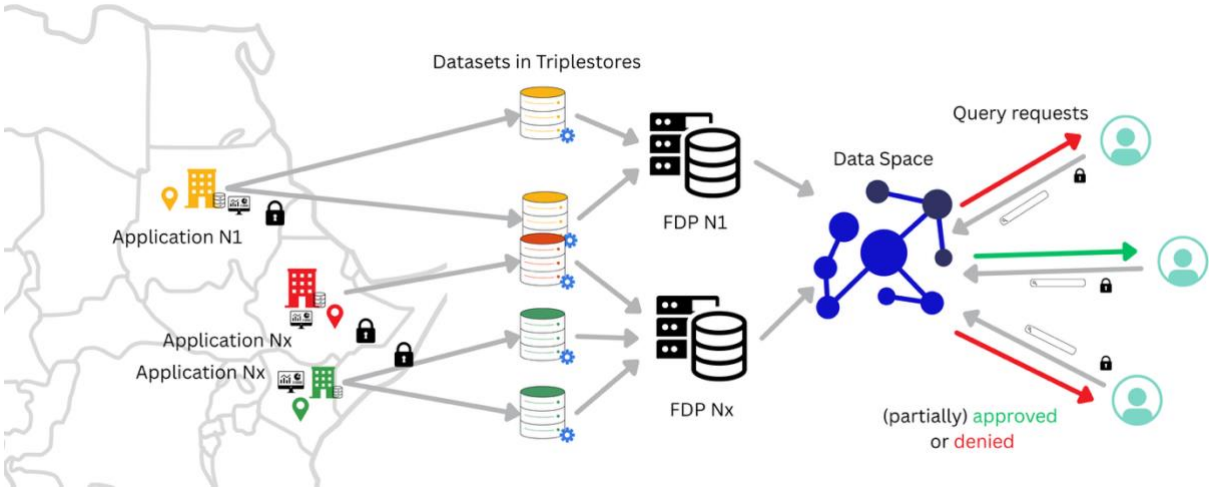


Figure 7.2. Overview of how the Humanitarian Data Space can link data from the ground to users from various institutions.

[5] https://humanitariandataspace.com

Rue Stevin 115
1000 Brussels
admin@eepa.be
admin@vodan-africa.be
www.eepa.be
www.vodan-totafrica.info

Chapter 8: Federated Data Analytics

The federated architecture, with repositories of data sets in various locations under control of data owners is the basis of a FAIR Data Space. It can handle knowledge generation on heterogenous data. The Federated architecture described in this guide is based primarily on FDPs that are used to make data discoverable, describe the contents and describe access conditions using standardised semantic metadata. The FDP only contains metadata while actual data is stored in triplestores or other repositories that are under strict control by the organizations that holds the data.

Federated query execution process

The access pattern we currently support is the following: A researcher has a query Q that they want to send to any number N of federated repositories which are discoverable through FDPs and managed by different data controllers DC. Below is the process for one dataset. Repeating this N times gives the entire query execution process in the federated architecture. This process is also presented in *Figure 8.1*.

1. Researcher discovers the relevant datasets for their question through the FDPs.
2. For each dataset they identify the email addresses of the data controller in the FDP.
3. The researcher sends the query (and other identifying information) to each email address
4. The DC verifies the processing request.
5. If approved, the DC processes the query locally and verifies the results.
6. If necessary, the results are further processed (e.g. anonymized or aggregated).
7. If approved, the DC responds to the request with the results and other information.

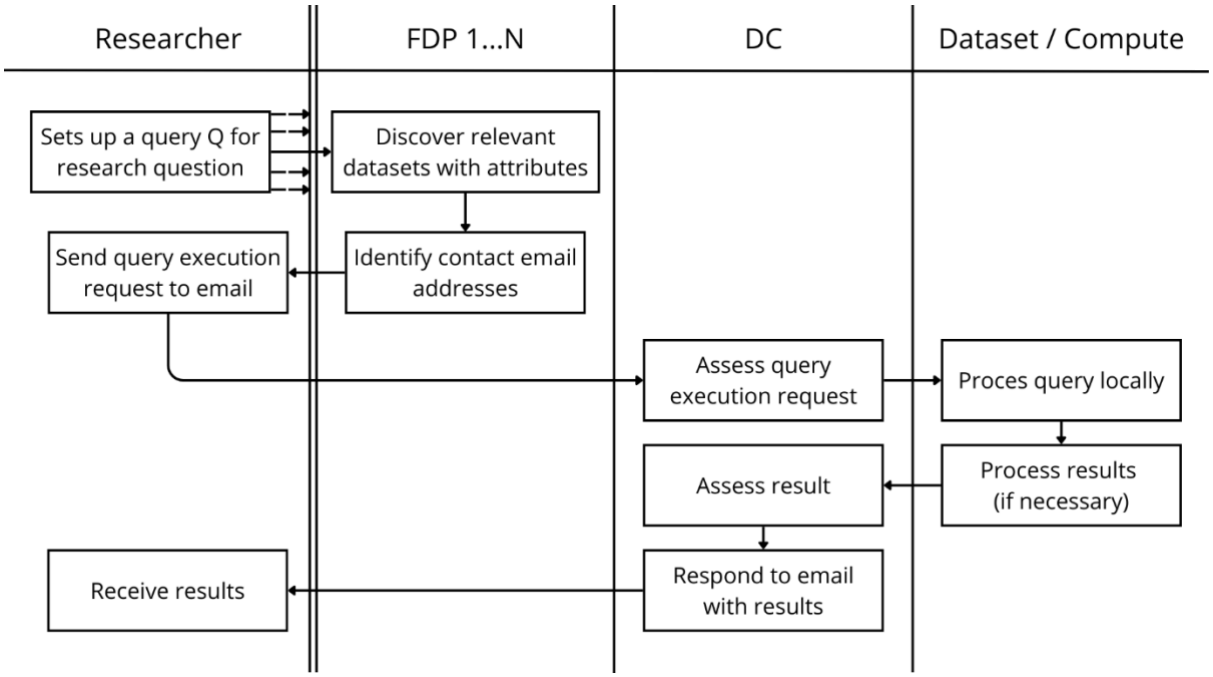


Figure 8.1. Visual process representing the flow for query execution according to the access pattern described in text.

Rue Stevin 115
1000 Brussels
admin@eepa.be
admin@vodan-africa.be
www.eepa.be
www.vodan-totafrica.info

Chapter 9: Querying across linked-data repositories

The ability to query across linked-data repositories can enable better decision-making and more informed interventions in both humanitarian and health spaces. Linked-data facilitates the interconnection of diverse data sources, enabling better access to vast amounts of information across various domains, datasets and geographies. By deploying standardised formats and protocols, organisations can integrate data from disparate repositories, such as health records, demographic databases, and incident reports. This interconnectedness not only enhances data discoverability but also allows for comprehensive analyses that can reveal critical insights into health trends, service gaps, and emerging crises. The capability to query these linked datasets also fosters collaboration among various stakeholders, such as research institutions, NGOs, clinics and governments.

Instruction for querying across multiple Allegrograph repositories

Step 1: Upload data

Create multiple repositories in Allegrograph. *Note*: if you are using the free server, you need a different e-mail address for each repository.

Step 2: Make repository accessible for anonymous users or a specific user

Note: this is a workaround for now. Make sure to revoke to anonymous user's access rights following the query if the data on the repository is sensitive.

- In each repository, go to admin users
- Click 'new user'
- Username should be: anonymous
- Do not fill in a password
- Click 'create'
- Give the anonymous access to the repository by clicking 'add new access'
- Select the repository (catalogue should be root, then the relevant repository)
- Select the permissions
- Don't forget to click 'save access'

Specific user

You can also specify a user with login credentials and the right access permissions.

Step 3: Get the SPARQL endpoint URL

The SPARQL endpoint URL should be used for querying the different repositories. This will look like: https://[URL].cloud/repositories/[repositoryname]/sparql

For example: https://ag1s5fv23e82hv78.allegrograph.cloud/repositories/Kenya-R./sparql

Make sure you have all the links for the repositories you want to query.

For making graphs make sure to replace the SELECT command with CONSTRUCT. SELECT is better for analyses, but CONSTRUCT brings in the whole triple, which enables the creation of a graph view.

Rue Stevin 115
1000 Brussels
admin@eepa.be
admin@vodan-africa.be
www.eepa.be
www.vodan-totafrica.info

Step 4: Create a query

Now you should be able to create a query across repositories. You do this with the 'service' command, and then stringing the commands together through UNION.

See the example below:

```
SELECT ?s ?p ?o
WHERE {
  {
    SERVICE
<https://ag1s5fv23e82hv78.allegrograph.cloud/repositories/Keny
a-R./sparql> {
      ?s ?p ?o .
    }
  }
  UNION
  {
    SERVICE
<https://ag1hnv7azzmpj8zr.allegrograph.cloud/repositories/SORD
042025/sparql> {
      ?s ?p ?o .
    }
  }
    UNION
  {
    SERVICE
<https://ag1texvqmg0c8w92.allegrograph.cloud/repositories/DPO2
/sparql> {
      ?s ?p ?o .
    }
  }
    UNION
  {
    SERVICE
<https://ag14sydfcedr0nhg.allegrograph.cloud/repositories/DPO1
/sparql> {
      ?s ?p ?o .
    }
  }
    UNION
  {
    SERVICE
<https://ag1nckkupkhjagdy.allegrograph.cloud/repositories/MUT2
026M1/sparql> {
      ?s ?p ?o .
    }
  }
    UNION
  {
    SERVICE
<https://ag1pjuf30af2z8mk.allegrograph.cloud/repositories/KS01
/sparql> {
      ?s ?p ?o .
    }
  }
}
```


Rue Stevin 115
1000 Brussels
admin@eepa.be
admin@vodan-africa.be
www.eepa.be
www.vodan-totafrica.info

Query for graph view, with content:

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

construct {?s ?p ?o}
WHERE {
{
SERVICE
<https://ag1s5fv23e82hv78.allegrograph.cloud/repositories/Keny
a-R./sparql> {
?s ?p ?o .
}
}
UNION
{
SERVICE
<https://ag1hnv7azzmpj8zr.allegrograph.cloud/repositories/SORD
042025/sparql> {
?s ?p ?o .
}
}
UNION
{
SERVICE
<https://ag1texvqmg0c8w92.allegrograph.cloud/repositories/DPO2
/sparql> {
?s ?p ?o .
}
}
UNION
{
SERVICE
<https://ag14sydfcedr0nhg.allegrograph.cloud/repositories/DPO1
/sparql> {
?s ?p ?o .
}
}
UNION
{
SERVICE
<https://ag1nckkupkhjagdy.allegrograph.cloud/repositories/MUT2
026M1/sparql> {
?s ?p ?o .
}
}
UNION
{
SERVICE
<https://ag1pjuf30af2z8mk.allegrograph.cloud/repositories/KS01
/sparql> {
?s ?p ?o .
}
}

FILTER(
(
(BOUND(?label) && regex(lcase(str(?label)), "kenya")) ||
```


Rue Stevin 115
1000 Brussels
admin@eepa.be
admin@vodan-africa.be
www.eepa.be
www.vodan-totafrica.info

```
(isLiteral(?o) && regex(lcase(str(?o)), "kenya"))
)
||
(
(BOUND(?label) && regex(lcase(str(?label)), "ethiopia")) ||
(isLiteral(?o) && regex(lcase(str(?o)), "ethiopia"))
)
)
}
LIMIT 200
```

Additional resources

- Demo video query across 6 repositories

Rue Stevin 115
1000 Brussels
admin@eepa.be
admin@vodan-africa.be
www.eepa.be
www.vodan-totafrica.info

Chapter 10: Use case - Creating custom tooling for knowledge graphs and FAIR-based human trafficking data

By Kai Smits, Mehul Upase, Nimish Pandey, Nina van Rossum, Parthipan Ramakrishnan, Senjuti Bala

Contact: Kai Smits <kaidsmits@gmail.com>

This chapter is designed to provide comprehensive guidance on steps towards creating FAIR data and knowledge graphs specifically for human trafficking data. As FAIR-based tooling for human trafficking research data is few and far between, this manual will look at some of the key challenges with human trafficking data, including security, non-structured data and human trafficking ontology. The manual was based on the process of creating knowledge graphs and FAIRification of research data on human trafficking collected by EEPA.

The manual is structured into several key sections. The "challenges of human trafficking data" section outlines the overarching challenges associated with human trafficking data and the importance of addressing these issues. The "Problem Exploration" section delves into the specific elements of the human trafficking problem, providing a detailed analysis. The "Technical Approach" section describes the methodologies and tools employed to implement FAIR data principles. Additionally, the "Data Protection Impact (DPI) Assessment" section rigorously examines ethical considerations in managing sensitive data.

This manual serves as a valuable resource for anyone involved in the analysis of human trafficking data, providing essential information and best practices for creating and managing FAIR data.

The challenges of human trafficking data

The analysis of human trafficking data presents significant challenges, particularly when dealing with large databases derived from sensitive and complex information. Such datasets often originate from raw in-person interviews with refugees who have been victims of human trafficking, making the data both rich and intricate.

Understanding the detailed nuances of this data requires careful handling and precision, especially since it is often collected through qualitative methods. The potential for data leaks poses serious risks to the safety and privacy of the individuals involved, necessitating a high-level ethical responsibility in data processing.

The dataset that was used for the FAIRification process in the example presented in this training manual, was a dataset created for ethnographic semi-structured interviews with refugees, which were coded and labelled. The data was largely unstructured. The data was collected from refugees and migrants, of which a majority was Eritrean, who have been trafficked for ransom in Libya.

By applying FAIR data principles—ensuring that data is Findable, Accessible, Interoperable, and Reusable—researchers can enhance the effectiveness of their investigations and contribute to more informed interventions in the fight against human trafficking. In addition, the data becomes suitable for re-use under specific conditions, which enhances opportunities for more extensive research.

The dataset

The dataset that was used to test this technical approach was an Excel sheet of 126 coded and labelled interviews with refugees. The interviews related to human trafficking along the Central Mediterranean Route. The interviewees were refugees who were taken through the human trafficking route.

The interviews were ethnographic and semi-structured. The coding-labelling process looked at key topics that were described by the refugees and migrants about aspects of their journey.

In the coding process, each stage of the participant's journey was represented by a new data row. In this row, all the data was recorded about a specific location or border crossing.

The result was a largely unstructured dataset with inconsistencies in uses of terms and with a lot of text. This presented challenges in the FAIRification of the data – structuring it according to the FAIR principles.

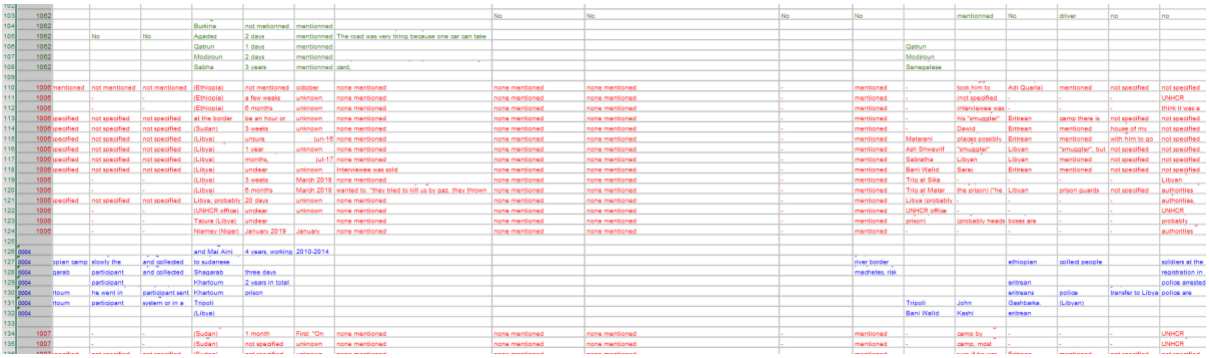


Figure 10.1. Part of the Excel sheet that was FAIRified for this manual

Technical approach

The strategy for addressing human trafficking through data analysis centers on the development of a FAIR-based ontology. This structured framework is designed to organize data ethically and facilitate meaningful insights.

Key Highlights of the Approach:

- **Deliverable Approach**: A FAIR-based ontology is created specifically for trafficking data analysis, ensuring that ethical considerations are integrated throughout the entire data analysis lifecycle. This includes everything from initial data cleaning to final visualization in tools like AllegroGraph.
- **Tools and Resources**: Data cleaning is performed manually using tools such as Excel, Python, and VBS Scripts to maintain data integrity. Additionally, CEDAR is utilized for effective vocabulary management, while AllegroGraph is employed for comprehensive data mapping.
- **Ethical Considerations**: There is a strong commitment to GDPR compliance and stringent data privacy practices. To protect sensitive information, unique IDs are used in place of refugees' names, ensuring that data integrity is maintained while safeguarding individual privacy.

Rue Stevin 115
1000 Brussels
admin@eepa.be
admin@vodan-africa.be
www.eepa.be
www.vodan-totafrica.info

This approach not only enhances the quality of data analysis but also aligns with the ethical standards necessary for handling sensitive information related to human trafficking. By implementing FAIR principles, the analysis can yield valuable insights that contribute to combating human trafficking effectively

The following steps provide more details on the comprehensive approach:

1. The first step is to identify and define the necessary data fields required to understand your human trafficking data. For the Excel datasheet used in the example for this manual, this included data fields such as trafficking locations and perpetrator names. These important data fields can be extracted from the human trafficking database.
2. Once we have identified the necessary data fields, the next step is to select an appropriate framework and template from a portal to organize this data effectively.
3. After establishing a framework and template, the focus shifts towards creating a structured framework for the ontology by identifying key concepts and their relationships.
4. After integrating the data fields, the next step involves reviewing the system to identify any missing or inadequate data. This step focuses on creating or enhancing new data fields (vocabularies) to fill these gaps, ensuring a more comprehensive representation of data related to human trafficking.
5. Following the creation or enhancement of data categories, it is essential to seamlessly incorporate these new elements into the chosen template or framework.
6. The subsequent step involves the development of a streamlined process to add a large volume of data to the system efficiently. This process should be designed to accommodate different types of data formats.
7. The system's data should comply with the principles of being FAIR. This step involves validating that the data is structured and organized in a manner that ensures its usefulness and accessibility.
8. After making sure our data is usable and accessible, it is essential to prioritize data security and compliance. We'll focus on assessing what data falls under sensitive categories and plan how to secure and manage it according to GDPR guidelines. This includes practices to protect individuals' personal information involved in human trafficking cases.
9. Initiate the process by uploading all structured data into the system in a single comprehensive upload. Subsequently, conduct a thorough check to verify the accuracy and proper fitting of all data within the system.
10. You can use tools such as AllegroGraph to visualize your data. This tool will particularly focus on displaying relationships and patterns in your data, and you can use the tool to query (ask questions) to analyse your data.

Critical decision points include data types, platform selection, data enhancement, validation procedures, and tool selection. This training manual will explain the steps in more detail.

Data Protection Impact (DPI) Assessment

A Data Protection Impact Assessment (DPIA) is a structured evaluation of the potential risks and impacts of processing personal data. Its goal is to identify, assess, and mitigate data protection risks, ensuring compliance with regulations such as the General Data

Rue Stevin 115
1000 Brussels
admin@eepa.be
admin@vodan-africa.be
www.eepa.be
www.vodan-totafrica.info

Protection Regulation (GDPR). A DPIA is useful for organizations and law enforcement agencies dealing with human trafficking victims. It helps implement robust security measures and encryption techniques to prevent unauthorized access to sensitive data. Additionally, the DPIA process involves engaging with relevant stakeholders, including representatives of the victims, to ensure a comprehensive risk assessment.

1. **Identify the Need for a DPIA**
   Determine if the data processing activity is likely to result in a high risk to the rights and freedoms of individuals, particularly in the context of sensitive data related to human trafficking victims.
2. **Describe the Information Flow**
   Document how personal data will be collected, stored, used, and shared. This includes identifying the data sources, data subjects, and any third parties involved in the processing.
3. **Identify and Assess Risks**
   Identify potential risks to the rights and freedoms of data subjects, including risks of unauthorized access, data breaches, and misuse of data. Assess the likelihood and severity of these risks.
4. **Consult with Stakeholders**
   Engage with relevant stakeholders, including representatives of victims, legal experts, and data protection officers. Gather input on potential risks and mitigation strategies.
5. **Document the DPIA Findings**
   Compile the findings of the DPIA into a report that includes the assessment of risks, proposed mitigation measures, and the rationale for decisions made during the assessment.
6. **Review and Approve the DPIA**
   Present the DPIA report to relevant decision-makers within the organization for review and approval. Ensure that the report is accessible to stakeholders involved in the data processing activity.
7. **Implement Mitigation Measures**
   Put in place the identified measures to mitigate risks before commencing the data processing activity. Ensure that all staff involved are trained on data protection practices.
8. **Monitor and Review**
   Continuously monitor the data processing activity and the effectiveness of the implemented measures. Review the DPIA periodically or when there are significant changes to the processing activity.
9. **Update the DPIA as Necessary**
   If there are changes in the processing activity, new risks are identified, or if the context changes, update the DPIA accordingly to reflect the current situation.

In conclusion, a DPIA is a systematic and transparent approach that empowers organizations to proactively identify and address potential privacy risks. This fosters a culture of responsible and ethical handling of sensitive data related to victims of human trafficking. Ultimately, the goal is to contribute significantly to protecting the human rights and dignity of those affected by such heinous crimes.

Rue Stevin 115
1000 Brussels
admin@eepa.be
admin@vodan-africa.be
www.eepa.be
www.vodan-totafrica.info

Defining the ontology

An ontology is a formal representation of a set of concepts within a domain and the relationships between those concepts, used to enable shared understanding and interoperability of data. This allows for standardizing terms related to human trafficking, facilitating data sharing, and improving data interoperability.

You can use ontology development tools such as Protégé or WebVOWL to visually model your ontology, allowing for easy editing and visualization of classes and relationships. It is important to leverage existing global ontologies and vocabularies related to human trafficking or related fields (e.g., schema.org, Dublin Core) to ensure consistency and interoperability.

If a suitable ontology does not yet exist for the purpose of your data, you can create and share one so that the community can further contribute to it. Use the following steps:

1. **Identify Key Concepts**: List the main concepts relevant to human trafficking, such as "Victim," "Perpetrator," "Trafficking Incident," "Location," and "Type of Trafficking" (e.g., labor, sexual).
2. **Establish Relationships**: Determine how these concepts relate to one another. For example, a "Victim" may be involved in multiple "Trafficking Incidents," and a "Perpetrator" may be linked to multiple victims.
3. **Create Classes and Properties**: Define classes for each key concept and create properties (both object properties and data properties) to describe relationships and attributes. For example, a "Victim" class may have properties like "hasAge," "hasGender," and "isLocatedIn."
4. **Document the Ontology**: Provide clear documentation for your ontology, including definitions for each class and property, to facilitate understanding and use by others.
5. **Test and Refine**: Validate your ontology by testing it with sample data to ensure it accurately represents the concepts and relationships in human trafficking. Refine as necessary based on feedback and testing results.
6. **Publish and Share**: Once finalized, publish your ontology in a suitable format (e.g., OWL, RDF) and share it with the community to promote its use and further development. You can share it on repositories such as GitHub, Ontoportal or another platform.

The ontology is then used to structure your data. When you collect or prepare your data, annotate it using the terms defined in your ontology. This involves mapping your data attributes to the corresponding classes and properties in the ontology. For example, if you have a dataset of trafficking incidents, you would represent each incident as an instance of the <TraffickingIncident> class and link it to instances of <Victim> and <Perpetrator> using the defined properties.

FAIRification process in detail

Data pre-processing

In the case of highly unstructured data, it is necessary to pre-process the data in order to make it readable for machines and humans alike. In the data used in this case-study, the data was highly unstructured. Even city names and details were not standardized.

Rue Stevin 115
1000 Brussels
admin@eepa.be
admin@vodan-africa.be
www.eepa.be
www.vodan-totafrica.info

As part of the process, geographical coordinates were given to locations to avoid any confusion, and individuals such as refugees, smugglers and traffickers mentioned, were given a unique ID in order to hide their identity.

Below are the steps undertaken in the data pre-processing:

1. Selecting relevant data
2. Distinguish between data and metadata
3. Sequence mapping and tracing
4. Name extraction from text
5. Geographical coordinate mapping
6. Merging journey in chronological order of events for each testimony

Based on the data you are working with, you can follow these steps and any additional steps necessary to standardize your data.

Selecting data for the pre-processing stage: Assess whether all the data in the dataset should be included in the analysis. If there are elements in the data that are not relevant to analyse, you can remove these from your selection.

- **Distinguish between data and metadata**: Data refers to the actual information collected from the interviews, such as the victims' narratives, names, locations, and events that occurred during their trafficking experiences. This is the raw content that provides insight into the victims' journeys. Metadata, on the other hand, is the information that describes the data itself. This includes details such as the date and time of the interviews, the identity of the interviewers, the format of the data, and any other contextual information that helps to understand the data's structure and origin. Metadata is essential for organizing, managing, and analysing the data effectively, as it provides context and facilitates data retrieval and interpretation. In order to make the data FAIR, a clear description of the metadata is essential.
- **Sequence mapping and tracing**: In the dataset used, there were testimonies with numbered rows that were all numbered with the unique ID of the person who gave the testimony. Those rows were serially numbered so that the rows would stay in the right order.

  The interviewers IDs were mapped to the correct interviewee IDs.

- **Name extraction from the data**: A simple python script was developed that could detect names and unique names in the given text or string and push result into the 'names' column in sheet. This list was used along with a list of names that was created by the researchers to create a data validation table that would validate the data from the given name list and return the particular keyword or word from the text or conversation.
- **Location Mapping with Geographical Coordinate's bound area**: A geographical frame was used that involved mapping a point-based boundary of geo-coordinates that were used to define a large geographical area which mostly included the northern African countries, middle-east, and other few parts of Europe as well, which featured in the data. Google Maps API was used to trigger a string search mechanism for every cell in the location columns and the Google Maps API would eventually find the approximate match for the place name with a good accuracy and then use the actual

Rue Stevin 115
1000 Brussels
admin@eepa.be
admin@vodan-africa.be
www.eepa.be
www.vodan-totafrica.info

place, city, country name in Google Maps to push the final locations in a new column in comma-separated format.

The final column for each place field contained the data in the following order:

′ < placename >′, < cityorregionname >′,′ < countryname >′

and similarly for City and Country fields. This gave a very clear list of locations, cities, and countries.

Below are the steps to implement the codes in Excel sheets. You can adjust these to your own data. Below the instructions, you can find the macros that we used as an example.

How to apply code to an Excel sheet

To execute the provided **Visual Basic for Applications (VBA)** code in an Excel sheet, follow these steps:

1. **Open Excel**: Launch Microsoft Excel and open the workbook where you want to run the code.
2. **Access the Developer Tab**:
   - If the Developer tab is not visible, you need to enable it:
     i. Go to File > Options.
     ii. In the Excel Options dialog, select Customize Ribbon.
     iii. In the right pane, check the box for Developer and click OK.
3. **Open the VBA Editor**:
   - Click on the Developer tab.
   - Click on Visual Basic in the toolbar. This will open the VBA editor.
4. **Insert a New Module**:
   - In the VBA editor, right-click on any of the items in the Project Explorer (usually on the left side).
   - Select Insert > Module. This will create a new module.
5. **Copy and Paste the Code**:
   - In the new module window, copy the provided VBA code and paste it into the module.
6. **Close the VBA Editor**:
   - After pasting the code, you can close the VBA editor by clicking the X in the upper right corner or by going to File > Close and Return to Microsoft Excel.
7. **Run the Code**:
   - Back in Excel, you can run the code by going back to the Developer tab.
   - Click on Macros.
   - In the Macro dialog, you should see the macro you inserted listed. Select it and click Run.
8. **Check the Results**:
   - After running the macro, check the specified columns in your worksheet to see the results of the code execution.

Make sure that your data is set up correctly in the specified columns (N for strings, Y for cities, and the adjacent column for countries) as indicated in the code comments.

Rue Stevin 115
1000 Brussels
admin@eepa.be
admin@vodan-africa.be
www.eepa.be
www.vodan-totafrica.info

Macros

Use the following macros:

Macro-1

```
Sub FindCityCountry()
    Dim rng As Range
    Dim cell As Range
    Dim cities As Range
    Dim city As Range
    Dim cityCountry As String

    ' Assuming column N has the strings, X has the cities and Y has the countries
    Set rng = Worksheets("Data Sheet –1").Range("Q2:Q1198")
    Set cities = Worksheets("Data Sheet –1").Range("AA2:AA139")

    For Each cell In rng
        cityCountry = ""
        For Each city In cities
            If InStr(1, cell.Value, city.Value, vbTextCompare) > 0 Then
                cityCountry = city.Value & ", " & city.Offset(0, 1).Value
                Exit For
            End If
        Next city

        ' Write the result in column O
        cell.Offset(0, 1).Value = cityCountry
    Next cell
End Sub
```

Macro-2

```
Sub FindCityCountry()
    Dim rng As Range
    Dim cell As Range
    Dim cities As Range
    Dim city As Range
    Dim cityCountry As String

    ' Assuming column N has the strings, Q has the cities and AA has the countries
    Set rng = Worksheets("Data Sheet –1").Range("N2:N1300")
    Set cities = Worksheets("Data Sheet –1").Range("Y2:Y5102")

    For Each cell In rng
        cityCountry = ""
        For Each city In cities
            If InStr(1, cell.Value, city.Value, vbTextCompare) > 0 Then
                cityCountry = city.Value & ", " & city.Offset(0, 1).Value
                Exit For
            End If
        Next city

        ' Write the result in column R
        cell.Offset(0, 1).Value = cityCountry
    Next cell
End Sub
```


Rue Stevin 115
1000 Brussels
admin@eepa.be
admin@vodan-africa.be
www.eepa.be
www.vodan-totafrica.info

Final Locations	Borders Crossed [DONE]	City / Locations Crossed [DONE]	Locations in Libya [DONE]
Mali, Senegal,	Mali, Senegal		
Mali, Niger,	Mali, Niger		
Libya, Niger, Agadez, Niger,	Libya, Niger	Agadez, Niger	
Libya, Qatrun, Libya, Qatrun, Libya	Libya	Qatrun, Libya	Qatrun, Libya
Libya,	Libya		
Libya	Libya, Niger	Sabha, Libya	Sabha, Libya
Senegal,	Senegal		
Burkina Faso,	Burkina Faso		
Mali,	Mali		
Niger, Niamey, Niger,	Niger	Niamey, Niger	
Libya, Niger, Agadez, Niger,	Libya, Niger	Agadez, Niger	
Bor, South Sudan,		Bor, South Sudan	
Libya, Gaza, Palestinian Territory,	Libya	Gaza, Palestinian Territory	
Qatrun, Libya,		Qatrun, Libya	
Mahdia, Tunisia,		Mahdia, Tunisia	
Tripoli, Lebanon,		Tripoli, Lebanon	
Tripoli, Lebanon,		Tripoli, Lebanon	
Sabha, Libya,		Sabha, Libya	
Agadez, Niger,		Agadez, Niger	
Senegal,	Senegal		
Mali,	Mali		
Niger,	Niger		
Sabha, Libya			Sabha, Libya
Tripoli, Libya			Tripoli, Libya
Libya, Ghat, Libya	Libya		Ghat, Libya
Algeria,	Algeria		
Algeria,	Algeria		
Algeria,	Algeria		
Niger,	Niger		
Benin, Guinea, Guinea-Bissau,	Benin, Guinea, Guinea-Bissau		
Benin, Niger, Nigeria,	Benin, Niger, Nigeria		

Figure 10.2. The results after macro operations.

Results after Macro Operations

Merging Journey in Chronological Order of Events for each Victim: This was the last step in the data pre-processing stage. This was necessary given that the interviews were split across multiple columns and also the number of rows used for each interviewee were not the same. The serial numbers done in the earlier step were used to make this process easier.

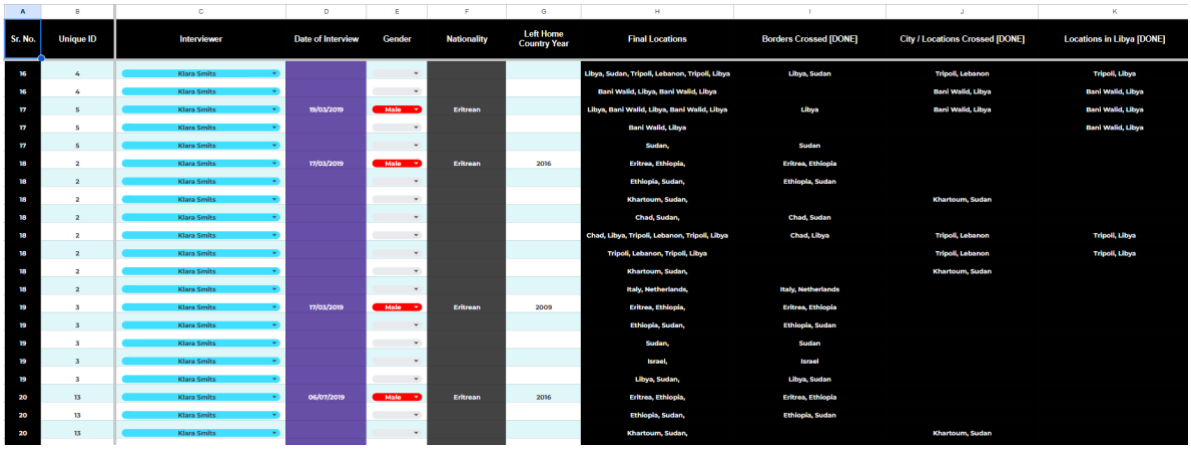
Resulting columns:

- Sr. No.
- Unique ID
- Interviewer
- Date of Interview
- Gender
- Nationality
- Left Home Country Year
- Borders Crossed
- Cities / Locations Crossed
- Countries
- Smugglers / Traffickers

Rue Stevin 115
1000 Brussels
admin@eepa.be
admin@vodan-africa.be
www.eepa.be
www.vodan-totafrica.info

- Hierarchical Relations
- Human traffickers/ Chief of places

Final processed column



Sr. No.	Unique ID	Interviewer	Date of Interview	Gender	Nationality	Left Home Country Year	Final Locations	Borders Crossed [DONE]	City / Locations Crossed [DONE]	Locations in Libya [DONE]
16	4	Klara Smits					Libya, Sudan, Tripoli, Lebanon, Tripoli, Libya	Libya, Sudan	Tripoli, Lebanon	Tripoli, Libya
16	4	Klara Smits					Bani Walid, Libya, Bani Walid, Libya		Bani Walid, Libya	Bani Walid, Libya
17	5	Klara Smits	19/03/2019	Male	Eritrean		Libya, Bani Walid, Libya, Bani Walid, Libya	Libya	Bani Walid, Libya	Bani Walid, Libya
17	5	Klara Smits					Bani Walid, Libya			Bani Walid, Libya
17	5	Klara Smits					Sudan,	Sudan		
18	2	Klara Smits	17/03/2019	Male	Eritrean	2016	Eritrea, Ethiopia,	Eritrea, Ethiopia		
18	2	Klara Smits					Ethiopia, Sudan,	Ethiopia, Sudan		
18	2	Klara Smits					Khartoum, Sudan,		Khartoum, Sudan	
18	2	Klara Smits					Chad, Sudan,	Chad, Sudan		
18	2	Klara Smits					Chad, Libya, Tripoli, Lebanon, Tripoli, Libya	Chad, Libya	Tripoli, Lebanon	Tripoli, Libya
18	2	Klara Smits					Tripoli, Lebanon, Tripoli, Libya		Tripoli, Lebanon	Tripoli, Libya
18	2	Klara Smits					Khartoum, Sudan,		Khartoum, Sudan	
18	2	Klara Smits					Italy, Netherlands,	Italy, Netherlands		
19	3	Klara Smits	17/03/2019	Male	Eritrean	2009	Eritrea, Ethiopia,	Eritrea, Ethiopia		
19	3	Klara Smits					Ethiopia, Sudan,	Ethiopia, Sudan		
19	3	Klara Smits					Sudan,	Sudan		
19	3	Klara Smits					Israel,	Israel		
19	3	Klara Smits					Libya, Sudan,	Libya, Sudan		
20	13	Klara Smits	06/07/2019	Male	Eritrean	2016	Eritrea, Ethiopia,	Eritrea, Ethiopia		
20	13	Klara Smits					Ethiopia, Sudan,	Ethiopia, Sudan		
20	13	Klara Smits					Khartoum, Sudan,		Khartoum, Sudan	

Figure 10.3. The final processed columns.

The above-given list has all the final columns that we were left with after completing the data pre-processing steps.

RDF creation

We used the cleaned dataset to create the RDF file for the Allegrograph. The RDF data was created using the **rdflib** library in Python. This involved defining a namespace, iterating over a dataset, and adding triples to an RDF graph. Each refugee's record was represented as a unique URI and linked with various properties such as hasBordersCrossed, hasGender, hasNationality, and hasTraffickerName, each associated with relevant literals. The data was serialized in Turtle format.

Key aspects of the RDF data creation:

- Utilization of rdflib for graph creation and manipulation.
- Dynamic generation of unique URIs for each refugee record.
- Inclusion of various properties to represent detailed information about each refugee.
- Serialization of the RDF graph into a TTL file for persistent storage.

Integration of RDF data with AllegroGraph

After creating the RDF data in Turtle format, it was uploaded it to AllegroGraph, a graph database designed to efficiently store RDF data and facilitate complex queries. AllegroGraph allows users to visualize the RDF data, enabling a more comprehensive analysis of the relationships and patterns within the dataset.

In order to upload your data to AllegroGraph, follow these steps:

1. **Open AllegroGraph**: Launch the AllegroGraph web interface in your browser.
2. **Select a Repository**: Navigate to the repository where you want to upload the database or create a new one.
3. **Go to the Upload Section**:
   - Click on the "Data" tab in the repository view.
   - Select "Upload Data" from the dropdown menu.
4. **Choose Your File**:

Rue Stevin 115
1000 Brussels
admin@eepa.be
admin@vodan-africa.be
www.eepa.be
www.vodan-totafrica.info

- Click on the "Choose File" button to select the RDF of your data.

5. **Configure Upload Options (if necessary):**
   - Set any specific options for the upload, such as graph name or import settings.
6. **Upload the File**:
   - Click the "Upload" button to start the upload process.

Once your data is uploaded, SPARQL queries can be used to retrieve specific subsets of data from the AllegroGraph database for analysis and visualization purposes.

```
PREFIX ex: <https://github.com/surya-parthipan/DSIP_FieldLab_7/tree/main/>

SELECT ?refugee ?trafficker1 ?trafficker2 ?location
WHERE {
  ?refugee ex:hasTraffickerName ?trafficker1 ;
           ex:hasTraffickerName ?trafficker2 ;
           ex:hasBordersCrossed ?location .
  FILTER (?trafficker1 != ?trafficker2)
  FILTER (STR(?trafficker1) < STR(?trafficker2))
}
GROUP BY ?refugee ?trafficker1 ?trafficker2 ?location
```

1,000 ROWS | DOWNLOAD RESULTS | QUERY INFORMATION

refugee	trafficker1	trafficker2	location
refugee-1043	"Muhamad"	"Tawila"	"Gambia"
refugee-1030	"Abdela"	"John"	"Agadez"
refugee-4	"Cegora"	"Kashi"	"Ethiopia"
refugee-1023	"Tuemuzgi"	"Walid"	"Egypt"
refugee-2007-1	"Tim"	"Wedi Isaac"	"Libya"
refugee-1019	"Aziz"	"El Amo"	"Tunisia"
refugee-1008	"Take"	"Walid"	"Sudan"
refugee-1027	"Aforki"	"Mahmoud Abdel Salam"	"Ethiopia"

Figure 10.4. Query and the result to get insights on locations where two traffickers worked together.

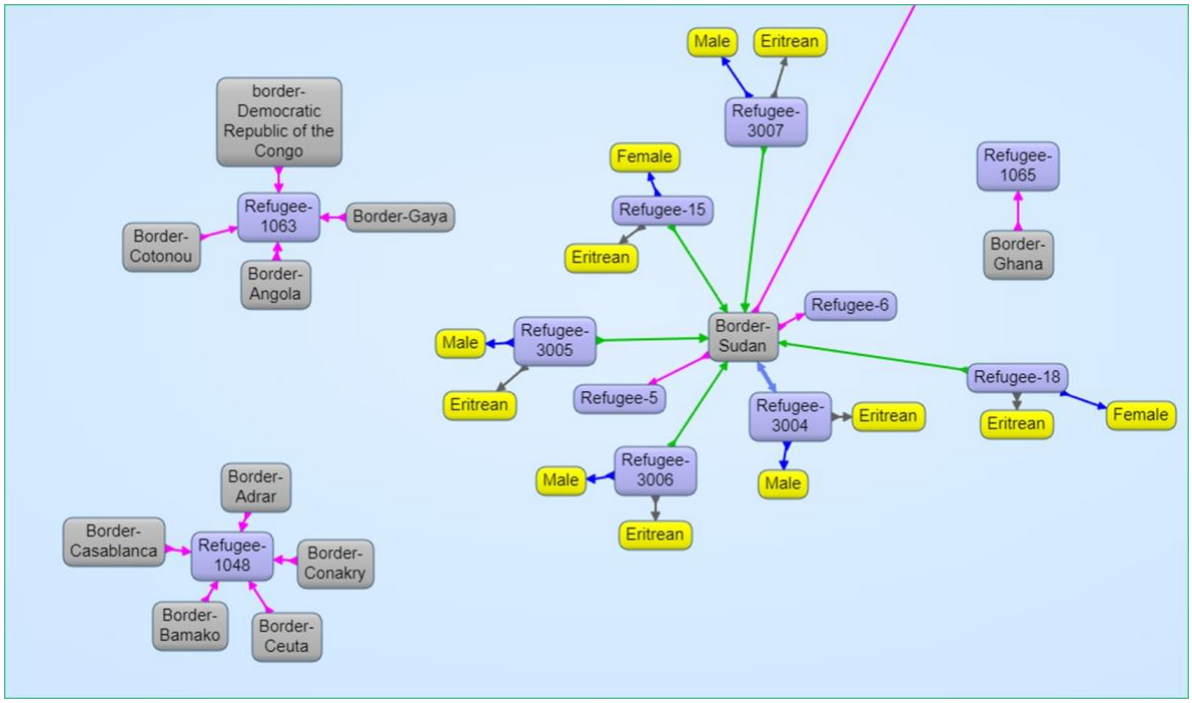


Figure 10.5. Visualization of sample triples from the created RDF in Allegrograph.

Creating SPARQL queries

Below you can find the instructions to create SPARQL queries in AllegroGraph.

1. **Accessing the SPARQL Query Interface**
   a. *Open AllegroGraph*: Launch the AllegroGraph web interface.
   b. *Select a Repository*: Choose the repository you want to query from the list of available repositories.

Rue Stevin 115
1000 Brussels
admin@eepa.be
admin@vodan-africa.be
www.eepa.be
www.vodan-totafrica.info

c. *Navigate to the SPARQL Query Interface*: Click on the "SPARQL" tab to access the query interface.

2. Basic Structure of a SPARQL Query

A SPARQL query typically consists of the following components:

- SELECT: Specifies the variables to return.
- WHERE: Contains the triple patterns that define the data to match.
- FILTER: (Optional) Applies additional constraints to the results.

Example query

```
sparql
PREFIX ex: <http://example.org/>

SELECT ?subject ?predicate ?object
WHERE {
?subject ex:hasProperty ?object .
?subject ?predicate ?object .
}
```

3. Writing SPARQL Queries

a. *Prefixes*: Use the PREFIX keyword to define namespaces for URIs to make your queries more readable.
b. *Triple Patterns*: Use the ? symbol to denote variables. A triple pattern consists of a subject, predicate, and object.
c. *Filters*: Use the FILTER function to restrict results based on conditions.

Example with FILTER

```
sparql
PREFIX ex: <http://example.org/>
SELECT ?subject
WHERE {
?subject ex:hasProperty ?object .
FILTER(?object = "SomeValue")
}
```

4. Executing SPARQL Queries

a. *Enter Your Query*: Type or paste your SPARQL query into the query editor in the SPARQL tab.
b. *Run the Query*: Click the "Execute" button to run your query.
c. *View Results*: The results will be displayed in a table format below the query editor. To visualize the results of your query in a knowledge graph, click on

Rue Stevin 115
1000 Brussels
admin@eepa.be
admin@vodan-africa.be
www.eepa.be
www.vodan-totafrica.info

the "Graph" tab in the results pane to display the data as a graph visualization.

5. **Advanced Query Features**
   a. *ORDER BY*: Sort the results based on one or more variables.

Example

```
sparql
SELECT ?subject
WHERE {
?subject ex:hasProperty ?object .
}
ORDER BY ?subject
```

   b. *LIMIT and OFFSET*: Control the number of results returned and pagination.

Example

```
sparql
SELECT ?subject
WHERE {
?subject ex:hasProperty ?object .
}
LIMIT 10 OFFSET 20
```

   c. *Construct Queries*: Use CONSTRUCT to create new RDF graphs based on the results of your query.

Example

```
sparql
CONSTRUCT {
?subject ex:hasNewProperty ?object .
}
WHERE {
?subject ex:hasProperty ?object .
}
```

6. **Best Practices**
   a. *Test Queries*: Start with simple queries and gradually add complexity.
   b. *Use Comments*: Add comments in your queries using # for better readability.

Rue Stevin 115
1000 Brussels
admin@eepa.be
admin@vodan-africa.be
www.eepa.be
www.vodan-totafrica.info

c. *Optimize Performance*: Use LIMIT, OFFSET, and FILTER to improve query performance.

7. **Resources for Further Learning**
   a. *AllegroGraph Documentation*: Refer to the official AllegroGraph documentation for detailed information on SPARQL and AllegroGraph features.
   b. *SPARQL Specifications*: Familiarize yourself with the SPARQL 1.1 specifications for advanced querying techniques.

Storing metadata

You can use AllegroGraph or another programme to manage your metadata. This is possible in the following ways:

Using Named Graphs: Create a named graph specifically for metadata. You can store metadata as triples within this graph, allowing you to organize and query metadata separately from your main data.

Adding Metadata Triples: Insert RDF triples that describe the metadata. For example, you can store information such as creation date, author, or description.

SPARQL

```
INSERT DATA {
  GRAPH <http://example.org/metadata> {
    <http://example.org/resource1> <http://purl.org/dc/elements/1.1/creator> "John Doe" .
    <http://example.org/resource1> <http://purl.org/dc/elements/1.1/date> "2023-01-01" .
    <http://example.org/resource1> <http://purl.org/dc/elements/1.1/description> "This is a sample resource." .
  }
}
```

Using RDF Properties: You can also add metadata directly to your existing resources by using specific RDF properties that represent metadata attributes.

Using a programme specifically for metadata such as CEDAR. You can define the kind of metadata templates you need and create or reuse them, as well as store your metadata in various formats.

In order to make the data Findable, the data was presented and linked to the EEPA FDP: https://fairdp.eepa.be/ in the Catalogue 'Human Trafficking Researchers' under Kai Smits.

Conclusion

The final steps towards making the data fully FAIR go beyond this chapter. Data is considered fully FAIR when it adheres to the principles of Findability, Accessibility, Interoperability, and Reusability. This means that the data is assigned a unique and

Rue Stevin 115
1000 Brussels
admin@eepa.be
admin@vodan-africa.be
www.eepa.be
www.vodan-totafrica.info

persistent identifier (such as a DOI) to ensure it can be easily found, is stored in a way that allows for easy access (preferably through open standards and protocols), is described with rich metadata that enables it to be integrated and understood across different systems and domains, and is provided with clear usage licenses and documentation that facilitate its reuse by others. By meeting these criteria, data can be effectively shared and utilized in a manner that promotes collaboration and innovation in research and beyond.

The example presented here moves towards this by showing how custom tooling for FAIR data can be applied in the human trafficking domain. The final steps of ensuring that your data is available, depends on your data. You can choose to publish (part of) your metadata in a location with a persistent identifier, so that people can see a description of what your dataset entails, and what the context of it is. That way, persons interested could contact you to see if there is a way to access it. This promotes the reuse of data and allows us to eventually query across databases to obtain new insights from the data.

Rue Stevin 115
1000 Brussels
admin@eepa.be
admin@vodan-africa.be
www.eepa.be
www.vodan-totafrica.info

Useful links

Below is an overview of relevant links for further information and training material for the implementation of the Humanitarian Data Space.

Github links

VODAN

Contains all pages on the Humanitarian Data Space and related applications.

Refugee Protection Data System (RPDS)

FAIR-aligned application designed for the structured collection, management, and semantic publication of refugee protection information.

SafeLine

A web-based incident reporting system designed to collect, organize, and analyze sexual violence data.

Emergency Diary

Emergency Diary is a decentralized web application designed to help people in emergency situations securely store, manage, and selectively share sensitive personal data with trusted humanitarian organisations.

SafeVoice

SafeVoice is a data pipeline application designed to transform raw human trafficking incident reports into semantic RDF triples.

Thinx

Thinx is a research platform designed to help social scientists and humanitarian researchers analyze human trafficking data without programming skills.

Federated Lighthouse

A prototype federated dashboard enabling the execution of predefined (routine) SPARQL queries across distributed RDF/SPARQL endpoints and presents results through a simple web-based interface.

COMPASS

This project demonstrates a real-time pipeline that converts relational healthcare data into RDF and publishes it to AllegroGraph using Ontop.

Resources of the Humanitarian Data Space

Humanitarian Data Space

This is the Humanitarian Data Space Endpoint, where queries can be executed over repositories findable in the Humanitarian Data Space

Africa Health Data Space

This website shows the resources used to establish the Humanitarian Data Space

FAIR Data Points

Rue Stevin 115
1000 Brussels
admin@eepa.be
admin@vodan-africa.be
www.eepa.be
www.vodan-totafrica.info

This page provides the links to the FAIR Data Points where repositories are located which are prepared with data models from the Humanitatian Data Space

Training modules

This is a link to the Humanitarian Data Space training resources

Technical links

FAIRdatapoint.org

FAIR Data Point software specification document

Instructions on setting up FAIR Data Points on Ubuntu

The ontology for the Humanitarian Data Space

A practical guide to FAIR data stewardship

Guide to contributing to the HDS applications

Rue Stevin 115
1000 Brussels
admin@eepa.be
admin@vodan-africa.be
www.eepa.be
www.vodan-totafrica.info