AI Platform for Integrated Sustainable and Circular Manufacturing

Deliverable

D4.1 Circular TwAIn Industrial Data Platform, Standards Ontologies - 1st version

Actual submission date: 30/06/2023

Co-funded by
the European Union

Project Number:	101058585		
Project Acronym:	Circular TwAIn		
Project Title:	AI Platform for Integrated Sustainable and Circular Manufacturing		
Start date:	July 1st, 2022	**Duration:**	36 months
D4.1 - Circular TwAIn Industrial Data Platform, Standards Ontologies - 1st version			
Work Package:	WP4		
Lead partner:	SINTEF		
Author(s):	An Lam (SINTEF), Gabriele De Luca (ENG), Iñigo Sedano Pérez (TECN), Friedrich Volz (FhG-IOSB)		
Reviewers:	Ljiljana Stojanovic (FhG-IOSB), Nenad Stojanovic (NISSATECH)		
Due date:	30/06/2023		
Deliverable Type:	OTHER	**Dissemination Level:**	PUBLIC
Version number:	1.0		

Revision History

Version	Date	Author	Description
0.1	13/04/2023	SINTEF	First release of the template
0.2	09/06/2023	SINTEF, ENG, TECN, FhG-IOSB	Ready for internal review
0.3	19/06/2023	SINTEF, ENG	Revised version after internal review
0.4	27/06/2023	SINTEF	Consolidate all changes and accept track changes
1.0	30/06/2023	SINTEF	Final coordinator review before submission

Table of Contents

List of Figures

List of Tables

Definitions and acronyms

AAS	*Asset Administration Shell*
ABAC	*Attribute-Based Access Control*
AI	*Artificial Intelligence*
API	Application Platform Interface
CA	*Consortium Agreement*
CAP	*Cognitive Automation Platform*
DAT	*Dynamic Attribute Token*
DB	*Database*
DPP	*Digital Product Passport*
DSP	*Data Spaces Protocol*
DT	*Digital Twin*
EC	*European Commission*
ETL	*Extraction Transformation and Loading*
EU	*European Union*
GA	*Grant Agreement*
GE	*Generic Enabler*
IdM	*Identity Management*
IDP	*Industrial Data Platform*
IDSA	*International Data Space Association*
IoT	*Internet of Things*
IT	*Information Technology*
ONEIROS	*Open-ended Neuro-Electronic Intelligent Robot Operating System*
PDP	*Policy Decision Point*
PEP	*Policy Enforcement Point*
PPE	*Personal Protection Equipment*
RDBMS	*Relational Database Management System*
SME	*Small Medium Enterprise*
SotA	*State of the Art*
SSI	*Self-Sovereign Identity*
SSO	*Single sign-on*
WP	*Work Package*
XAI	*eXplainable Artificial Intelligence*

Disclaimer

Executive Summary

D4.1 - "Circular TwAIn Industrial Data Platform, Standards Ontologies - 1st version" – is the first deliverable of WP4 and aims to provide an overview of the activities conducted in the first year within Task 4.1 and Task 4.2 of the Circular TwAIn Project. Task 4.1 focuses on defining data models and ontologies for materials, products, processes, and assets, while Task 4.2 focuses on the Industrial Data Platform and agreements for circularity. This deliverable serves as a documentation report for the first version of the Circular TwAIn Industrial Data Platform and Ontology Library, which facilitates the development of Digital Twins and Data Spaces for FAIR data sharing in support of circularity and sustainability.

During the preparation of this deliverable, the specific collection of relevant requirements defined by the pilots and the other technical work packages were not ready yet. Therefore, we proceeded to select and describe relevant ontologies widely applied in industrial and research projects where we have been involved. In particular, we have as reference the EU project OntoCommons[1], which is dedicated to the standardisation of data documentation across all domains related to materials and manufacturing. This is accomplished by defining, adapting, and integrating a collection of relevant state-of-the-art ontologies.

A revised version of this deliverable (**D4.4 - "Circular TwAIn Industrial Data Platform, Standards Ontologies – 2nd version"**) will include a more specific selection of ontologies, data models and standards based on a refined collection of requirements posted by circularity industrial use cases. We plan to submit this deliverable on M30.

Progress on the specification of the Circular TwAIn Ontology Library
Section 2 discusses relevant ontologies considered for the Circular TwAIn Ontology Library, which is responsible to harmonise the terminology used in the data consumed and produced in the Project. An ontology library follows a hierarchical structure where domain independent ontologies are allocated on top and middle levels and domain-specific ontologies are located at the bottom of the hierarchy. The section also explains the purpose of the ontology library and its role in achieving semantic interoperability and data sharing in Digital Twins and Data Spaces.

Evaluation of a selection of Tools for Storing, Querying and Validating the Circular TwAIn Ontology Library
Section 3 summarizes the evaluation of relevant tools for storing, querying, and validating ontologies. This evaluation was conducted to get a better understanding of the state-of-the-art and help us to select suitable tools for the Industry Data Platform presented in Section 5. In particular, we measured query and reasoning performances of a relevant selection of the state-of-the-art triple stores and OWL 2 reasoners.

Application of the Circular TwAIn Ontology Library
Section 4 discusses the idea of applying the Circular TwAIn Ontology Library in Digital Twin, Data Space, and Digital Product Passport related applications. It introduces the concept of Asset Administration Shell (AAS) and its submodels, which enable the representation of asset-related data. The section also introduces the Vocabulary Provider component and its

[1] https://ontocommons.eu/

role in managing ontologies and providing runtime lookup services. Furthermore, it highlights the potential integration of the Circular TwAIn Ontology Library in the Digital Product Passport (DPP) for data sharing towards circularity.

Industrial Data Platform Implementation

Section 5 provides an overview of the Circular TwAIn Reference Implementation of the Industrial Data Platform. It outlines the open-source technologies used in the implementation, including FIWARE, Apache, Eclipse, and IDSA. The section emphasizes the goal of promoting interoperability Circular Twins for end-to-end sustainability.

I Introduction

1.1 Scope of the document

The main objective of D4.1 is to offer a comprehensive overview of the progress achieved during the first year of the project towards the implementation and deploying of the first version of the **Circular TwAIn Industrial Data Platform and Ontology Library**, which enables the development of Digital Twins and Data Spaces for FAIR data sharing towards circularity and sustainability. This corresponds to Task 4.1 "**Definition of data models and ontologies for materials, products, processes and assets**" and Task 4.2 "**Industrial Data Platform & Agreements for Circularity**".

As a deliverable of type **"OTHER"**, the report is accompanied by a collection of software artefacts available in ***the Project's repository*** (https://github.com/Circular-TwAIn) which will be continuously maintained and updated throughout the development of the Project.

In the next iteration of T4.1 and T4.2, a new version of this deliverable (**D4.4 -** "**Circular TwAIn Industrial Data Platform, Standards Ontologies – 2nd version**") will be delivered in M30 with a more refined specification and implementation of the Circular TwAIn Industrial Data Platform and Ontology Library. This implementation will be based on a more mature collection of requirements defined by the Circular TwAIn pilots and technical work packages.

The current document receives input from the following deliverables[2]:

- D2.2 provides a detailed description of the Circular TwAIn pilots, which are critical for defining specific functional requirements for the specification and implementation of the Circular TwAIn Industrial Data Platform and Ontology Library.
- D3.1 introduces the conceptual framework and reference architecture which is driving the development of the Circular TwAIn Industrial Data Platform: a software infrastructure capable of ingesting, transforming, processing, and making available data from various sources.

The outcome of the activities performed in D4.1 will be also used as input in the implementation work packages WP4 (i.e., T4.3 and T4.4 regarding the design of Data Spaces), WP5 (i.e., Digital Twin realization), and WP6 (i.e., implementation of the pilots).

1.2 Document Structure

In addition to the current section, the deliverable also includes the following sections:

Section 2 defines the notions of ontology and ontology library and illustrates these notions with prominent examples. This section also includes a brief overview of current candidates of ontologies to be adapted and integrated in the Circular TwAIn Ontology Library.

Section 3 summarizes the evaluations of a representative selection of state-of-the-art tools for storing, querying, and validating the Circular TwAIn Ontology Library.

Section 4 discusses how the Circular TwAIn Ontology Library could enhance the capabilities provided by Digital Twins, Data Spaces and Digital Product Passports.

[2] https://www.circular-twain-project.eu/project-deliverables

Section 5 describes the progress regarding the implementation of the Circular TwAIn Reference Architecture and the design of the Industrial Data Platform.

At last, Section 6 provides the conclusions and outlook on the future work to be carried out in the next phase of T4.1 and T4.2.

2 Ontologies for building the Circular TwAIn Ontology Library

This section defines the notions of ontology and ontology library and illustrates these notions with prominent examples. This section also includes a brief overview of current candidates of ontologies to be adapted and integrated in the Circular TwAIn Ontology Library.

2.1 Ontologies and Ontology Libraries

An **ontology** can be defined as "a formal, explicit specification of a shared conceptualization" [1]. *Formal* means that the ontology is defined using a machine-readable language, which implies it is unambiguous and implicit knowledge can be automatically inferred. *Shared* means that it is agreed among several people. Finally, *conceptualization* means that an ontology represents an abstract view of the world [2].

In more common terms:

- An ontology is a digital representation artefact that can be interpreted by humans and machines.
- It defines a "digital lingua franca" agreed among several people for information exchange.
- It defines (unambiguously) a collection of terms (including classes, relations, and entities).

An ontology usually contains:

- **Individuals** (or entities) representing anything that exists such as objects, processes, and qualities.
- **Literals** (or data values), such as strings or integers, can be understood as specific types of individuals denoting data values.
- **Classes** (or types) defining sets of individuals sharing the same relations (or properties).
- **Relations** (or properties) including:
    - Subclass (or subtype) relations between classes to define which classes are more specific (or general). Same for relations themselves.
    - Unary relations between individuals and classes also known as class membership or assertions.
    - Binary (or larger) relations associating pairs (or larger number) of individuals also known as property assertions.
    - Relations associating individuals with literals (or data values) also known as data property assertions.
- A **hierarchical structure** based on defined subclass (or subtype) relations (taxonomy). A similar structure might be also available for relations.
- **Restrictions** over classes and relations defined using logical statements or axioms (machine interpretable).

Class, property and data property assertions define how individuals (data) are organized. The "schema" (or terminology) is defined by the classes, properties, and their definitions and (subclass or sub property). In this document, Section 2 is only concerned to the definition of terms and not to how data is stored.

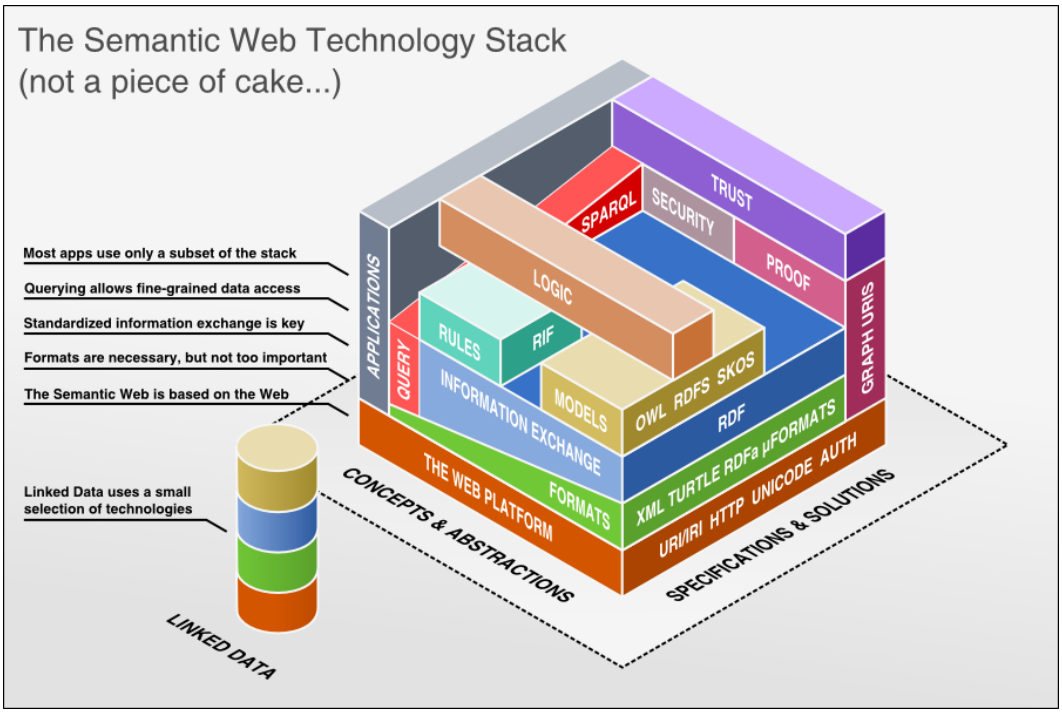


Figure 1: Semantic Web Technology Stack[3].

To define classes, relations and restrictions over them, we need a (machine-readable) modelling language. The World Wide Web Consortium (W3C)[4], founded in 1994 and lead by Tim Berners-Lee, is responsible for the development of several relevant standards (known as "W3C recommendations")[5] for the World Wide Web (or Web). W3C has published several standards relevant for the implementation, validation and publication of ontologies and data on the Web. These standards, known as "Semantic Web recommendations" (Figure 1), include two ontology modelling languages, **OWL**[6] and **RDFS**[7], one query language, **SPARQL**[8], one constraint language for data validation known as **SHACL**[9], and one (graph) data model, **RDF**[10].

An example of a successful implementation and application of ontologies is **Wikidata**[11], a free, general purpose, multilanguage, and collaboratively edited ontology hosted by Wikimedia Foundation[12]. It defines more than 93 million terms, and it is currently maintained by more than 23,000 active editors[13]. It is implemented using W3C standards such as RDF, OWL and SPARQL. Companies such as Apple, Amazon, Google, and IBM are integrating Wikidata to improve the functionality of some of their products including, Apple Siri, Amazon product catalogue, Google search and IBM Watson.

[3] http://denigma.de/data/entry/semantic-web-technology-stack
[4] https://www.w3.org/
[5] https://www.w3.org/standards/semanticweb/
[6] https://www.w3.org/TR/owl2-syntax/
[7] https://www.w3.org/TR/rdf-schema/
[8] https://www.w3.org/TR/sparql11-query/
[9] https://www.w3.org/TR/shacl/
[10] https://www.w3.org/TR/rdf11-concepts/
[11] https://www.wikidata.org/
[12] https://www.wikimedia.org/
[13] https://www.wikidata.org/wiki/Wikidata:Statistics/es

In the Biomedical community, which is one of the scientific and industrial communities that have contributed more on the development of ontologies, the **Open Biological and Biomedical Ontologies (OBO) Foundry**[14] has become a reference from an organizational and technical point of view. The mission of the OBO Foundry is to develop a family of interoperable ontologies that are both logically well-formed and scientifically accurate. To achieve this, the members of the OBO Foundry has defined a collection of principles[15], which includes the adoption of OWL as a reference ontology language. The ontologies implemented by the members of the OBO Foundry are organized in a hierarchical import structure, where domain independent ontologies are located at the top of the hierarchy and use case specific ontologies are located at the bottom of the hierarchy. This approach for organizing ontologies to facilitate interoperability is also known as an **ontology library** (Figure 2). This approach was also adopted in the OntoCommons project[16], which aims to facilitate interoperable, harmonised, and standardised data for industry through ontologies, as well as the Industry Ontology Foundry (IOF)[17]. For Circular TwAIn, we will also organize the ontologies as an ontology library, named as the **Circular TwAIn Ontology Library**.

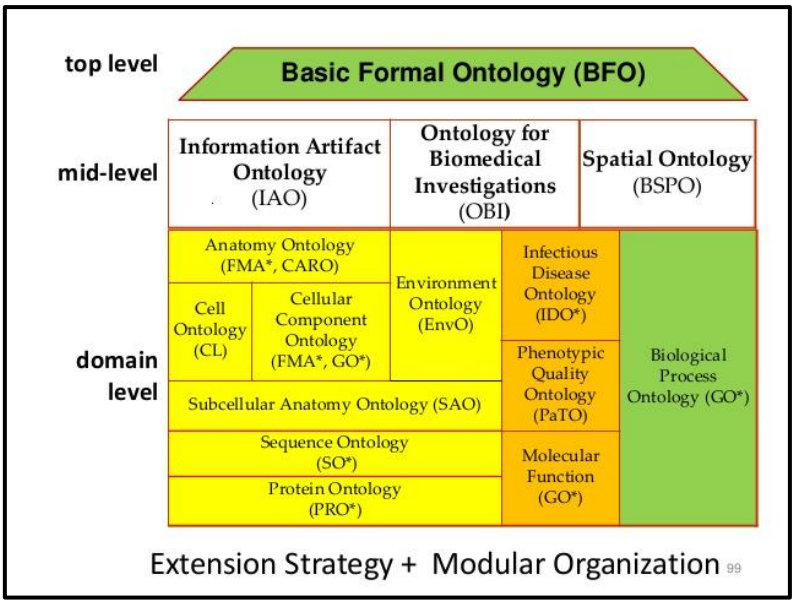


Figure 2: OBO Foundry ontology library [3].

2.2 Ontology-specific requirements

To guide the selection of suitable ontologies for the Circular TwAIn ontology library, the following (Project independent) requirements defined by Martin-Recuerda et al. [6] have been considered[18]. We are currently discussing project specific requirements that we will help us to refine the selection of ontologies presented in this document.

R1. Public license

[14] https://obofoundry.org
[15] https://obofoundry.org/principles/fp-002-format.html
[16] https://ontocommons.eu/
[17] https://industrialontologies.org
[18] *These requirements only refer to the terminological definitions ("schema") and no to the data hosted by Circular TwAIn. The key word MUST are to be interpreted as described in RFC 2119.*

A candidate ontology MUST be distributed under a public copy-right licence without commercial limitations.

R2. Actively maintenance

A candidate ontology MUST be actively maintained.

R3. Coverage

A candidate ontology MUST contribute to define all the necessary terms (classes and properties) for Circular TwAIn pilots.

R4. Modular

A candidate ontology covering different topics MUST be defined as a collection of self-contained modules that only define the relevant terms (classes and properties) for a particular topic[19].

R5. OWL 2 Direct Semantics[20]

A candidate ontology MUST be implemented using OWL 2 Direct Semantics[21], which includes less expressive profiles such as OWL 2 EL, RL, and QL.

R6. OWL tools compliance

A candidate ontology MUST be compliant with "de facto" reference tools for OWL 2 such as (Web) Protégé and OWLAPI.

R7. Reasoning performances[22]

A candidate ontology MUST ensure acceptable reasoning performances when validating consistency and computing the classification of classes and properties using the state-of-the-art OWL reasoners.

2.3 Analysis of relevant ontologies for implementing the Circular TwAIn ontology library

In this section, we will briefly describe some relevant ontologies that we are considering for the Circular TwAIn ontology library. The description of each ontology will include some basic statistics about their number of components. This will help us to get a basic understanding of their size and complexity. We plan to improve and extend this selection of ontologies based on Project specific requirements which are currently under discussion. We will present the result of this refining process in deliverable D4.4 - "Circular TwAIn Industrial Data Platform, Standards Ontologies – 2nd version", which will be submitted in M30.

2.3.1 Top-level ontologies

In this section, we will briefly present three relevant candidates for top ontologies to be included in the Circular TwAIn Ontology Library: BFO, DOLCE and ISO 15926-14. The EU

[19] *Requirement R4 aims to avoid the implementation of monolithic ontologies difficult to maintain.*
[20] *Requirement R5 is a prerequisite to fulfil requirements R6 and R7*
[21] https://www.w3.org/TR/owl2-direct-semantics/
[22] *Testing consistency and computing implicit subclass and subproperties relations can be very challenging with some ontologies [7]. Requirement R7 aims to avoid that too difficult ontologies from a reasoning point of view are included in the Circular TwAIn ontology library.*

project OntoCommons also chose these three ontologies as reference top ontologies in the industry[23]. Notice also that BFO is the top ontology in the OBO Foundry ontology library.

Basic Formal Ontology (BFO)[24]

BFO is a top-level ontology with more than 19 years of history. The development started in 2002 and continue since then. The BFO taxonomy of entities are organized into "continuants", entities existing at a given time, and "occurrents", processes unfolding through time. BFO was originally defined in first-order logic (FOL). The latest implementation BFO was released in 2020[25], and it was included in the standard ISO/IEC 21838-2[26]. This release includes FOL and OWL 2 axiomatizations. The OWL 2 version contains 274 logical axioms, 36 classes, 64 object properties and 10 annotation properties. BFO 2020 fulfils all the requirements listed in Section 2.2.

As mentioned earlier, BFO is very well supported by the Biomedical community as part of the Open Biological and Biomedical Ontology (OBO) Foundry. In April 2021, at least 145 ontologies of the 201 ontologies supported by OBO Foundry, were under active development [8].

BFO is also supported by the Industrial Ontology Foundry (IOF)[27] that aims to extend BFO with specific new ontologies to support the manufacturing and engineering industry needs and advance data interoperability. The Common Core Ontologies (CCO)[28] comprise eleven ontologies covering information entities, agents, qualities, events, information artifacts, time, geospatial, units of measure, currency units, extended relations, and modal relations.

Descriptive Ontology for Linguistic and Cognitive Engineering (DOLCE)[29]

DOLCE is a foundational ontology first released in 2002. DOLCE is formally specified in first-order logic (FOL), but given the popularity of the OWL language, at least two simplified versions defined in this language have been released: DOLCE Lite[30] and Dolce+DnS Ultralite[31] (DUL). At the highest level, DOLCE distinguishes four core classes: objects (called endurants), events (perdurants), qualities and abstracts.

DUL, which is a very simplified version of the original DOLCE, is the largest top ontology discussed in this section. It contains 600 logical axioms, 76 classes, 107 object properties, 5 data properties and 5 annotation properties. DUL fulfils all the requirements listed in Section 2.2.

DUL is a popular top ontology in the Semantic Web community. The ontologies PROV-O and SOSA/SSN (also considered for the Circular TwAIn Ontology Library and discussed below) import DUL and define explicit mappings to DUL terms.

---

[23] *Otten et al. "D2.2 - TLO/MLO Landscape Analysis Report". OntoCommons 2021.*
[24] https://basic-formal-ontology.org/
[25] https://standards.iso.org/iso-iec/21838/-2/ed-1/en/
[26] https://www.iso.org/standard/74572.html
[27] https://www.industrialontologies.org/
[28] https://github.com/CommonCoreOntology/CommonCoreOntologies
[29] http://www.loa.istc.cnr.it/index.php/dolce/
[30] http://www.loa.istc.cnr.it/old/ontologies/DLP3971.zip
[31] http://ontologydesignpatterns.org/wiki/Ontology:DOLCE+DnS_Ultralite

ISO 15926-14

ISO 15926 is an ISO standard defined for the representation of process industry facility life-cycle information. It is organised as a series of separately published parts, where Part 2 (ISO 15926-2)[32], published as a standard in 2003, is the most fundamental of these parts. ISO 15926-2 specifies a generic, conceptual data model, and it is implemented using the language EXPRESS[33]. In 2016, Part 12 (ISO 15926-12) was published to produce a new specification of ISO 15926-2 using OWL 2 under RDF-Based Semantics[34]. We did not consider ISO 15926-12 as a top ontology for Circular TwAIn because it does not fulfil requirements R5, R6 and R7 listed in Section 2.2.

Part 14 (ISO 15926-14)[35], which is under review by ISO, represents an attempt to define a subset of ISO 15926-2 using OWL 2 under direct model-theoretic semantics[36]. ISO 15926-14 fulfils the requirements defined in Section 2.2, including R1, because ISO copyrights restrictions only applies to documents and not to related digital artefacts as ontologies. The ISO 15926-14 taxonomy of entities are organized around three disjoint classes: "Activity", "Aspect" and "Object". The former, the class Activity, represents processes in which objects participate. The class Aspect defines entities that represent attributes and properties of objects. The class Object is created to specify the main, independent, entities of interest.

The latest release of ISO 15926-14 can be downloaded from the POSC Caesar Association (PCA) website[37] and it contains 143 logical axioms, 33 classes, 63 object properties, 3 data properties and 37 annotation properties. ISO 15926-14 fulfils all the requirements listed in Section 2.2.

ISO 15926-14 is gaining popularity thanks to the READI project[38], which is supported by relevant companies in the Oil & Gas industry. Siemens AG has also adopted ISO 15926-14 for its corporative ontology library.

2.3.2 Mid-level ontologies

In this section we will discuss some relevant domain independent ontologies that we are considering for the Circular TwAIn Ontology Library. These ontologies should be placed under the top ontology and before the domain specific ontologies in the hierarchical import structure. In particular:

- EMMO for physics, chemistry, and materials science concepts.
- IAO for representing "information content entities".
- SKOS for defining simple data dictionaries and mappings.
- PROV-O for representing provenance information.
- QUDT and OM2 for representing quantity kinds (e.g., weight, velocity, temperature, etc), units of measure (e.g., kg, m/s, °C) and quantity values.

[32] https://www.iso.org/standard/29557.html
[33] https://www.iso.org/standard/38047.html
[34] https://www.w3.org/TR/owl2-rdf-based-semantics/
[35] https://www.iso.org/standard/75949.html
[36] https://www.w3.org/TR/owl2-direct-semantics/
[37] http://rds.posccaesar.org/ontology/lis14/ont/core
[38] https://readi-jip.org/

- SOSA/SSN for representing observations, sensors, and actuators.

Elementary Multiperspective Material Ontology (EMMO)[39]

EMMO, developed by the European Materials Modelling Council (EMMC), is an ontology that offers a formal means of describing the core principles of physics, chemistry, and materials science. Its primary objective is to establish semantic interoperability by creating a shared foundation for delineating materials, models, and data across various domains.

EMMO functions as a comprehensive framework, encompassing predefined classes and axioms (ontology) devised by experts from EMMC. By employing ontological elements (individuals), EMMO allows end users from industry, research, and academia to represent tangible real-world entities (such as materials and devices), models, and properties in a standardized manner. This standardized representation facilitates seamless interactions and exchanges (of data, software, and knowledge) among the diverse communities and stakeholders involved in material modelling and characterization.

The latest version of EMMO (1.0.0-beta3) was released in 2019. It contains 449 logical axioms, 199 classes, 70 object properties, 1 data property, 21 annotation properties and 1 individual.

Information Artifact Ontology (IAO)[40]

IAO was created to support the needs of the Ontology for Biomedical Investigations (OBI)[41] and it is integrated in the OBO Foundry ontology library as an extension of BFO. The main design goal behind IAO was to serve as a domain-neutral resource for the representation of types of "information content entities" (ICEs) such as documents, databases, and digital images [9] which can be relevant for the artifacts shared within the data space of Circular TwAIn Project.

IAO is released as a single file that includes also other ontologies such as BFO and OMO[42]. The latest release (7-Nov-2022) contains 523 logical axioms, 264 classes, 53 object properties, 4 data properties, 72 annotation properties and 20 individuals. IAO fulfils all the requirements listed in Section 2.2.

Simple Knowledge Organization System (SKOS)[43]

SKOS is an RDF vocabulary suitable for implementing thesauri, classification schemes, subject heading lists, taxonomies, terminologies, glossaries, and other types of controlled vocabulary also known as Knowledge Organisation Systems (KOSs). A W3C recommendation since 2009, the data model for SKOS was formally defined as an OWL Full ontology (normative)[44], but a non-normative version in OWL DL was also released[45].

SKOS defines three disjoint top classes: "Concept", "Concept Scheme" and "Collection". The individuals of the class "Concept" represent abstract entities, which might include ideas

[39] https://emmo-repo.github.io/
[40] https://github.com/information-artifact-ontology/IAO
[41] http://obi-ontology.org/
[42] https://github.com/information-artifact-ontology/ontology-metadata
[43] https://www.w3.org/TR/skos-reference/
[44] https://www.w3.org/TR/skos-reference/#L895
[45] https://www.w3.org/2009/08/skos-reference/skos-owl1-dl.rdf

or units of thought. It is up to each practitioner to determine their interpretations [10]. SKOS concept schemes are informal hierarchies of SKOS concepts. Each concept scheme is represented by an individual of the class "Concept Scheme". The property "inScheme" is used to relate a SKOS concept to one or more SKOS concept schemes. The informal hierarchical relations between SKOS concepts in a SKOS concept scheme are defined using SKOS semantic properties such as "broader". When it is required to group SKOS concepts, but no hierarchical relations are expected, SKOS provides the notion of SKOS collections. A SKOS collection is represented by an individual of the class "Collection", where the relevant SKOS concepts are related to this individual using the properties "member" (for unordered sets) or "memberList" (for ordered sets). Notice that SKOS semantic and mapping properties only apply to SKOS concepts defined in SKOS concept schemes and not in SKOS collections.

The latest release of the non-normative OWL version of SKOS was published in 2009. It contains 41 logical axioms, 4 classes, 17 object properties, 1 data property and 14 annotation properties.

PROV-O[46]

PROV is a W3C standard that defines a data model for the interchange of provenance information in the Web. The latest version includes the PROV ontology (PROV-O)[47], which was defined using OWL 2 language. This ontology defines three core classes: "Entity", "Activity" and "Agent". The class Entity defines any kind of "things" real or imaginary including physical and digital objects (e.g., a machine, a laptop, etc.). The class Activity represents processes that used or generated entities (e.g., start production, disassemble the laptop, etc.). The class Agent specifies "things" that receive attribution for entities and are responsible of activities. Some agents can be also entities.

The latest release of PROV-O was published in 2013. It contains 181 logical axioms, 31 classes, 44 object properties, 6 data properties and 22 annotation properties.

Quantities, Units, Dimensions and Types ontology (QUDT)[48] and the Ontology of units of Measure (OM)[49]

Ontologies of units of measure are crucial to facilitate interpretation, exchange, and process of quantitative information in Circular TwAIn. For the Circular TwAIn Ontology Library, we are considering two well-known ontologies of units of measure: QUDT and OM. Both ontologies provide definitions for a broad variety of kinds of quantities (or quantity kinds) and units of measure, which include not only the quantity kinds and units defined by the International System of Quantities (ISQ)[50], but also the Imperial system[51] and the US customary measurement system[52].

[46] https://www.w3.org/TR/prov-o/
[47] https://www.w3.org/TR/prov-o/
[48] https://github.com/qudt/qudt-public-repo
[49] https://github.com/HajoRijgersberg/OM
[50] https://www.iso.org/standard/30669.html
[51] https://en.wikipedia.org/wiki/Imperial_units
[52] https://en.wikipedia.org/wiki/United_States_customary_units

Keil et al. [11] and Martin-Recuerda et al. [6] provide a detailed analysis of both ontologies. As the latter publication discussed, both ontologies do not fulfil requirements R7 and QUDT does not fulfil requirements R5 and R6 also. These ontologies can be easily modified, however, to ensure that the requirements are fulfilled. We plan in Circular TwAIn to produce a modified version of the latest release of the selected units measure ontology for the Circular TwAIn Ontology Library.

Sensor-Observation-Sampling-Actuator Ontology (SOSA) and Semantic Sensor Network (SSN)[53]

The SSN ontology is published as a W3C standard to primarily describe sensors and their observations over properties that belongs to "things" (known as features of interests). SSN was released with a lightweight ontology defined as a core module of SSN and named as the Sensor, Observation, Sample, and Actuator (SOSA) ontology [12].

SOSA defines the following core classes: "Feature of Interest", "Sensor", "Observation", "Actuator", "Result" and "Platform". The class Feature of Interest represents things whose properties are being observed (by a sensor) or manipulated (by an actuator). The class Sensor represents devices or agents that detect changes in a property of a feature of interest. The class Observation represents an activity for estimating or calculating the value of a property of a feature of interest. The class Actuator represents devices that have the capability of changing the state of the world. The class Result stores the result of an observation or an actuation. The class Platform defines entities that host other entities such as sensors, actuators, samplers (a device that produces samples) and other platforms.

The latest release of SOSA contains 12 logical axioms, 16 classes, 2 data properties, 17 annotation properties and 21 object properties, where 9 object properties are inverse properties. SOSA fulfils all the requirements listed in Section 2.2.

2.3.3 Domain specific ontologies

In this section, we provide an overview of domain specific ontologies for the Circular TwAIn Ontology Library. In particular, the following topics are covered by these ontologies:

- Physics, chemistry, and materials science concepts.
- Battery Interface and Value Chain.
- General Process Characteristics.

As mentioned earlier, more ontologies will be developed/introduced, or the upper-level ontologies will be extended to cover the concepts in the pilots of the Project. This process will be done based on sample data and requirements for data representation and exchange of the pilots.

Battery INterFace Ontology (BattINFO)[54]

BattINFO has been developed for the Battery Interface Genome – Materials Acceleration Platform project (BIG-MAP) and BATTERY 2030+. It is constructed on top of EMMO ontology. BattINFO's primary objective is to formalize the existing knowledge pertaining to

[53] https://www.w3.org/TR/vocab-ssn/
[54] https://github.com/BIG-MAP/BattINFO

battery interfaces. This formalization supports the development of computational tools and facilitates the utilization of interoperable data within the battery development domain.

In BattINFO, the definitions provided are largely based on established standards outlined by the International Union of Pure and Applied Chemistry (IUPAC) or preeminent textbooks in the field. The design of BattINFO's classes and their interconnections is guided by three key principles: (i) to be scientifically rigorous and accurate, (ii) to reflect current battery orthodoxy and dominant jargon, and (iii) to be flexible to describe a range of battery chemistries, not only Li-ion.

The latest version of BattINFO (0.1.0) was released in 2021. It has 144 logical axioms, 140 classes, 5 object properties and 11 annotation properties.

Ontology for the Battery Value Chain (BVC)[55]

While BattINFO primarily concentrates on internal components and chemical processes within batteries, BVCO is specifically dedicated to the higher-level process chains involved in material processing and manufacturing. As a result, BVCO complements BattINFO by addressing higher-level aspects. Furthermore, BVCO includes the fundamental definition of a battery as a system.

BVCO was developed based on EMMO and GPO ontologies. It has 277 logical axioms, 182 classes, 11 object properties, and 6 annotation properties.

General Process Ontology (GPO)[56]

GPO aims to model processes which are "holistic perspective elements that transform inputs/educts (matter, energy, information) into output/products (matter, energy, information) with the help of tools (devices, algorithms)". They can be decomposed into sub-processes and have predecessor and successor processes. This ontology can be used to model processes within different domains such as logistics, information and data processing, manufacturing, and mining.

GPO has 268 logical axioms, 200 classes, 24 object properties, and 9 annotation properties.

2.4 Positioning of Circular TwAIn Ontology Library within the Circular TwAIn Project

The Circular TwAIn Ontology Library is a collection of ontologies organized in a hierarchical import structure that defines data models and terminologies to represent materials, products, processes, and assets required by the Circular TwAIn pilots.

The Circular TwAIn Ontology Library aims to provide the core building block to create common information models and terminologies for the Circular TwAIn pilots. It is also a reference component for the Circular TwAIn Reference Architecture. This ontology library and the required applications for storing, querying, and validating ontologies will be included into the Circular TwAIn Industrial Data Platform (introduced in Section 5.3). The combination

[55] https://github.com/Battery-Value-Chain-Ontology/ontology
[56] https://github.com/General-Process-Ontology

of the ontology library and a selection of semantic applications will facilitate the implementation of solutions towards semantic interoperability for Digital Twins and Digital Product Passports and enable seamless data sharing for the Circular TwAIn Industrial Data Space.

Following the description of the task T4.1 in the DoA, we are evaluating suitable and relevant industrial ontologies including ISO 15926-14, BFO, DOLCE, EMMO or QUDT. Whenever is possible, we are also considering relevant industrial standards (such as IEC 61360, IEC 61987, ISO 15926 or IEC 81346). When a more mature collection of requirements from the pilots and the other technical work packages become available, we will make a more refined selection of ontologies and standards and we will proceed to their integration to produce a first version of the Circular TwAIn Ontology Library.

To illustrate how the Circular TwAIn Ontology Library can be applied in the Project to improve data understandability, we discuss a concrete example using data and metadata defined by the PETROCHEMICAL pilot. Figure 3 shows two tables. The first table provides a short metadata specification and the second table the readings of several sensors.

Sensor ID	Info	Unit
2201.LIC252.PID_PLA.PV	C-204 Middle Tray Valve	%
2201.TI2501.DACA.PV	C-204 Recycle Gas Temperature	°C
2201.TI2502.DACA.PV	C-204 Top Out Recycle Gas Temperature	°C
2201.PI251.DACA.PV	C-204 Top Pressure	kg/cm2

	A	B	C	D	E
1	TimeStamp	2201.LIC252.PID_PLA.PV	2201.TI2501.DACA.PV	2201.TI2502.DACA.PV	2201.PI251.DACA.PV
2	2022-01-01T00:00:00	20.17954826	36.23183441	26.49759674	0.322400272
3	2022-01-01T01:00:00	19.8398819	36.44403839	26.73258591	0.323370606
4	2022-01-01T02:00:00	20.08555222	36.33295059	26.50614166	0.322277427
5	2022-01-01T03:00:00	20.10697937	36.40345001	26.40573692	0.320857286
6	2022-01-01T04:00:00	19.9887085	36.54657745	26.54459381	0.322127879
7	2022-01-01T05:00:00	19.87942886	36.18768692	26.18142891	0.327410609
8	2022-01-01T06:00:00	19.71526527	36.3543129	26.1600666	0.322416306
9	2022-01-01T07:00:00	20.14368439	36.58716583	25.98168945	0.322395384

Figure 3: Example of sample data from the PETROCHEMICAL pilot (bottom) together with its description (top).

We can observe that the dataset was described using proprietary terms and codes, which make this data difficult to understand to non-domain experts. Instead of using proprietary terms and codes, we can use terms defined by suitable ontologies such as SOSA and QUDT introduced in Section 2.3. Figure 4 shows a possible representation of a subset of the sample data represented as an RDF graph, where proprietary terms have been replaced by terms defined by SOSA and QUDT ontologies. Please notice that this is just an example of semantic transformation using OWL 2 ontologies. Depending on specific needs of the pilots, we can provide services to (for instance):

- Convert data into RDF graphs as shown in Figure 4.
- Replace or map the description (meta-data) of the data with the terms defined in the ontologies.

The first option can be useful for data representation in the Data Space or Digital Passport while the second option can be used to retrieved semantic identification for the Digital Twin data properties (e.g., AAS Semantic ID).

```
<tempSensor/2201> rdf:type sosa:Sensor ;
  rdfs:comment "C-204 Top Out Recycle Gas Temperature"@en ;
  sosa:observes  <tempSensor/2201#temperature> ;
  sosa:madeObservation <tempSensor/2201/0001>, <tempSensor/2201/0002> .

<Observation/0001> rdf:type sosa:Observation ;
  sosa:observedProperty  <machine/134> ;
  sosa:madeBySensor <tempSensor/2201> ;
  sosa:hasResult [
    rdf:type qudt-1-1:QuantityValue ;
    qudt-1-1:numericValue "26.49759674"^^xsd:double ;
    qudt-1-1:unit qudt-unit-1-1:DEG_C ] .

<Observation/0002> rdf:type sosa:Observation ;
  sosa:observedProperty  <machine/134> ;
  sosa:madeBySensor <tempSensor/2201> ;
  sosa:hasResult [
    rdf:type qudt-1-1:QuantityValue ;
    qudt-1-1:numericValue "26.73258591"^^xsd:double ;
    qudt-1-1:unit qudt-unit-1-1:DEG_C ] .
```

Figure 4: Example of PETROCHEMICAL temperature data represented by SOSA ontology.

In both cases, mappings between the data features (e.g., columns of sample data) and the entities of the ontologies need to be defined. This could be done manually with the RDF Mapping Language (RML)[57]. However, because dataset may have many features and ontologies usually cover numerous concepts, identifying relevant semantic entities for a particular dataset can be very time-consuming and might require knowledge from different domains. A promising approach to reduce the manual efforts required to identify equivalent entities is to applying deep learning algorithms, and in particular, Pre-trained Large Language Models (e.g., BERT [4], GPT [5]). This approach will be evaluated in the next iteration of Task 4.1. The ultimate goal is to enable semantic interoperability in the Circular

[57] https://rml.io/specs/rml/

TwAIn Project and the Industrial Data Platform we are building. We specifically discuss the usage of ontologies to support semantic interoperability in Digital Twins and Data Spaces in Section 4.

To conclude this section, we would like to emphasize that storing, querying, editing and validating ontologies require specific types of applications developed by the members of the Semantic Web community. Two relevant types of applications are relevant for this Project: RDF databases (a.k.a triplestores) and OWL 2 reasoners. Triplestores are responsible for storing and querying ontologies and OWL 2 reasoners are very useful to ensure correctness of OWL 2 ontologies. Because there is large variety of triplestores and OWL 2 reasoners, we have conducted two evaluations, one for each type of tool. In the first evaluation, we tested the query performances of a selection of relevant RDF triplestores over very large ontologies. We also evaluated their ability to export and import large ontologies, which is relevant in this Project where data would be stored using different persistent mechanisms. The second evaluation tested loading and reasoning performances of relevant OWL 2 reasoners. In particular, testing consistency or computing the classification of an ontology is critical to ensure correctness. In the next section, we summarize the main results of these evaluations.

3 Tools for Storing, Querying and Reasoning with ontologies selected to the Circular TwAIn Ontology Library

The Circular TwAIn Ontology Library is core component of the Circular TwAIn Industry Data Platform as it is defined as a hierarchical collection of OWL 2 ontologies. To implement, store, query and validate this ontology library, specific tools are required. Given the variety of tools created by the Semantic Web community for each category, it can be difficult to select a suitable tool for the specific needs of a Project or application. In this section, we summarize the results of two evaluations that we conducted. The evaluations have been recently presented at the 20th Extended Semantic Web Conference (ESWC) 2023 [13] and at the 1st International Workshop on Data Management for Knowledge Graphs (DMKG) 2023 [14]. In the first evaluation, we tested the import, export and query performances of a selection of relevant state-of-the-art RDF triplestores over very large ontologies. The second evaluation tested loading and reasoning performances of relevant state-of-the-art OWL 2 reasoners using a diverse and large collection of ontologies.

3.1 An Evaluation of a Representative Selection of RDF Triplestores

In this section, we present an evaluation of the performance of five representative RDF triplestores, including Ontotext GraphDB[58], Apache Jena Fuseki[59], Amazon Neptune[60], Oxford Semantic Technologies RDFox[61] and Stardog[62], and one experimental SPARQL query engine, QLever[63]. We compare importing time, loading time and exporting time using a complete version of the knowledge graph Wikidata[64], one of the largest and more relevant open knowledge graphs. We also evaluate query performances using 328 queries defined by Wikidata users. To test the scalability of the selected triple stores, we used a prominent synthetic benchmark: SP^{2}Bench[65]. We observed that most of the systems we considered for the evaluation were able to complete the execution of almost all the queries defined by Wikidata users before the timeout we established. We noticed, however, that the time needed by most systems to import and export Wikidata might be longer than required in some industrial and academic projects, such as Circular TwAIn.

3.1.1 Triplestores

To ensure that our limited selection of triplestores is representative and diverse, the following triplestores (Table 1) were evaluated: (1) Jena Fuseki 4.4.0 with Jena TDB2 triplestore, (2) Amazon Neptune Engine 1.0.5.1, (3) GraphDB Enterprise Edition 9.10.0, (4) RDFox 5.4, (5) QLever (commit version 742213facfcc80af11dade9a971fa6b09770f9ca), and (6) Stardog 7.8.0. In this selection: there are commercial and non-commercial (Jena Fuseki and QLever) applications; there is one triplestore distributed as native cloud-service (Neptune); and there

[58] https://graphdb.ontotext.com/
[59] https://jena.apache.org/documentation/fuseki2/
[60] https://aws.amazon.com/neptune/
[61] https://www.oxfordsemantic.tech/product
[62] https://www.stardog.com/
[63] https://github.com/ad-freiburg/qlever
[64] https://www.wikidata.org/wiki/Wikidata:Main_Page
[65] http://dbis.informatik.uni-freiburg.de/index.php?project=SP2B

is one in-memory triplestore (RDFox). All triplestores support SPARQL 1.1 syntax and provide querying services via SPARQL endpoints.

Table 1: List of the evaluated triplestores.

Triplestore	License	Comment
Apache Jena Fuseki w/TDB2	Open Source	
Ontotext GraphDB EE	Commercial	
OST RDFox	Commercial	In-memory database.
Stardog	Commercial	
Amazone Neptune	Commercial	Cloud-based. Based on Blazegraph.
Qlever	Open Source	Experimental. Did not fully support SPARQL.

3.1.2 Datasets

We aim to evaluate the scalability and query performances of the selected triplestores. To this aim, we generated four different datasets with 125M, 250M, 500M, and 1B triples using the dataset generator provided by SP^{2}Bench. To stress even more the selected triplestores, we used a full version of Wikidata (downloaded on 2021-11-19) with more than 16B triples.

3.1.3 SPARQL Queries

SP^{2}Bench comes with a set of 14 SELECT and 3 ASK queries which were designed to cover several relevant SPARQL constructs and operators as well as to provide diverse execution characteristics in terms of difficulty and result size. For Wikidata, the set of 356 SPARQL query examples defined by Wikidata users was selected. Some of these queries use proprietary service extensions deployed for the Wikidata Query Service. We modified the queries to not use these service extensions and discarded some queries that are not compliant with SPARQL 1.1 specification or use proprietary built-in functions not supported by the evaluated triplestores. As a result, a set of 328 queries is used for the evaluation.

3.1.4 Configuration Details

This evaluation was conducted on the AWS cloud. We used Amazon Elastic Compute Cloud (EC2) instances with Elastic Blob Store (EBS) volumes.

The evaluation was carried out simultaneously with one triplestore running on one instance. For SP^{2}Bench, ***r5.8xlarge*** (256GB RAM) was used to deploy all triplestores. For Wikidata, we ran the evaluation on ***r5.4xlarge***, ***r5.8xlarge***, and ***r5.16xlarge*** (128, 256, and 512GB RAM), except RDFox that was deployed only on ***x1.32xlarge*** (~2TB RAM). Due to some differences in the hardware configuration of r5 and x1, we performed a sensitivity analysis of their performance.

To avoid the impact of network latency, the triplestores (i.e., SPARQL server) and the evaluation scripts (i.e., SPARQL client) were deployed on the same machine. Neptune is provided as database-as-a-service in the cloud. Thus, the SPARQL client needs to run on a separate machine. To estimate the effect of network latency, we set up a test with GraphDB where the SPARQL client was running on a separate machine. This analysis helped us to adjust and make the results for Neptune comparable with the others.

3.1.5 Evaluation results using SP^{2}Bench

Import Time. Figure 5 illustrates the import time of SP^{2}Bench datasets. For all triplestores, the import time increases proportionally to the size of the dataset. Jena Fuseki showed poor

import performance while RDFox is the fastest when importing the datasets. RDFox also exhibited similar loading times. With regard to the other triplestores, they show similar import times and very fast loading times. For instance, they were able to load the synthetic dataset with 1B triples in less than a minute.

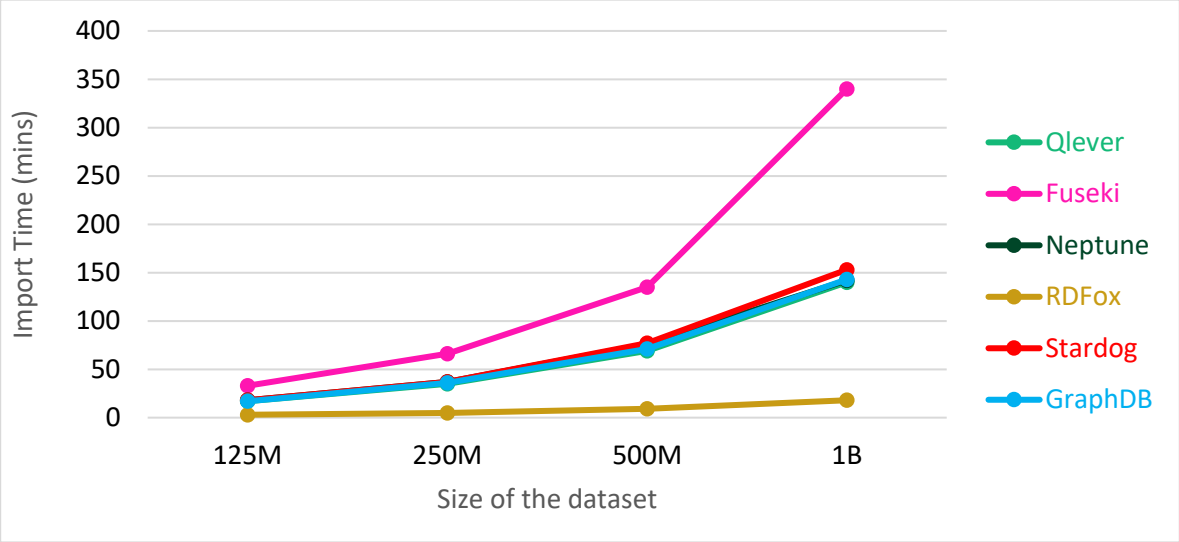


Figure 5: Import time of the triplestores on different SP²Bench datasets.

Query Execution Time. We compute arithmetic mean and geometric mean (the n^{th} root of the product over n number) of the execution times. Accordingly, the failed queries (e.g., timeout, error) were penalized with the double of timeout value. Arithmetic mean is used as an indicator of a high success and failure ratio (i.e., a smaller value indicates a higher ratio of success queries) while geometric mean is used to evaluate the overall performance over success queries (i.e., a smaller value as an indicator of shorter execution time for the success queries).

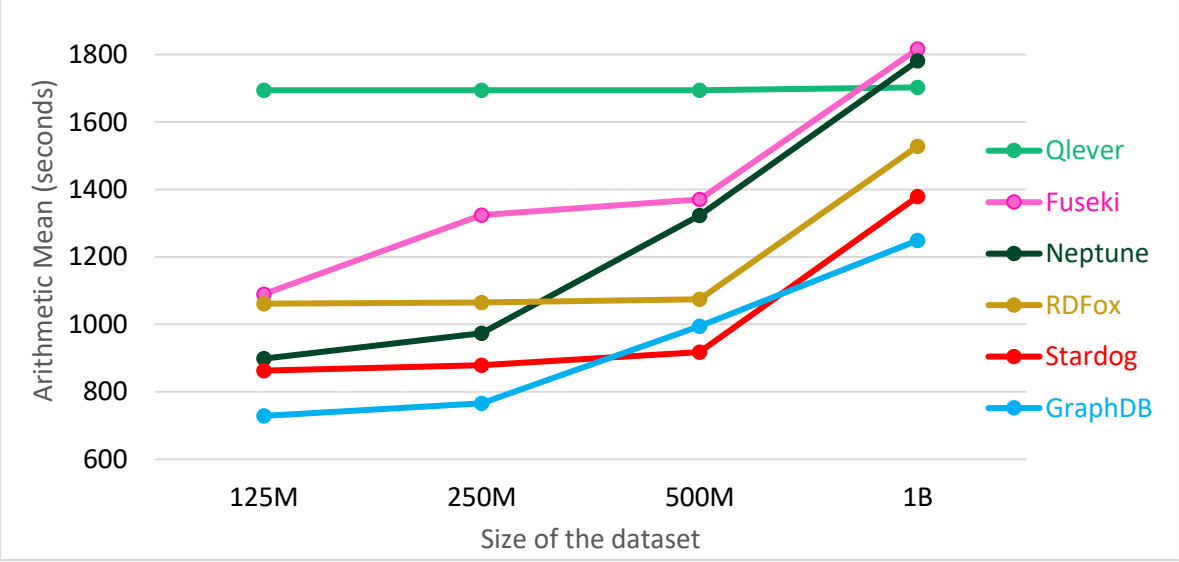


Figure 6: Arithmetic mean of the triplestores execution time on SP²Bench queries.

Figure 6 and Figure 7 present average execution time for the four SP²Bench datasets. Accordingly, the size of the dataset and results has a significant effect on the performance

of all triplestores, in particular as the dataset grows from 125M to 1B triples. Regarding the global performance, the arithmetic means of GraphDB were superior to the others since it had a higher number of success queries. However, RDFox had better performance over successful queries, so its geometric means were the smallest. In all cases, Stardog was always in the top two. It had more success queries than RDFox and executed difficult queries slightly faster than GraphDB. Jena Fuseki delivered the poorest performance while Neptune had mixed results on query execution. QLever was very fast on success queries, but it offered limited support for queries with complex SPARQL constructs.

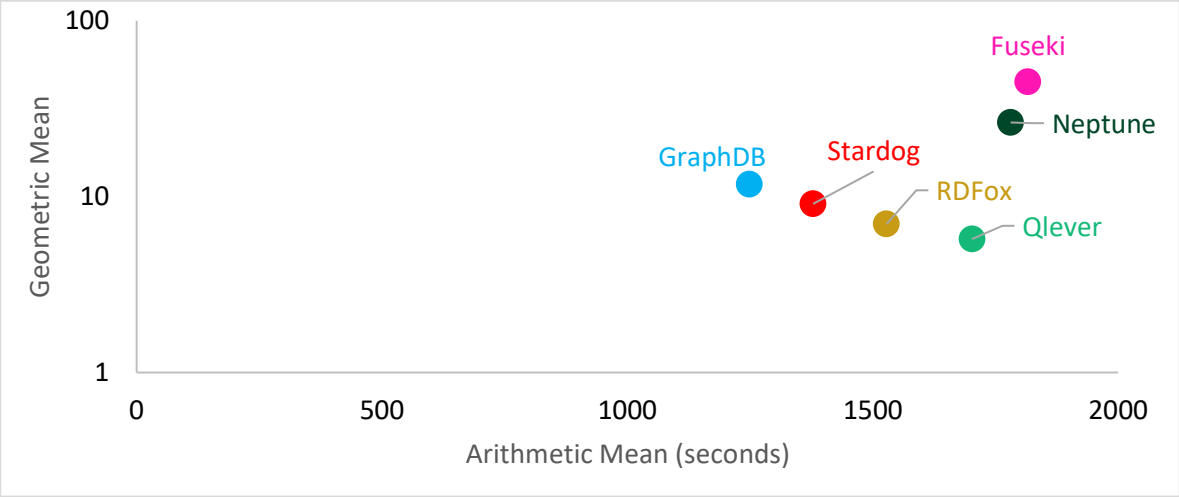


Figure 7: Comparison between Arithmetic Mean and Geometric Mean of the triplestores on SP²Bench queries.

3.1.6 Evaluation results using Wikidata

Import and Export Time. Figure 8 presents the performance for importing and loading Wikidata. RDFox was much faster than the others. This result is consistent with the figures reported in SP²Bench where all triplestores are evaluated using the same machine configuration. However, as the triplestores were restarted, RDFox required around 3.75 hours (40% faster than its initial import time) to reload the data while the others took only a few minutes.

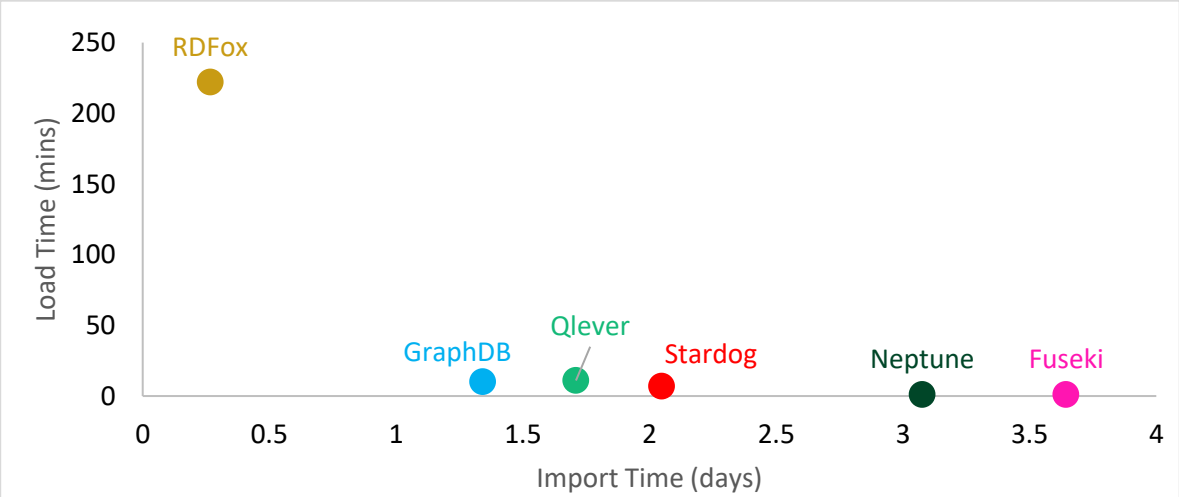


Figure 8: Comparison of Wikidata Import Time and Load Time of the evaluated triplestores.

To measure the export time, we set a timeout of 4 days for the triplestores. Except for QLever which has no support for data exporting, the other triplestores provide native functions to export the data. However, RDFox is the only triplestore that succeeded in exporting Wikidata within the timeout. Stardog did not show any progress or runtime output while Amazon Neptune encountered an error after exporting 503M statements in 1.5 hours. GraphDB took 28 days and 8 hours to export Wikidata. Due to cost constraints, we did not continue the exporting process for the others after 4 days. Based on this figure, it is obvious that exporting is not the prioritized feature of most triplestores.

Query Execution Time. Figure 9 presents the arithmetic mean and geometric mean of triplestores on Wikidata queries. When comparing the execution time of all triplestores, RDFox and GraphDB are the top two triplestores with arithmetic mean of around 14 seconds compared to around 31 seconds for Amazon Neptune. Jena Fuseki and QLever are the slowest triplestores due to a lot of errors and timeouts. Regarding geometric mean, RDFox is still the fastest with a value of 0.04 which is 50% smaller than the value of GraphDB. Stardog is in third place. Its geometric mean is around 0.18 which is 35% faster than Amazon Neptune.

This insight can also be noticed from Figure 10 that compares the best performance (i.e., on r5.16xlarge machines) of the triplestores for the top 50 easy and difficult queries (excluding error queries). Accordingly, for easy queries, it can be clearly identified the order of the triplestores where RDFox is the fastest and Amazon Neptune is the slowest. However, there are mixed results for difficult queries.

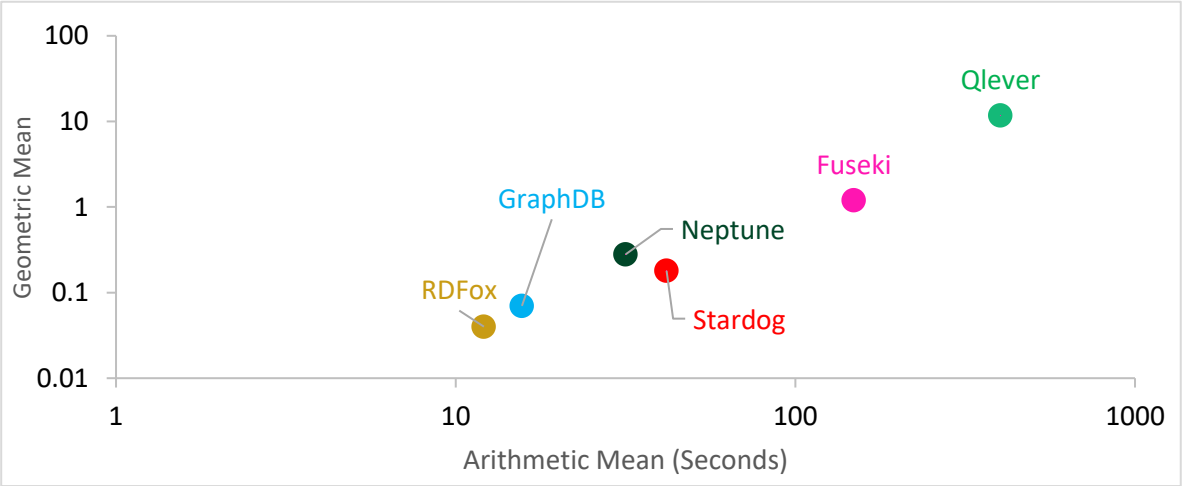


Figure 9: Comparison between Arithmetic Mean and Geometric Mean of the triplestores on Wikidata queries.

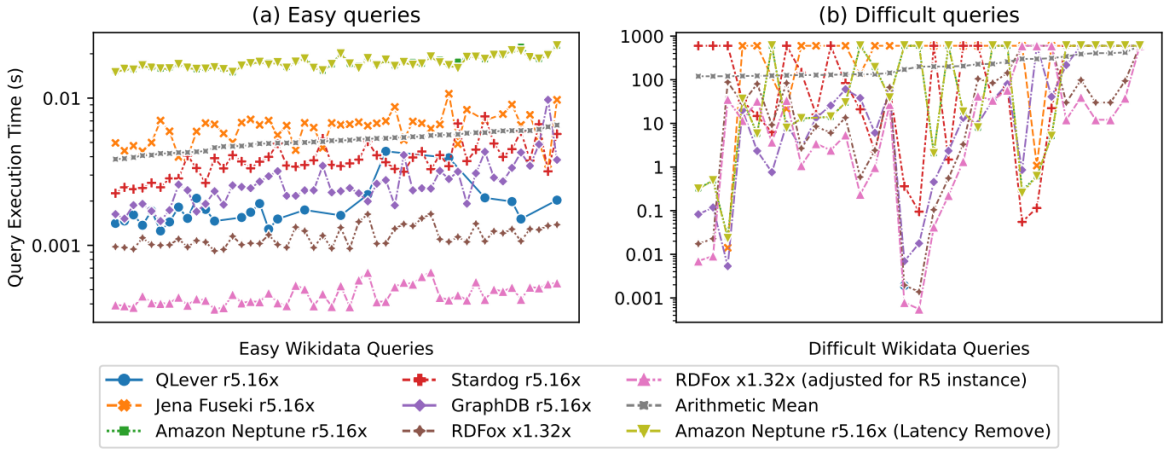


Figure 10: Query execution time of the top 50 easy and difficult Wikidata queries on average (excluding error queries). The queries (x axis) are ordered by the arithmetic mean of the execution time of all triplestores.

3.1.7 Highlights from the triplestore evaluation results

SP2Bench proved to be a great choice to test scalability and common query optimization techniques, which helps us to collect useful insights of the triplestores selected for this evaluation. RDFox was the fastest triplestore importing the synthetic datasets generated for this study and it has better performance over success queries. Regarding the global query performances, GraphDB was superior to the others, followed very closely by Stardog.

SP2Bench also has some limitations. It only provides 17 SPARQL queries offering limited coverage of SPARQL constructs and features. Moreover, some of these queries and the synthetic datasets do not seem to be practical in real use case applications. Our evaluation employing a complete version of Wikidata with 328 queries defined by its users seems to overcome these limitations. This evaluation helps us to stress the triplestores and identify relevant insights. Importing Wikidata, and especially, exporting Wikidata was challenging for all triplestores, where RDFox was significantly more efficient. Loading Wikidata, however, was done much faster by the other triplestores, although a different configuration for RDFox might reduce export and loading time significantly. Exporting Wikidata was an even bigger challenge. RDFox was the only triplestore that managed to export Wikidata, and it completed this operation in a few hours.

In terms of query performances, RDFox reported the best overall performances followed by GraphDB. It is remarkable how consistent GraphDB and Stardog were, in terms of query performances independent of the memory configuration of the machine. This indicates a careful optimization of the design of both triplestores in terms of memory consumption. With the exception of QLever and Jena Fuseki, most triplestores reported none or just one error in the execution of the queries.

3.2 An Evaluation of OWL2 DL Reasoners

In this section, we present a performance evaluation of six prominent OWL 2 DL reasoners: Pellet, FaCT++, JFact, Openllet, HermiT, and Konclude (Table 2).

Table 2: List of evaluated reasoners.

Reasoners	Version	OWLAPI Version	Release Date	Repository
Pellet	2.4.0	4.5.20	18.4.2015	https://github.com/ignazio1977/pellet
Fact++	1.6.5	4.5.20	31.12.2016	https://bitbucket.org/dtsarkov/factplusplus
JFact	5.0.3	5.1.19	23.09.2018	https://github.com/owlcs/jfact
Openllet	2.6.5	5.1.19	27.09.2019	https://github.com/Galigator/openllet
HermiT	1.4.5.519	5.1.19	18.02.2020	https://github.com/owlcs/hermit-reasoner
Konclude	0.7.0-1138	5.1.19	19.06.2021	https://github.com/konclude/Konclude

The evaluation was conducted on Amazon Elastic Compute Cloud (EC2) ***r5.2xlarge*** instances equipped with 8 vCPUs of Intel Xeon Platinum 8000 series processor at up to 3.1 GHz, 64 GB RAM, Ubuntu 18.04 LTS, and 100 GB EBS gp3 volume which offers SSD-performance with predictable, baseline performance of 3,000 IOPS and 125 MB/s throughput.

The evaluation has been carried out on two datasets: ORE 2015[66]and NCBO BioPortal[67] ontologies:

- **ORE 2015** contains 1920 ontologies from the OWL Reasoner Evaluation 2015 Competition dataset, which can be used to evaluate the performance of the reasoners on small and medium-size ontologies. Regarding the reasoning task realization, we only consider ontologies with more than 100 ABox axioms. This process results in a subset of 624 ontologies for the realization task.
- **NCBO BioPortal** is one of the largest libraries of biomedical ontologies and terminologies. Ontologies have been selected by taking the 20 largest ones in this portal. Additionally, the GALEN ontology[68] was also added to this dataset as it is known to be one of the most difficult ontologies for tableau reasoners. This ontology contains many cyclic axioms, which result in very large models and an extremely long time to classify and realize [15].

There are four different tasks were evaluated:

- **Loading reasoner**: regards the time for initializing a new instance of the reasoner and loading the ontology model into it.
- **Consistency**: verifies whether every class in ontology admits at least one individual.
- **Classification**: computes a class hierarchy with all super classes and subclasses of every class defined in an ontology.
- **Realization**: for each individual, it finds all classes, especially the most specific ones, where the individual is an instance of.

[66] https://zenodo.org/record/18578
[67] https://bioportal.bioontology.org/
[68] https://bioportal.bioontology.org/ontologies/GALEN

3.2.1 Evaluation results using ORE 2015 dataset

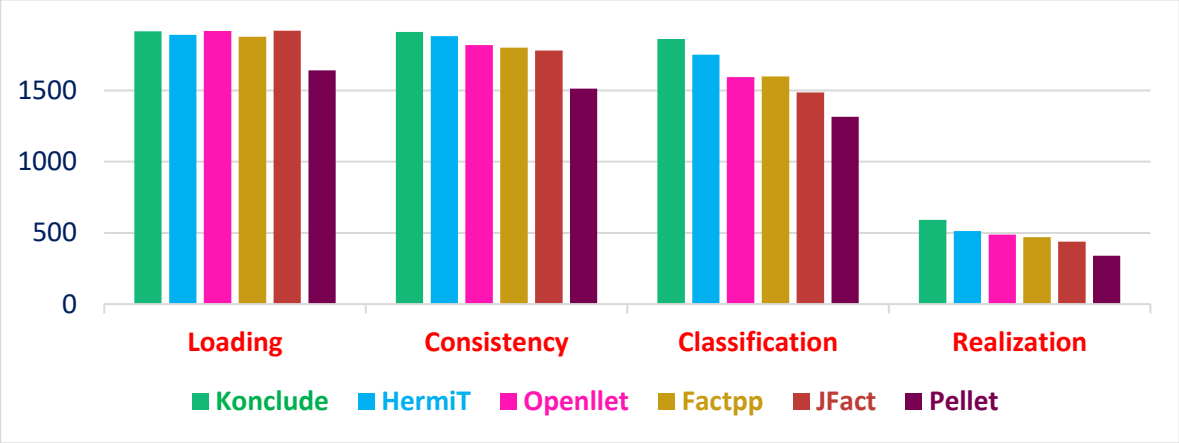


Figure *11*: Total number of successful executions on the ORE 2015 dataset.

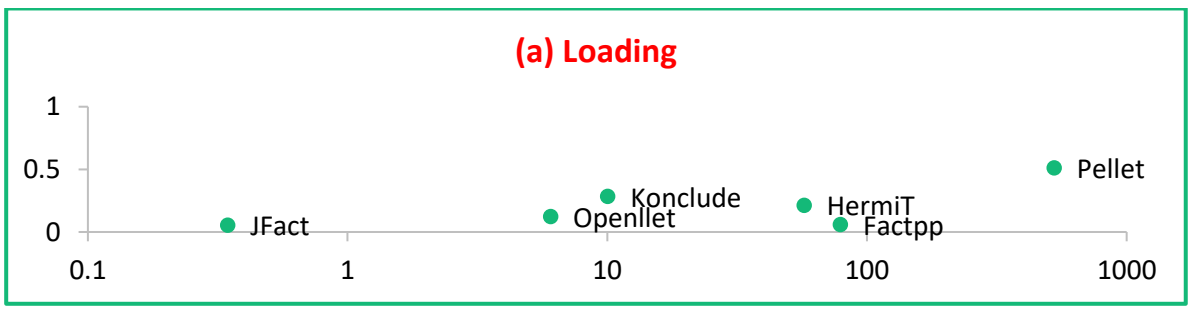
Figure 11 compares the total number of successful tasks with regard to four different type of reasoning tasks.

Accordingly, Konclude and HermiT are always in the top two. Although Openllet was developed based on Pellet, the reasoner had much better results with no errors, while Pellet had the most errors. Furthermore, JFact is the java port of Fact++. However, it has more errors than Fact++ which is the C++ version.

In Figure 12, we compare the arithmetic mean and geometric mean of the execution times of all evaluated reasoners. Regarding loading task, JFact is the fastest while Pellet is the slowest.

Regarding Consistency task, Konclude is the reasoner with most successful tasks, and it also has better performance (considering both arithmetic mean and geometric mean) than the other reasoners, followed by HermiT. Although Openllet has much better arithmetic mean than Fact++ (as because it has fewer errors), Fact++ tends to execute this task faster. As can be seen, Fact++ has much smaller value of geometric mean.

For Classification and Realization, Konclude is also the fastest reasoner overall while Pellet is the slowest. Furthermore, Fact++ tend to execute these two tasks faster than HermiT. As a result, Fact++ has better value of geometric mean than HermiT on these two tasks.



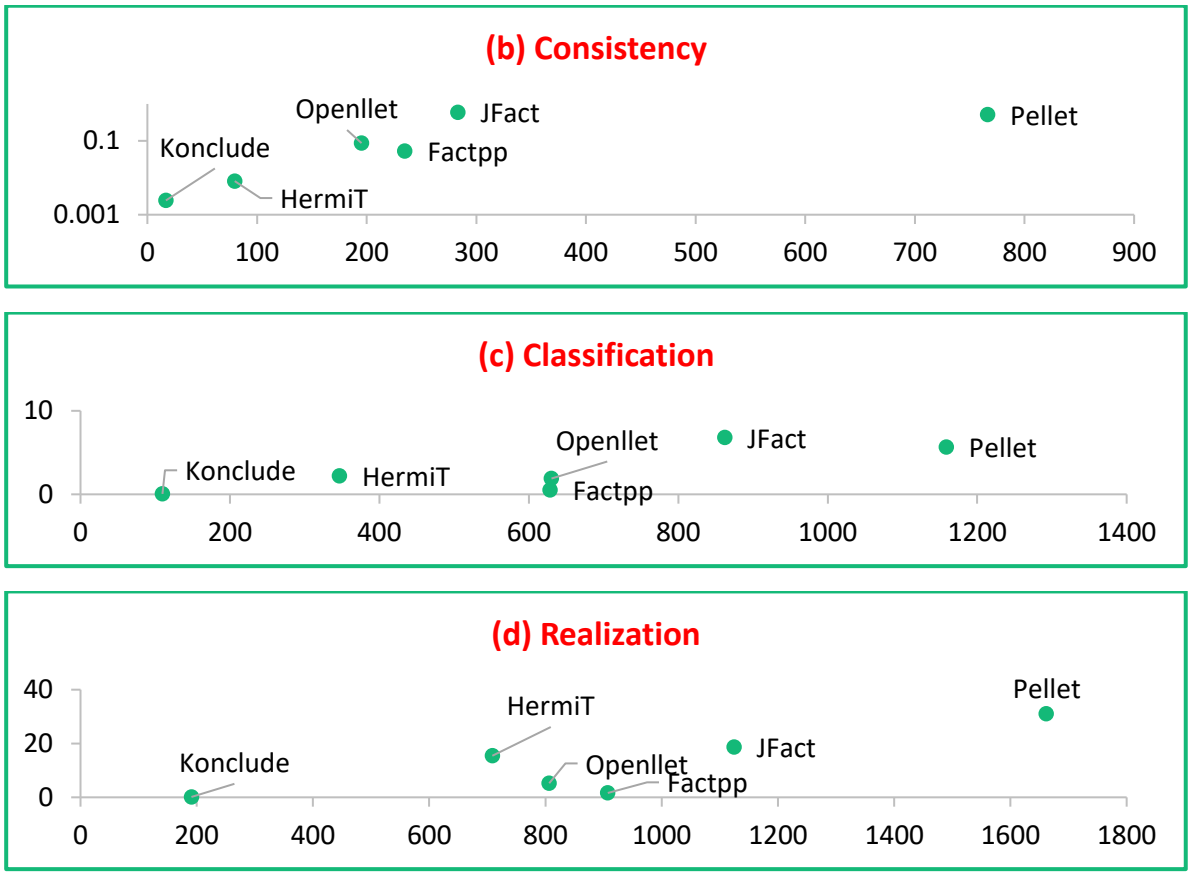


Figure 12: Arithmetic Mean (x-axis) and Geometric Mean (y-axis) of the ORE 2015 execution time.

3.2.2 *Evaluation results using Bio-ontologies dataset*

Figure 13 and Figure 14 presents the performance results of the evaluated reasoners on the 21 NCBO largest ontologies. Overall, the results of the NCBO bio-ontologies evaluation were consistent with the findings of the ORE 2015 evaluation. Konclude emerged as the top-performing reasoner, while JFact failed on nearly all ontologies. Notably, Konclude was the only reasoner to successfully complete all tasks on the GALEN ontology, which is considered one of the most difficult ontologies due to its many cyclic axioms. However, Konclude may encounter memory overflow errors when performing reasoning tasks on some very large ontologies. HermiT was the second-best performing reasoner with fewer errors and decent execution results, followed by Openllet, Pellet, and FaCT++, all of which had failures on almost half of the ontologies.

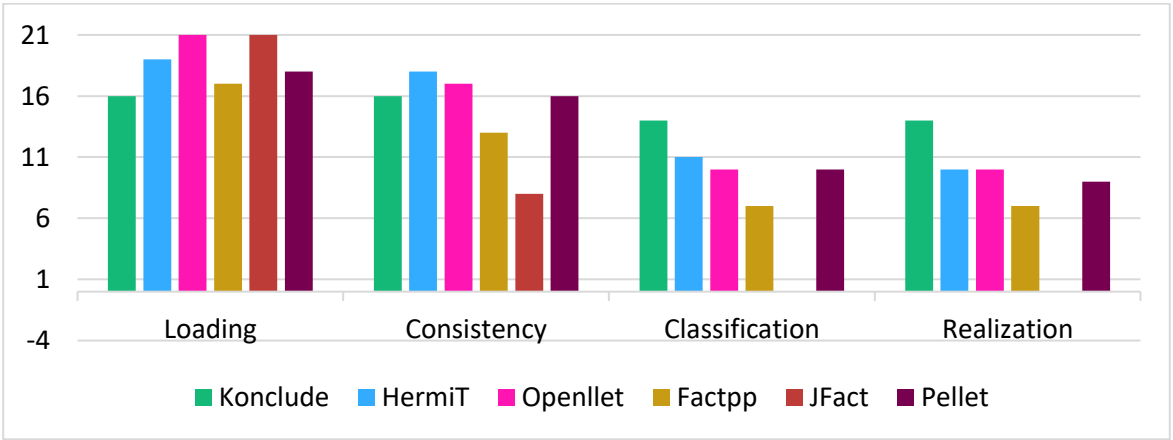


Figure 13: Total number of successful executions on the Bio-ontologies dataset.

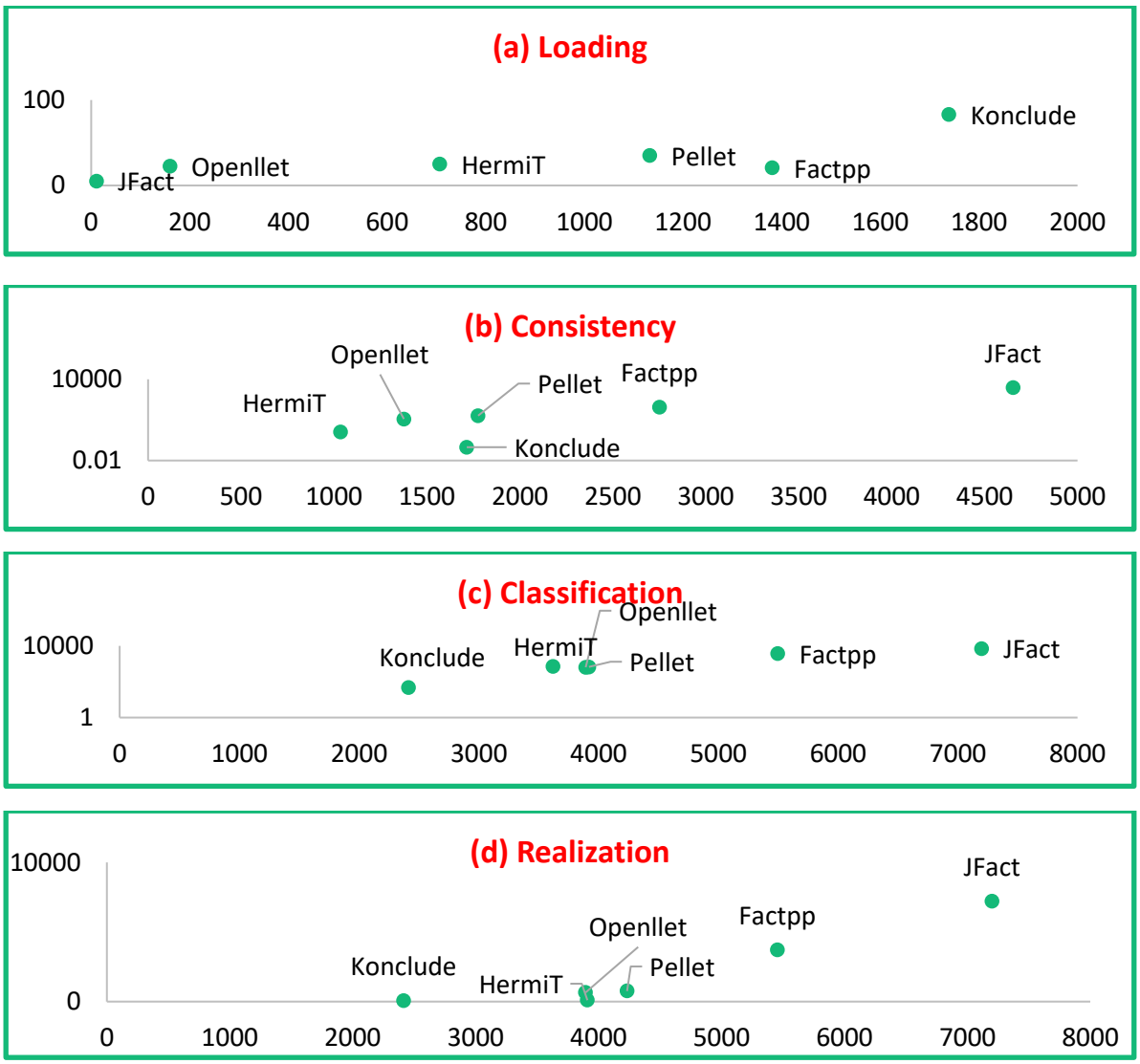


Figure 14: Arithmetic Mean (x-axis) and Geometric Mean (y-axis) of the Bio-ontologies execution time.

3.2.3 Highlights from the reasoner evaluation results

The evaluation results from both datasets were consistent, with Konclude and HermiT frequently ranking at the top for successful reasoning tasks, while JFact had the most failures. In terms of reasoning time, Konclude demonstrated superior performance, while the others showed varying performance across different reasoning tasks and ontology sizes. Our evaluation identifies the strengths and weaknesses of each reasoner, providing valuable insights for ontology developers and application designers when selecting the most suitable reasoning system for their specific needs.

4 Semantic Interoperability in Digital Twin and Data Space

Interoperability regards "the ability of two or more systems or components to exchange information and to use the information that has been exchanged" [16]. There are different types of interoperability [17] from communication protocol level up to organizational level:

- Technical interoperability is the ability to communicate among various devices using different protocols, technologies, software, etc.
- Syntactic interoperability is related to data type, structure, syntax, and format.
- Semantic interoperability regards the meaning of the exchanging content.
- Organizational interoperability is related to communication among various organizations.

While the other types of interoperability can be achieved by defining a harmonized software solution for data exchange, semantic interoperability might require additional universal vocabulary to represent the data in an unambiguous manner. As a result, ontologies are developed as a common approach to semantic interoperability.

Interoperability is important for seamless data sharing following the FAIR principles and enables collaboration among different actors and stakeholders in Circular Data Value Chain. As Digital Twin and Data Space are the main technologies used for data representation and exchange within the Circular AI Platform of this Project, we are considering the use of Circular TwAIn Ontology Library to achieve semantic interoperability in the Digital Twin and Data Space. In the following subsections, we discuss our proposal in this regard. The discussion in these subsections can provide insights regarding data integration and representation for other technical tasks to develop Digital Twins and Data Space towards circularity perspective.

Furthermore, this section also introduces the Smart Data Models initiative developed by the FIWARE Foundation, which includes information models and tools to support interoperability at service level. These tools and models will be considered to be included into the Circular TwAIn Industrial Data Platform discussed in Section 5.

4.1 Data Specification in AAS (Asset Administration Shell)

Assets are the core of industry 4.0. The target is to integrate assets into the IT world. The administration shell [18] is a digital representation of a physical asset, and it may even include a simulation model of the physical asset. Asset administration shells (AAS) are represented as UML diagrams, which can be map to XML, JSON, and RDF formats.

The AAS contains a worldwide unique ID for the asset itself (ID Asset) and for the digital twin (ID Administration Shell). Inside the AAS representation, we can find submodels[69]. Submodels represent distinct digital data regarding the asset. For instance:

- Identification: Contains data about the manufacturer and the product itself.
- Technical data: Contains theoretical data about the asset.
- Operational data: Contains actual data while the asset is in operation mode.

[69] https://industrialdigitaltwin.org/en/content-hub/submodels

- Documentation data: Contains docs about the product.

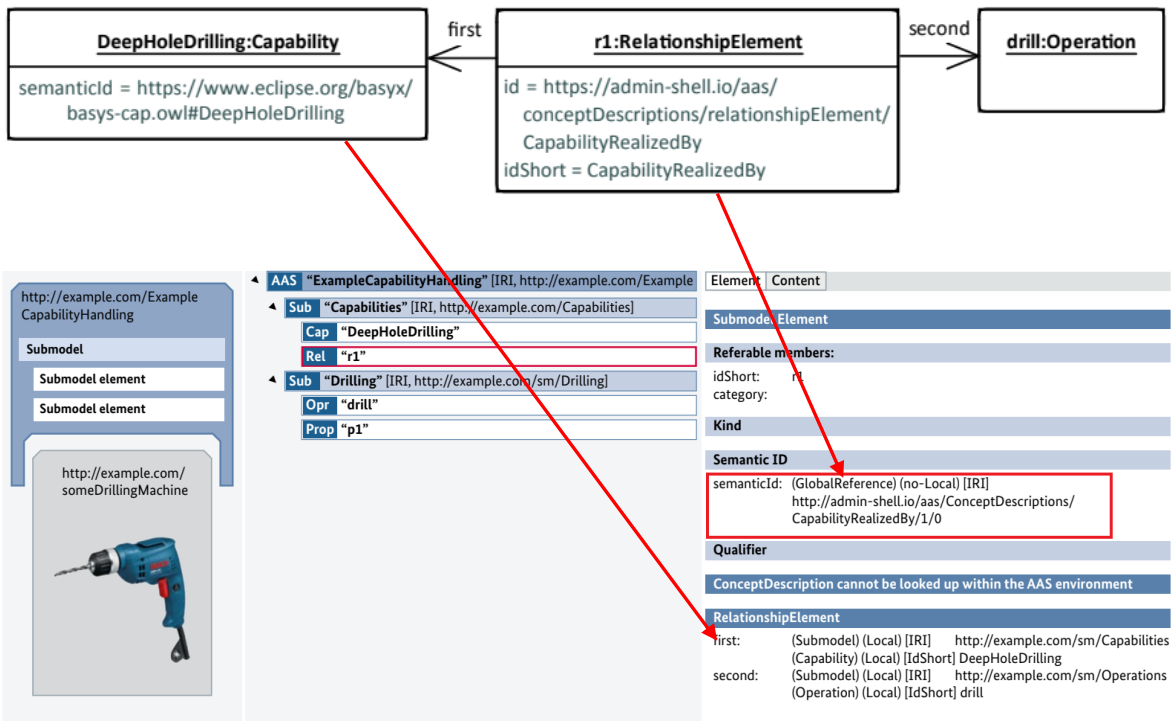


Figure 15: Example of modelling AAS with specification from ontology[70].

A submodel typically consists of a list of properties and such properties must be uniquely identified worldwide with a ***semantic ID.*** Usually, the semantic ID is the global reference to an element in the data specification such as ECLASS[71] or IEC 61360 – Common Data Dictionary[72] to enable the unique identification of the concept described by the property (e.g., a Temperature value, a drilling machine). Alternatively, the semantic ID may refer to the concept (with the URIs) defined in the ontologies. As demonstrated in Figure 15, the classes and properties of the ontologies can be used to describe the asset together with its capabilities. Usually, semantic IDs need to be defined manually during the design and creation of the AAS. As discussed earlier, the next step of Task 4.1 is providing a service to define mappings from AAS data properties to the data specification entities (ontology classes or ECLASS terms) in order to reduce related manual cost.

4.2 Vocabulary Provider for IDS Data Space

The Vocabulary Provider (known as Vocabulary Hub) is one of the technical components in the IDS Reference Architecture Model that "manages and offers vocabularies (i.e., ontologies, reference data models, or metadata elements) that can be used to annotate and describe datasets"[73]. Vocabulary Provider provides the information model which is the basis for the description of the data source. As demonstrated in Figure 16, the Vocabulary Provider provides service to the IDS Connector can usually be queried to find missing semantics for

[70] https://www.plattform-i40.de/IP/Redaktion/EN/Downloads/Publikation/Capabilities_Industrie40_Components.html
[71] https://eclass.eu/en
[72] https://cdd.iec.ch/cdd/iec61360/iec61360.nsf/TreeFrameset?OpenFrameSet
[73] https://internationaldataspaces.org/wp-content/uploads/IDS-Reference-Architecture-Model-3.0-2019.pdf

IDS Resources. Alternatively, as implemented in [19], the Vocabulary Hub can provide service to transform the data into RDF triples.

Furthermore, the Vocabulary Provider stores ontologies that define domain-specific ontologies and provides tools and functions to create, improve, and publish the terms. As a result, RDF triplestores for storing and querying ontologies would be the core component of the Vocabulary Provider. There are several existing implementations of the Vocabulary Provider such as the Semantic Treehouse[74] or the IDSA implementation[75]. The Vocabulary Provider also provides runtime lookup service to support the IDS Connector finding relevant semantics for the data properties. The state-of-the-art solution for this solution is using RML to define the mappings. However, the matching of the query to the semantic reference is not always perfect and might require manual definition of the mappings. As mentioned earlier, in the next step of Task 4.1, we will investigate NLP techniques to address this challenge.

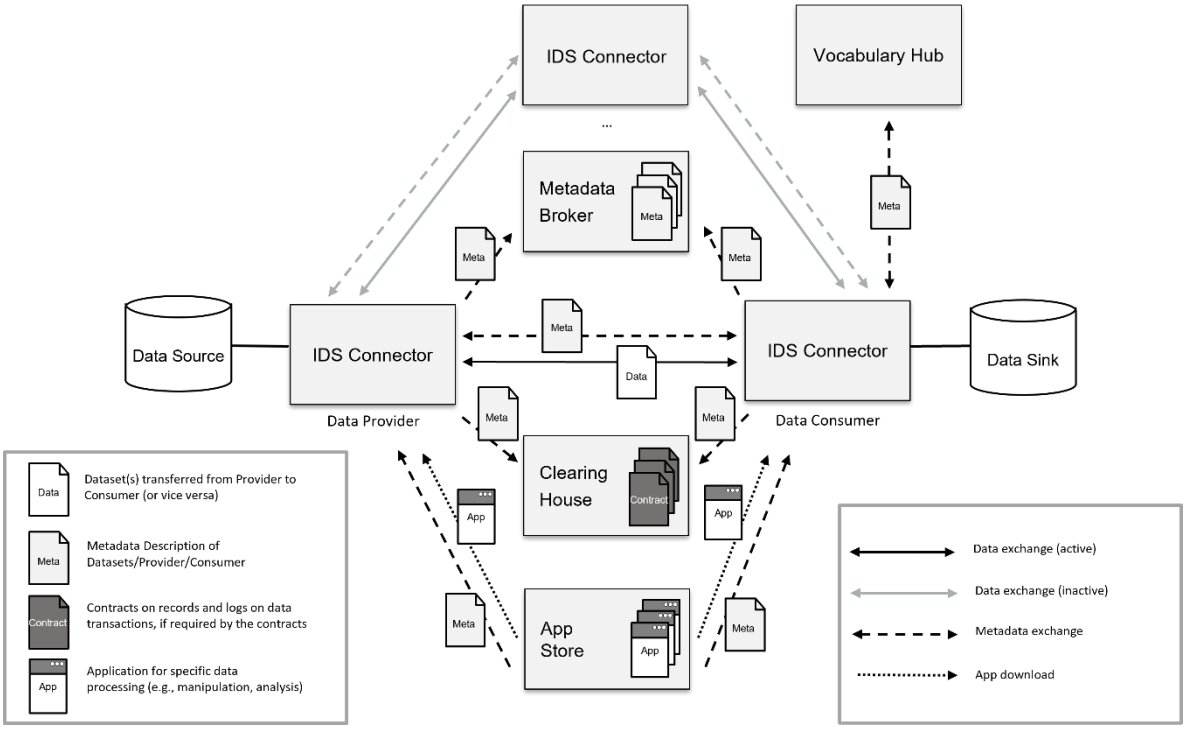


Figure 16: Interaction of technical components in the IDS-RAM 4[76].

4.3 Digital Product Passport (DPP)[77]

DPP aims to gather data on a product and its supply chain and share it across entire value chains so all actors, including consumers, have a better understanding of the materials and products they use and their embodied environmental impact.

The DPPs initiative is part of the proposed Eco-design for Sustainable Products Regulation and one of the key actions under the Circular Economy Action Plan (CEAP). The goal of this initiative is to lay the groundwork for a gradual introduction of a Digital Product Passport in

[74] https://www.semantic-treehouse.nl/
[75] https://github.com/International-Data-Spaces-Association/IDS-VocabularyProvider
[76] https://docs.internationaldataspaces.org/ids-ram-4
[77] https://www.circularise.com/blogs/digital-product-passports-dpp-what-how-and-why

at least three key markets by 2024. These include textiles, construction, industrial and electric vehicle batteries, and at least one other of the key value chains identified in the Circular Economy Action Plan such as consumer electronics, packaging, and food.

The implementation of DPPs in these value chains is designed to support:

- Sustainable product production - enables the transition to Circular Economy and therefore boosts material and energy efficiency, extends product lifetimes, and optimizes product use.
- Businesses to create value through Circular Business Models - thanks to the improved access to data with Digital Product Passports, more businesses can implement service and repair-based business models.
- Consumers to make more informed purchasing decisions - once they are provided with information on the total impact of their buying behaviors.
- Verify compliance with legal obligations - the Digital Product Passports will also act as a record of the standards a product complies with and provides auditors with the data to evaluate this.

What information will need to be included in a Digital Product Passports?

The proposed Eco-design for Sustainable Products Regulation outlines the general and technical requirements for creating, accessing, and sharing Digital Product Passports.

The general requirements for DPPs include complying with the following conditions:

- It shall be connected through a data carrier to a unique product identifier.
- The data carrier shall be physically present on the product, its packaging or on documentation accompanying the product.
- The data carrier and the unique product identifier shall comply with standard (ISO/IEC) 15459:2015.
- All information included in the Product Passport shall be based on open standards, developed with an interoperable format and shall be machine-readable, structured, and searchable.
- The information included in the Product Passport shall refer to the product model, batch, or item.
- The access to information included in the Product Passport shall be regulated in accordance with the essential requirements set out in Article 10 and the specific access rights at the product group level shall be identified in the applicable delegated act adopted pursuant to Article 4.

In order to ensure interoperability for DPP data, harmonized information models for the products and materials should be employed. The Circular TwAIn Ontology Library can be used to represent the DPP data. Regarding the implementation of DPP, apart from existing solutions, AAS also supports the development of DPPs. This solution is provided within the Industry 4.0 approach, namely Digital Product Passport for Industry 4.0 (DPP 4.0)[78].

[78] https://dpp40.eu/

Therefore, the information models defined in the Circular TwAIn Ontology Library can be exploited for the AAS and DPP data specification as described Section 4.1.

4.4 Smart Data Models[79]

A smart data model is a standardized representation of data elements and their relationships, designed to facilitate interoperability and seamless integration across various sectors. It aims to ensure that data availability within specific domains is consistent and harmonized, enabling the development of digital solutions that can be easily replicated and interconnected. These models provide a common framework that serves as a technical foundation for fostering innovation and unlocking the full potential of data. They are developed through collaborative initiatives, such as the SDM (Smart Data Models) initiative led by organizations like the FIWARE Foundation, TM Forum, IUDX, and OASC. The SDM initiative brings together intelligent data models, organizing them by specific domains or industrial sectors. Each domain has its own repository, containing sub-modules focused on relevant topics within that domain. Additionally, there are shared cross-cutting elements that are applicable across multiple domains. This can be seen below in Figure 17.

The SDM initiative brings together intelligent data models, organizing them by specific domains or industrial sectors. Each domain has its own repository, containing sub-modules focused on relevant topics within that domain. Additionally, there are shared cross-cutting elements that are applicable across multiple domains.

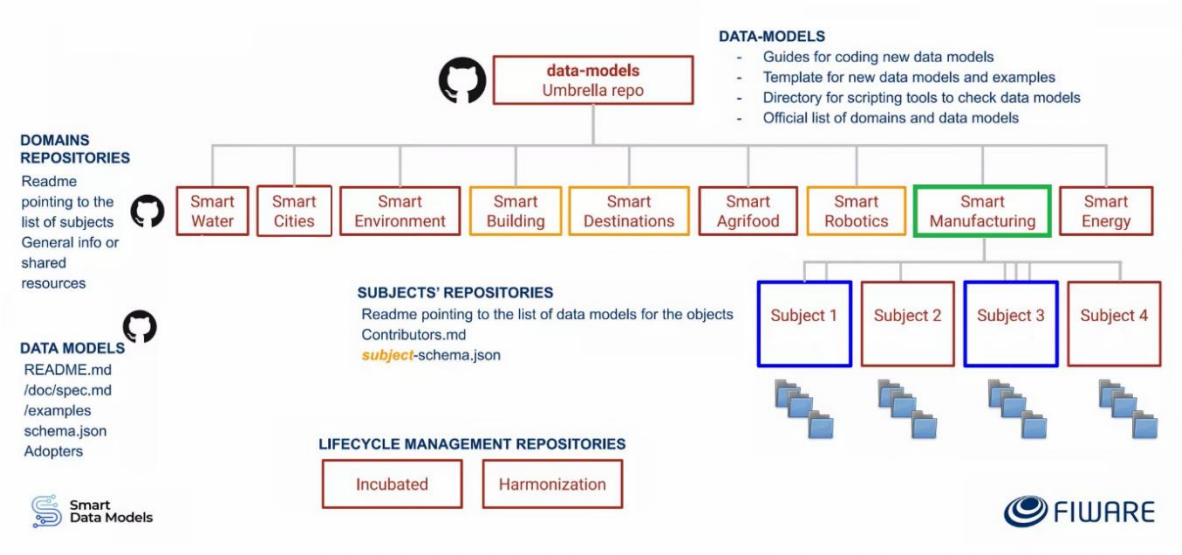


Figure 17: Organisation of data models in the Smart Data Models (SDM) program.

In the smart manufacturing domain, Smart Data Models can facilitate data flow integration from heterogeneous sources, such as different machines, systems, processes and/or human labourers. The data interoperability achieved thanks to the adoption of a common schema, reflects in use cases such as real-time monitoring, optimization, and automation of manufacturing operations. In the context of Circular Economy, the use of standard models

[79] https://www.fiware.org/smart-data-models/

to keep trace of materials, components and product lifecycle could improve the de/re-manufacturing operations.

The smart data model for smart manufacturing is designed to be flexible, extensible, and scalable; it adheres to open standards and supports interoperability among diverse systems, allowing the integration with IoT, cloud computing, and artificial intelligence.

What is in a Smart Data Model?

A Smart Data Model includes three elements:

- **The schema, or technical representation of the model** that defines the technical data types and its relationships, using JSON structures.
- **The specification of a written document** for human readers, it is explanatory of the data model.
- **Examples** for NGSI-LD payloads in several formats.

In Figure 18 there is a schema representing how these three elements are built:

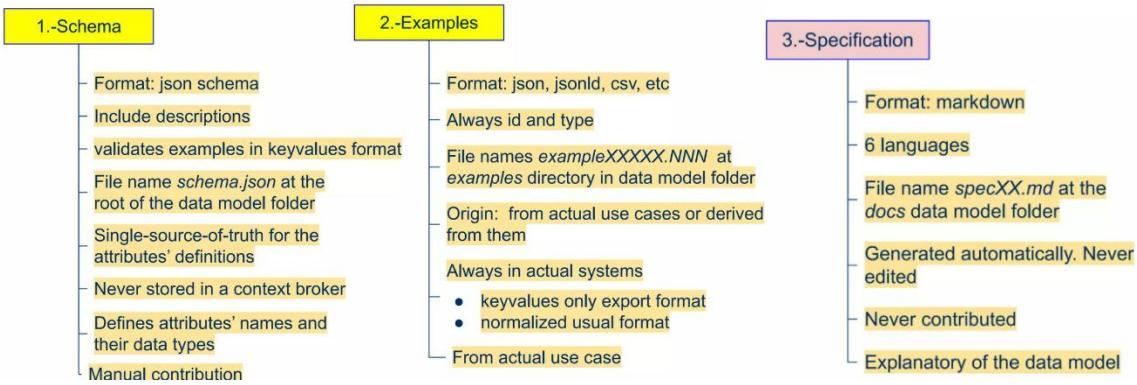


Figure 18: Schema of the Smart Data Models main elements.

An example of an existing data for smart manufacturing:

- **Manufacturing Machine**: Data models related to machinery for manufacturing. Located https://github.com/smart-data-models/dataModel.ManufacturingMachine. It includes these data models:
    - ManufacturingMachine
    - ManufacturingMachineModel
    - ManufacturingMachineOperation

All data models are public and of royalty-free nature. They are free for all forever and their open-source licensing mode grants free use, modification and sharing for users.

5 Industrial Data Platform

To fulfil the technological needs of adherence to digital models and data ontologies, as well as to foster data interoperability, a reference implementation is provided and described within this section. In the first part, an overview on selected components is provided, explaining their general functionality; in the second part, instead, it is put focus on the development of an Industrial Data Platform, explaining how the selected subset of components fulfil its aim.

5.1 The Circular TwAIn Reference Implementation Background

The Circular TwAIn Reference Architecture, as described in **Deliverable D3.1**, sinks its roots in other background projects, such as AI REGIO, CAPRI and COGNITWIN. Each project brought different functionalities that are implemented in the Circular TwAIn Project. For example, AI REGIO and AI4EU projects focused particularly in the Collaborative and Explainable AI field, CAPRI on IoT Data management and analytics, whereas COGNITWIN contributed with its know-how on Cognitive Twins. Furthermore, all projects considered the integration of platforms and toolkits in a Data Space environment.

The reference implementation of Circular TwAIn aims to provide a set of open-source technologies, mainly from the FIWARE and Apache communities, capable of meeting the above-mentioned needs while maintaining a high level of flexibility for integration with other systems.

CAPRI Reference Implementation

The Cognitive Automation Platform (CAP) reference implementation is depicted in Figure 19 and it is organized in functional layers. Instances of the reference implementation tailored on pilots' needs can be consulted through CAP blueprints.

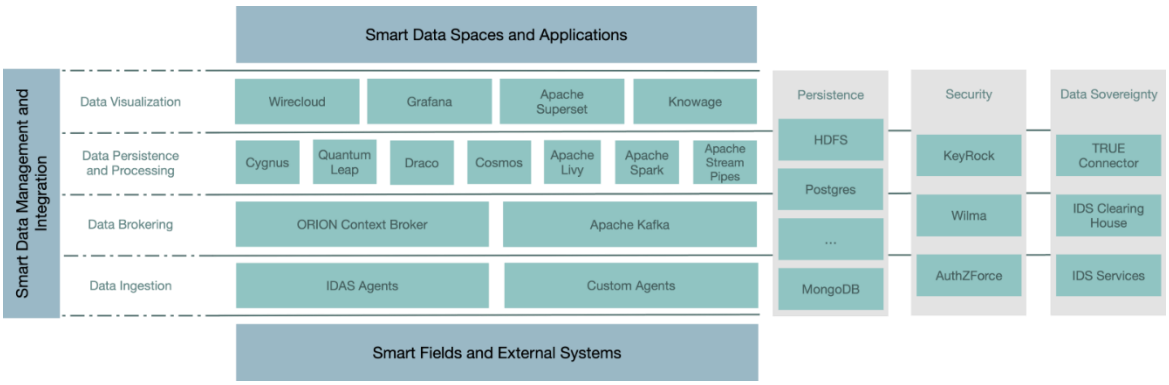


Figure 19: CAP Reference Implementation

Data Ingestion sublayer connects the physical layer to data brokering enables the sharing of device data with the broker in a standardized format. This allows tools to access and analyse the information. The FIWARE IDAS Agent Generic Enabler is an IoT component that converts IoT-specific protocols into the NGSI-LD context information protocol, which is the standard data exchange model in FIWARE. The FIWARE Catalogue includes IDAS Agents such as IoT Agent for OPC UA, IoT Agent for JSON, and IoT Agent for Ultralight.

Data Brokering sublayer in in charge of managing the persistence and processing phase. The key actors in this phase are the Orion-LD Context Broker and Apache Kafka.

The Data Persistence and Processing sublayer consists of various components from FIWARE (Cygnus, Quantum Leap, Draco, Cosmos) and Apache (Livy, Spark, StreamPipes, Hadoop). Its purpose is to store and process the collected data. Cygnus, Quantum Leap, and Draco act as data sinks for the persistence vertical, supporting data storage and pre-processing. Cosmos focuses on big data analysis of streaming and batch processing over context data, while Spark is a parallel processing framework for running large-scale, batch, and real-time data analytics applications across clustered computers. Draco, running Spark jobs through Apache Livy, enables the definition of data flows. StreamPipes is an Industrial IoT toolbox that allows non-technical users to connect, analyse, and explore IoT data streams. Its runtime layer supports the addition of pipeline elements through a built-in SDK in the form of microservices.

Data Visualization sublayer provides a clear understanding of the resulting data by presenting it in a visual context through maps or graphs. Various components cater to different scenarios and comply with diverse data sources, such as Wirecloud enabling the quick creation of web applications and dashboards, Grafana for interactive visualization, Knowage for its comprehensive set of tools for analytics and Apache Superset for its rapidity, lightness and intuitiveness.

AI REGIO Reference Implementation

AI REGIO project focus is mainly put on providing AI as a service using open-source tools, mainly from the Apache community. AI REGIO reference implementation is represented in Figure 20**.** Its reference implementation implements technologies following a three-tier architecture. In particular, edge and platform tiers were considered as SoTA for implementing AI functionalities in Circular TwAIn.

The **Edge Tier** includes a set of tools, namely the Data4AI platform, to ingest, broker, preprocess and persist data. Ingestion is performed with tools such as Apache Sqoop, Apache Nifi and Apache Flume. Brokers such as Kafka helps these tools communicating with each other, while Streampipes helps performing IoT data exploration and preprocessing. Out of the Data4AI platform, distributed computing is provided by Apache Spark and Apache Flink, supported by the Hadoop Distributed File System.

The **Platform Tier** receives data from the edge tier via the access network and handles data transformation and processing. It also facilitates the transmission of control commands from the enterprise tier to the edge tier. The majority of functions related to the information and operations domains are typically located within the platform tier. In this tier, common AI frameworks such as Keras, Tensorflow, PyTorch and Caffè are provided, as well as workflow orchestrator like Argo, Apache Airflow and Prefect. Platform tier is supported by many distributed databases, such as Hadoop, Cassandra or Apache HBase.

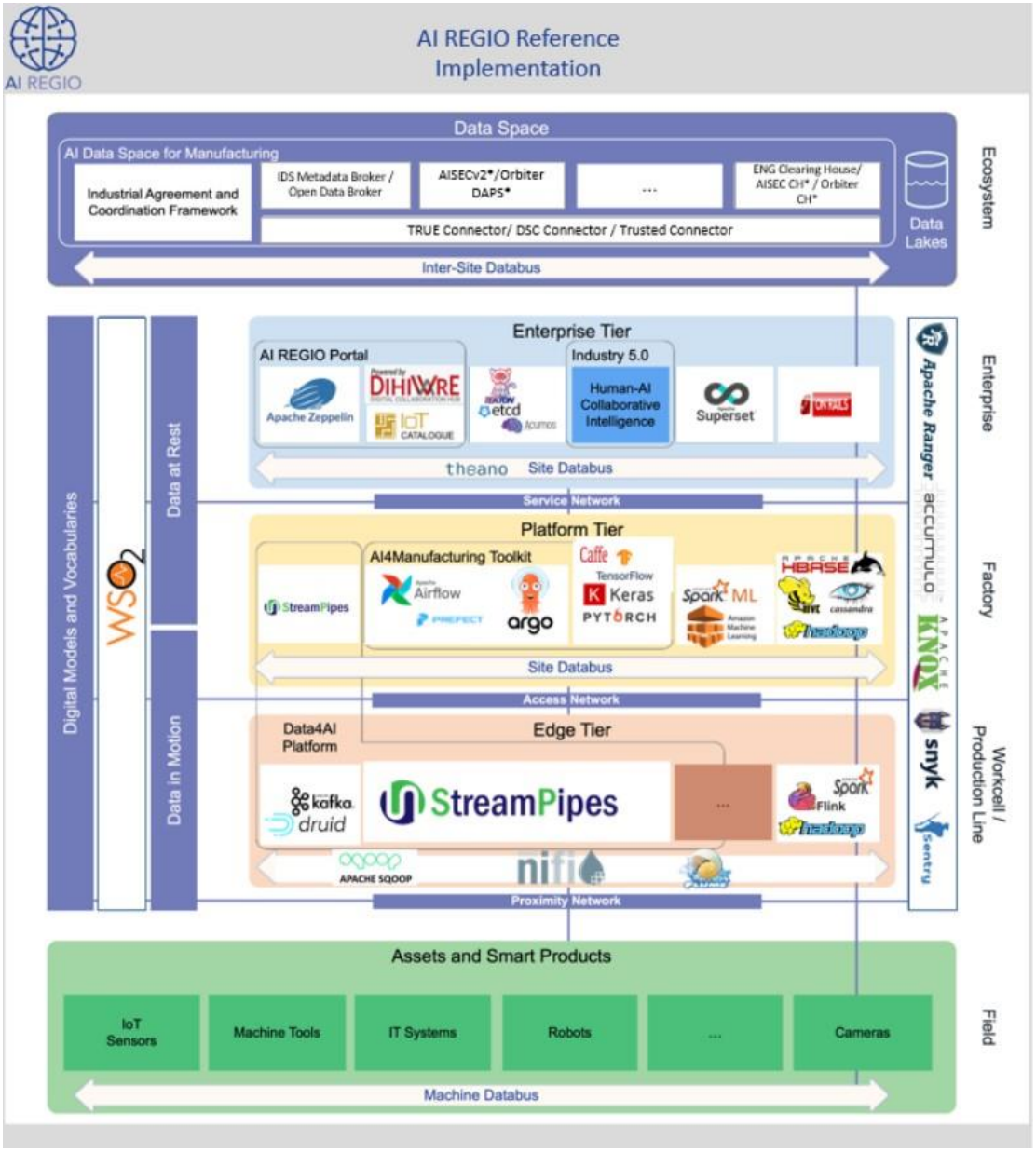


Figure 20: AI REGIO Reference Implementation

KITT4SME Reference Implementation

KITT4SME is designed offering an open-source platform for using up AI solutions in manufacturing for SMEs. The architecture is FIWARE based, a scalable open-source component framework, and structured in different layers as depicted in Figure 21. This solution aims to handle and manage context information from different data sources, distribute KITT4SME data and stream it into involved external components for persistence as well as for AI-based processing. Obtained the results, they can then be integrated back enabling actuation and the enrichment of the current context. The implementation result will be an affordable and tailor-made AI at scale, where manufacturers on a budget and no IT expertise can still benefit from AI, AI developers can focus on delivering business value and the platform provider pools resources to slash IT costs.

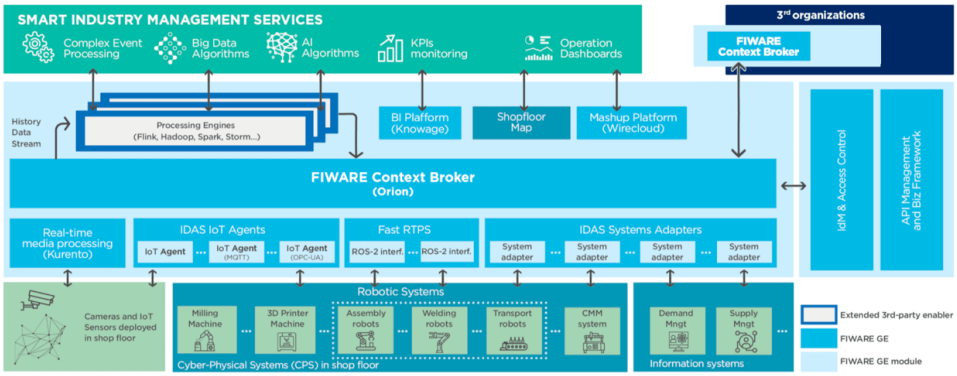


Figure 21: KITT4SME Reference Implementation.

The idea behind the platform is a service mesh, multi-tenant, cloud architecture to assemble AI components from a marketplace into a tailor-made service offering for a factory, connecting them to the shop floor and enable the store and exchange of data in an interoperable, secure, privacy-preserving and scalable way.

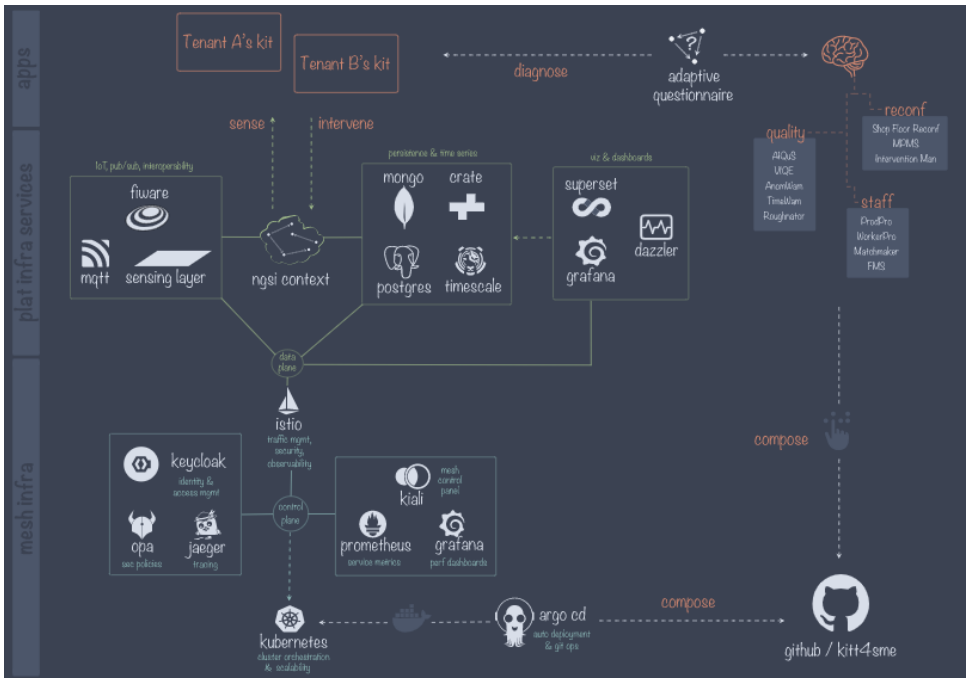


Figure 22: KITT4SME Selected Technologies

One possible deployment of the platform is an Istio[80] service mesh running in a Kubernetes cluster. The cluster hosts the essential building blocks of any respectable FIWARE solution: an agent to provision and manage devices as well as to collect data from and send commands to devices; the Orion Context Broker to provide a publisher/subscriber bus for NGSI services and to manage the current state of the system; NGSI time series, represented by QuantumLeap and supported by CrateDB as a back-end time series DB, to record how IoT data changes over time. A MongoDB service to stash away data in MongoDB, Grafana service to create all sorts of monitoring dashboards rounds off the picks. The solution still

[80] https://istio.io/

rely on Istio to secure communication so that data is protected while in transit—mutual TLS (mTLS) among services in the mesh, certificate management, TLS termination, and so on. For single sign-on (SSO) and identity & access management (IdM), Keycloak is the selected tool. Argo CD, a GitOps continuous delivery framework, sits in to monitors online Git repository to automatically deploy any changes to the cluster config pushed.

COGNITWIN Reference Implementation

The COGNITWIN project instantiated a Toolbox whose organization is based on a specific Digital Twin pipeline, consisting of various components that support Digital Twin (DT) functionality. These components can be interconnected and configured in different ways to create distinct pipeline instances for diverse application contexts, as shown in Figure 23.

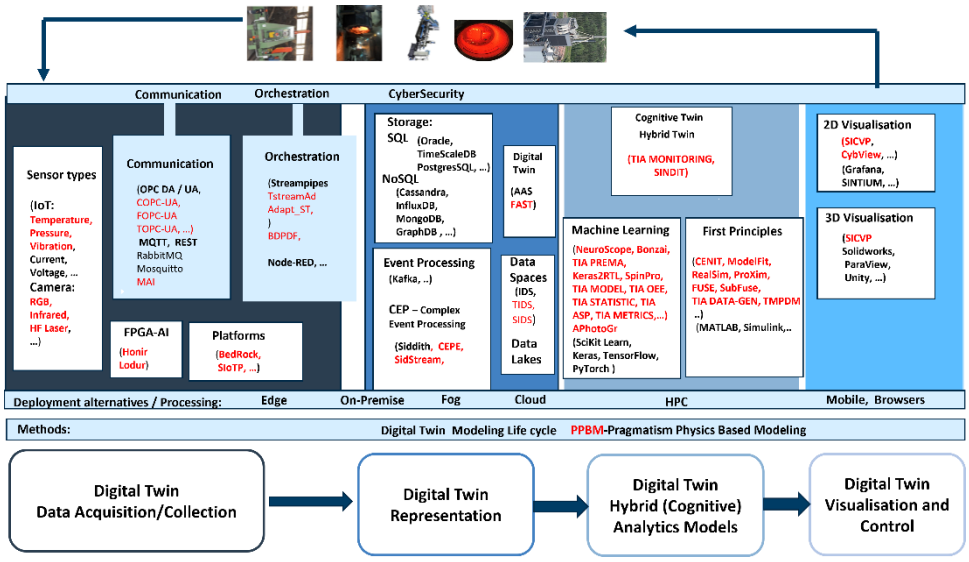


Figure 23: COGNITWIN Toolbox Reference Implementation

The set of tools provided by the reference implementation are provided in different blocks to cover the four main steps of a Digital Twin pipeline:

- ***Digital Twin Data Acquisition Tools and Services*** step is covered by a set of tools to perform data ingestion, harmonization, orchestration and preprocessing. Tools such as MAI are able to acquire weather data, FUSE OPC-UA or Cybernetica OPC UA Server or TIA IoT are used to acquire industrial data using different IoT Protocols, while Lodur acts as a frame grabber for cameras. Other tools, such as the Big Data Pipelines Deployment Framework (BDPDF) Node-Red or Apache StreamPipes, are used as orchestrators. Communication part is held by open-source brokers such as Eclipse Mosquito or RabbitMQ, while connections to specific components are managed by tools like TIA Connect: a connector that links streaming data to FIWARE components or distributed databases such as Cassandra. In the end, platform tools such as Bedrock or SIoTP are flexible instruments able to manage large fluxes of data.
- ***Digital Twin Representation Tools & Services*** step instantiate all tools to manage Digital Twin models and representation. In this step of the pipeline, tools such as FA³ST, PPBM, TIA DATA-GEN, TIA PREMA, SINDIT, SpinPro are used to represent Digital Twin data in different domains. These tools are supported by event processing

tools, such as Apache Kafka, StreamPipes Siddhi-Processor, CEP Editor, SQL (TimescaleDB, PostgreSQL, ...) and No-SQL (MongoDB, InfluxDB, Cassandra, …) storages and Data Space connectors such as IDS connectors or Trusted Factory Connector.

- ***Digital Twin Hybrid (Cognitive) Analytics Models*** step implements cognition in Digital Twin by leveraging open-sources AI frameworks such as scikit-learn, Keras, Tensorflow, Pytorch, as well as proprietary frameworks and tools to apply machine learning (Bonzai, Keras2RTL, Neuroscope, …) or First-Order Model Theory (FUSE, subFUSE, FouMon, …). In the end, Cognitive Twin designers based on StreamPipes are provided, such as CogConf_ST or HyDes_ST.
- ***Digital Twin Visualization and Control*** is the last step of the pipeline, interacting with users that can visualize results from previous steps and interact with Digital Twins. For example, tools such as Grafana, SINTIUM are able to provide 2D charts and visualization, while complex environment such as Unity are deployed for 3D visualization. More complete tools such as TIA UX (for 2D) and Cybernetica Viewer (for 3D) are used also to send commands, thanks the possibility to communicate with other components from the same software toolbox they belong.

5.2 The Circular TwAIn Reference Implementation

The Circular TwAIn Reference Implementation, as depicted in Figure 24, incorporates a range of open-source technologies based on FIWARE and Apache, but also from other open-source initiatives (i.e., Eclipse, IDSA, etc.). The open-source approach aims to prevent vendor lock-in, reduce costs, and eliminate barriers for developers while promoting interoperability and user adoption. The implementation includes essential tools integration to support the entire Project, ensuring a ready-to-be-deployed solution.

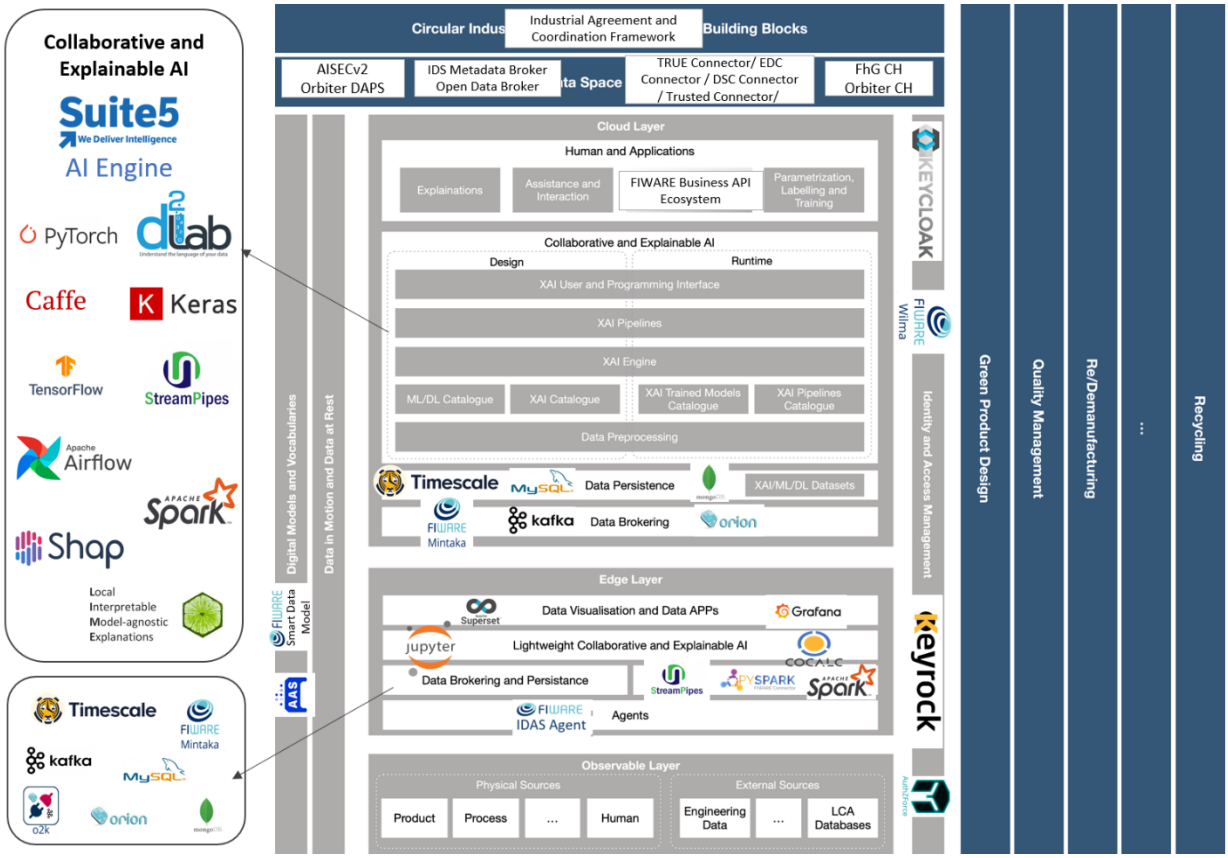


Figure 24: Circular TwAIn Reference Implementation

Digital Models and Vocabularies

Being the Circular TwAIn Reference Architecture a promoter for data exchange and interoperability in a Data Space environment, it is designed to support data specifications already described in Section 5.3.

Identity & Access Management

- **FIWARE KeyRock** is an Identity Management component within the FIWARE ecosystem. When used in conjunction with other security components like PEP Proxy and Authzforce, KeyRock adds OAuth2-based authentication and authorization security to services and applications. This integration enhances the overall security of the system, ensuring secure access control and data protection.
- **FIWARE AuthZForce** is an open-source policy decision point (PDP) and policy enforcement point (PEP) solution for Attribute-Based Access Control (ABAC) developed and maintained by the FIWARE community. It provides a framework for defining and enforcing access control policies based on attributes associated with users, resources, and other contextual information.
- **FIWARE Wilma** is a component of the FIWARE ecosystem that serves as a generic PEP Proxy (Policy Enforcement Point). It acts as an intermediary between the client applications and the backend services, enforcing access control policies based on the rules defined in the PDP (Policy Decision Point).
- **Keycloak** is an open-source software solution that enables single sign-on with identity and access management for modern applications and services. Originally a

WildFly community project, it has been under the stewardship of Red Hat since March 2018, serving as the upstream project for their RH-SSO product. With Keycloak, organizations can streamline authentication and authorization processes, providing users with a seamless and secure experience across multiple applications and services.

Edge Layer

- The **FIWARE IDAS Agents** are components designed to connect objects and enable data gathering and interaction in IoT scenarios. They are essential when integrating IoT devices or gateways with FIWARE-based ecosystems. The IoT Agents act as translators, converting IoT-specific protocols into the NGSI context information protocol, which is the standard data exchange model in FIWARE. Examples of IDAS Agents available in the FIWARE Catalogue include the IoT Agent for OPC UA, IoT Agent for JSON, and IoT Agent for Ultralight.
- **Apache StreamPipes** is a self-service IoT toolbox designed to empower non-technical users in connecting, analysing, and exploring IoT data streams. It provides a user-friendly interface for managing IoT data and offers runtime execution capabilities through an interchangeable layer. The pipelines in StreamPipes can be executed using various wrappers, such as standalone or distributed in Apache Flink. With the built-in SDK, users can easily implement new pipeline elements to meet their specific requirements. These pipeline elements are standalone microservices that can be deployed anywhere, whether centrally on a server, in a large-scale cluster, or at the edge of the network. StreamPipes simplifies the process of working with IoT data streams, making it accessible and adaptable for users with varying levels of technical expertise.
- **Apache Spark** is an open-source framework for distributed computing designed to handle large amounts of data and complex analysis tasks. It was developed to provide fast and scalable data processing in distributed environments, enabling developers to write parallel applications that can leverage the computational resources of a cluster of computers. It provides a set of Python API which took the name of PySpark, allowing developers to leverage its computing capabilities with the python language. This makes it suitable for batch and real-time AI applications, thanks to its possibility to integrate with other AI libraries in one of the most popular programming languages for data science.
- The **FIWARE PySpark Connector** is a software tool within the FIWARE platform that enables bidirectional communication between the FIWARE Context Brokers and PySpark. It consists of two subcomponents: a receiver and a replier. The connector operates at a low-level socket level, establishing a message passing interface between the CB and PySpark.
- The **FIWARE ORION Context Broker** is a powerful implementation of the Publish/Subscribe Broker Generic Enabler (GE). It provides comprehensive lifecycle management for context information, including updates, queries, registrations, and subscriptions. Built upon the NGSI-LD server implementation, it effectively manages context information and ensures its availability. With this GE, users can create context elements and easily handle them through updates and queries. Additionally,

they can subscribe to context information and receive notifications when specific conditions are met, such as changes in the context. The FIWARE ORION Context Broker offers a robust and flexible solution for managing and utilizing context information in various applications.

- **Apache Kafka** is a distributed publish-subscribe messaging system used to integrate applications and data streams. It is known for its speed, scalability, and reliability, making it a critical component in the Hadoop technology stack for real-time data analytics and monetization of IoT data. Kafka is capable of handling massive amounts of data without significant overhead. Unlike traditional messaging systems, Kafka is designed as a distributed system that is easily scalable. It offers three key advantages over other messaging protocols like AMQP and JMS.
- The **FIWARE O2K-Connector** is a Python-based lightweight connector that facilitates the sharing of context data from the Orion Context Broker to Apache Kafka. It functions by subscribing to the Orion Context Broker and subsequently publishing the received context data to a pre-configured Kafka topic within the connector itself.
- **MongoDB** is a distributed, document-based database designed for modern applications and the cloud. It stores data in JSON format, providing support for arrays and nested objects. With its advanced query language, users can easily filter data using any key within the JSON document. MongoDB combines the flexibility of a NoSQL database with the advantages of a relational database, offering features such as ACID transactions and the ability to perform joins in queries. This makes it a versatile choice for a wide range of applications, providing scalability, flexibility, and powerful querying capabilities.
- **FIWARE Mintaka** is as an NGSI-LD temporal retrieval API implementation for the Orion-LD Context Broker. It should be considered as an Orion-LD component and relies on a TimescaleDB installation where all historical data retrieved from the context broker are stored.
- **TimescaleDB** is a powerful and scalable relational database specifically designed to handle time-series data. It extends the functionality of PostgreSQL, adding features optimized for efficient storage, retrieval, and analysis of time-stamped data. It offers advanced capabilities like automatic time-based data partitioning, parallel query execution, and native support for time-series specific operations.
- **MySQL** is a widely used multiplatform relational database management system (RDBMS). It is an open-source component licensed under the GNU General Public License and is designed to conform to the ANSI SQL and ODBC SQL standards. MySQL is a powerful service that can efficiently manage distributed applications in a cloud-native environment. With its robust features and support for standard SQL, MySQL is a reliable choice for storing and retrieving data in various applications and platforms.
- **Project Jupyter** is an initiative aimed at developing open-source software, open standards, and services for interactive computing in various programming languages. It emerged as a separate project from IPython in 2014. Project Jupyter provides support for three primary programming languages: Julia, Python, and R.

Within the Project, several interactive computing tools have been developed and supported, including Jupyter Notebook, JupyterHub, and JupyterLab. These tools enable users to engage in interactive computing and create dynamic documents with code, visualizations, and text.

- **CoCalc** is a web-based virtual workspace that facilitates calculations, research, collaboration, and document authoring. It enables users to transition from their desktop to the cloud using a web browser. It allows users to organize their work in projects or separate workspaces. Each project contains files that can be accessed by the user and their collaborators. Real-time collaboration is a key feature of CoCalc, enabling simultaneous editing of files and instant synchronization of changes among collaborators. CoCalc provides specialized online editors for different file types, promoting an interactive and efficient work environment.
- **Apache Superset** is a contemporary and robust web application for business intelligence. It is designed to be fast, lightweight, and user-friendly, offering a wide range of options for data exploration and visualization. Users with varying skill levels can easily navigate and analyse their data using Superset, from simple line charts to intricate geospatial charts. Autonomic Manager.
- **Grafana**, an open-source analytics and visualization web application, was initially released in 2014. It offers a wide range of charts, graphs, and alerts for the web, provided that it is connected to supported data sources. Grafana supports expandability through a plug-in system, allowing users to customize their experience. With interactive query builders, end users can create sophisticated monitoring dashboards. Grafana is commonly used as a visualization tool in monitoring stacks and is often paired with time series databases.

Cloud Layer

The cloud layer can implement the same data brokering and persistence technologies already described in the edge layer such as:

- Orion Context Broker
- Apache Kafka
- FIWARE O2K
- MongoDB
- MySQL
- TimescaleDB
- FIWARE Mintaka

In contrast to the edge layer, machines on cloud can often leverage better hardware to implement a more robust Collaborative and Explainable AI layer including powerful frameworks and tools such as:

- The **SUITE5 AI Engine** is a powerful tool that combines the capabilities of Python and Spark libraries to construct AI pipelines through a user-friendly graphical interface. It seamlessly integrates with external tools like Apache Streampipes to gather data and efficiently construct AI pipelines with modules dedicated to data transformation, mapping, and processing using state-of-the-art Machine Learning libraries. The tool is highly versatile, allowing users to leverage the advantages of

both plain Python language and the parallel processing capabilities of the Spark engine, enabling efficient and scalable AI pipeline development.

- **Apache Airflow** is an open-source workflow management platform that enables organizations to author and schedule workflows programmatically. Developed by Airbnb, Airflow offers a built-in user interface for monitoring workflows. It is implemented in Python, and workflows are created using Python scripts. Airflow follows the "configuration as code" principle, which means that workflows are defined and managed using code rather than markup languages like XML. This approach allows developers to leverage Python libraries and classes to streamline the workflow creation process. Airflow is a powerful tool for data processing and enables efficient management of complex data pipelines.
- **Apache StreamPipes**, already described in the edge layer, allow the construction of more complex and resource-consuming pipelines.
- **Apache Spark**, already described above, leveraging scalability and powerful AI frameworks.
- **FIWARE Orion PySpark Connector**
- **PyTorch** is an open-source machine learning library that builds upon the Torch library. Torch itself is a powerful collection of machine learning tools, scientific computing capabilities, and a script language based on Lua. PyTorch extends the functionality of Torch and offers a broad range of algorithms specifically designed for deep learning tasks. It leverages the LuaJIT scripting language and an efficient underlying C implementation to provide high-performance computations. PyTorch is widely used in various applications, including computer vision and natural language processing, enabling researchers and developers to effectively tackle complex machine learning problems.
- **Keras** is an open-source software library that serves as a Python interface for building artificial neural networks. It acts as a user-friendly and modular interface for the TensorFlow library, allowing for rapid experimentation with deep neural networks. The primary focus of Keras is to provide an intuitive experience for developing and training neural networks. It emphasizes modularity and extensibility, making it easy to customize and extend the functionality. Keras originated as part of the ONEIROS (Open-ended Neuro-Electronic Intelligent Robot Operating System) research project, which aimed to develop an open and adaptable system for intelligent robotics.
- **TensorFlow** is a free and open-source software library that is widely used for machine learning applications. It provides a comprehensive framework for various machine learning tasks, with a specific emphasis on training and inference of deep neural networks. TensorFlow is built upon a symbolic math library and utilizes dataflow and differentiable programming concepts. It offers a versatile and flexible platform for developing and deploying machine learning models. TensorFlow is utilized extensively both in research and production environments at Google, showcasing its reliability and scalability for real-world applications.

- **Caffe** (Convolutional Architecture for Fast Feature Embedding) is an open-source deep learning framework initially created at the University of California, Berkeley. It is primarily implemented in C++, with a Python interface. Caffe is specifically designed for image classification and segmentation tasks, offering support for various types of deep learning architectures. It leverages the computational kernel libraries Nvidia cuDNN and Intel MKL to accelerate computations, making use of both GPU and CPU capabilities. With its focus on speed and efficiency, Caffe has gained popularity in the deep learning community for its ability to handle large-scale image-related tasks effectively.
- **D2LAB** is a solution-as-a-service that harnesses the power of data processing and analysis to create substantial value. By efficiently handling and analyzing vast amounts of data, it excels in detecting unusual service behavior, ensuring robust quality assurance, enabling predictive maintenance, and optimizing logistic efficiency. D2LAB empowers businesses to leverage their data resources effectively, leading to improved operational performance and enhanced decision-making capabilities.
- **SHAP** (SHapley Additive exPlanations) is a game-theoretic approach that offers insights into the output of machine learning models. By employing classic Shapley values from game theory and related extensions, it establishes a connection between optimal credit allocation and local explanations. SHAP is available as a Python package, providing an API for AI explainability and visualization tools that facilitate a comprehensive understanding of model behavior. It seamlessly integrates with various machine learning libraries, including PyTorch, TensorFlow, and scikit-learn, allowing users to leverage its capabilities across different frameworks.
- **LIME** (Local Interpretable Model-Agnostic Explanations) is a widely used Python library that offers a simplification approach to explainable AI. It simplifies complex models by employing a linear approximation technique. With its ability to provide explainability functionalities for any black box model, LIME empowers users to gain insights into the inner workings of their models. It achieves this by generating simple plots that effectively illustrate the explanations, making the interpretation of black box models more accessible and understandable.

Data Space Technical Building Blocks

- **FIWARE TRUE Connector** is an open-source connector that allows for secure and standardized data exchange within an IDS ecosystem, enabling active participation and facilitating trusted data linkage in a business ecosystem. It adheres to the latest IDS specifications and can be readily customized for various scenarios, thanks to the separation of the Execution Core Container and Data App components.
- The **Eclipse Dataspace Connector (EDC)** offers a framework that enables secure and decentralized data exchange between organizations. It incorporates the International Data Spaces Protocol (DSP) and other protocols associated with GAIA-X. The connector is designed to be flexible and adaptable, allowing for the integration of alternative protocols and seamless integration into different ecosystems.

- The **Dataspace Connector (DSC)** is an IDS connector component that adheres to the IDS Reference Architecture Model. It incorporates the IDS Information Model and utilizes the IDS Messaging Services for handling IDS functionalities and messages. The main component provides REST API for loading, updating, and deleting resources enriched with metadata, both locally and remotely. It ensures compatibility with other IDS connectors and components by enabling IDS-conforming message handling. Additionally, it implements usage control to support selected IDS usage policy patterns.
- The **Trusted Connector** is an IoT edge gateway platform. It complies with DIN Spec 27070 and ISO 62443-3 standards and enables the seamless connection of sensors with cloud services and other connectors through various protocol adapters. The Trusted Connector prioritizes data security with features like secure boot, remote platform attestation, and data usage control, ensuring users retain control over their data.
- The **AISECv2 "Dynamic Attribute Provisioning Service" (DAPS)** is an infrastructure component in the IDS ecosystem that enriches identities of organizations and connectors with additional attributes. It generates Dynamic Attribute Tokens (DATs) instead of X.509 certificates. This allows for attribute revocation without certificate revocation, inclusion of only necessary attributes, and simplified connector deployment. DAPS enhances the flexibility and security of attribute provisioning within the IDS framework. Example DAPS are the AISECv2 or Orbiter ones.
- The **IDS Metadata Broker** is an IDS Connector that handles the registration, publication, maintenance, and retrieval of Self-Descriptions. These Self-Descriptions encapsulate information about all IDS Connectors, including their capabilities, characteristics, offered interfaces, component ownership, and metadata of the data they provide. Operators of Connectors provide Self-Descriptions, which collectively form metadata within the IDS ecosystem.
- The **Fraunhofer Clearing House (CH)** is an implementation of the generic IDS CH that relies on a logging service to capture clearing, billing, and usage control information. It securely stores encrypted data and ensures practical immutability through measures such as preventing modification of log entries, chaining log entries through hash values, and providing signed receipts for data integrity verification.

Data Space Governance Building Blocks

- An **Industrial Agreement and Coordination Framework** forms the basis of business transactions, providing agreements among actors. It includes Data Space Boards for governance, overarching cooperation agreements, a Continuity Model for managing changes, and regulations. All operational and organizational building blocks rely on the existence of governing boards or sector specific bodies that provide common and accepted rules. Trust is ensured through accepted rules and monitoring, enforced by authorities. This separation of powers ensures governance in data spaces and promotes a soft infrastructure.

5.3 The Industrial Data Platform

The Circular TwAIn Industrial Data Platform is a technology infrastructure based on the Circular TwAIn Reference Architecture, enabling ingestion, transformation, mapping, brokering and analysis of data generated within an industrial environment. It serves as an optional local hub for managing and integrating data from heterogeneous sources, such as sensors, machines, devices, and production systems, ensuring data to adhere to defined models. The Industrial Data Platform is designed to be deployed at the ends of the industrial Data Space network to acting as a Man-in-the-Middle between Digital Twins and the Data Space itself.

The Industrial Data Platform architecture covers the following functionalities, using the components already described in the reference implementation of the Circular TwAIn Reference Architecture:

- The **Data Ingestion and Brokering:** the Industrial Data Platform is able to receive data from existing entities, such as physical products, processes, humans and/or their Digital Twin representation, enabling a bi-directional communication with them. The Platform is also able to integrate in a flexible way with other systems such as NOVAAS or FA³ST. In the end, the brokering components are in charge to provide both the state of Digital Twin and the results coming from the processing layer.
- The **Data Processing:** transformation and mapping techniques are applied to data, allowing users to work on data from heterogeneous data sources while adhering to data models. This layer also enables cognition to Digital Twins, enabling batch and real-time processing on their data.
- The **Data Persistence:** whenever necessary, the Industrial Data Platform is designed to persist historical data, especially when time-series data are involved in fast-changing entities.
- The **Data Models and Vocabularies:** finally, the Industrial Data Platform operators are provided with a comprehensive range of standardized data models and metadata vocabularies thanks to the FIWARE Smart Data Models initiative. This crucial provision enables the facilitation of seamless communication within an Industrial Data Space environment.

The Industrial Data Platform fits well in the following contexts:

- **Brownfield Integration:** the Industrial Data Platform can extend existing digital twins with new technologies thanks to its flexibility, without intaking their previous implementation.
- **Integration of several data sources:** the platform allows the integration of heterogeneous data sources through Extraction Transformation and Loading (ETL) operations.
- **Cognition enabling:** the IDP enables digital twins with cognition thanks to batch and real-time data processing.
- **Data Space connection:** thanks to deployment of data space connectors, data providers can connect with other data space actors and share their data.

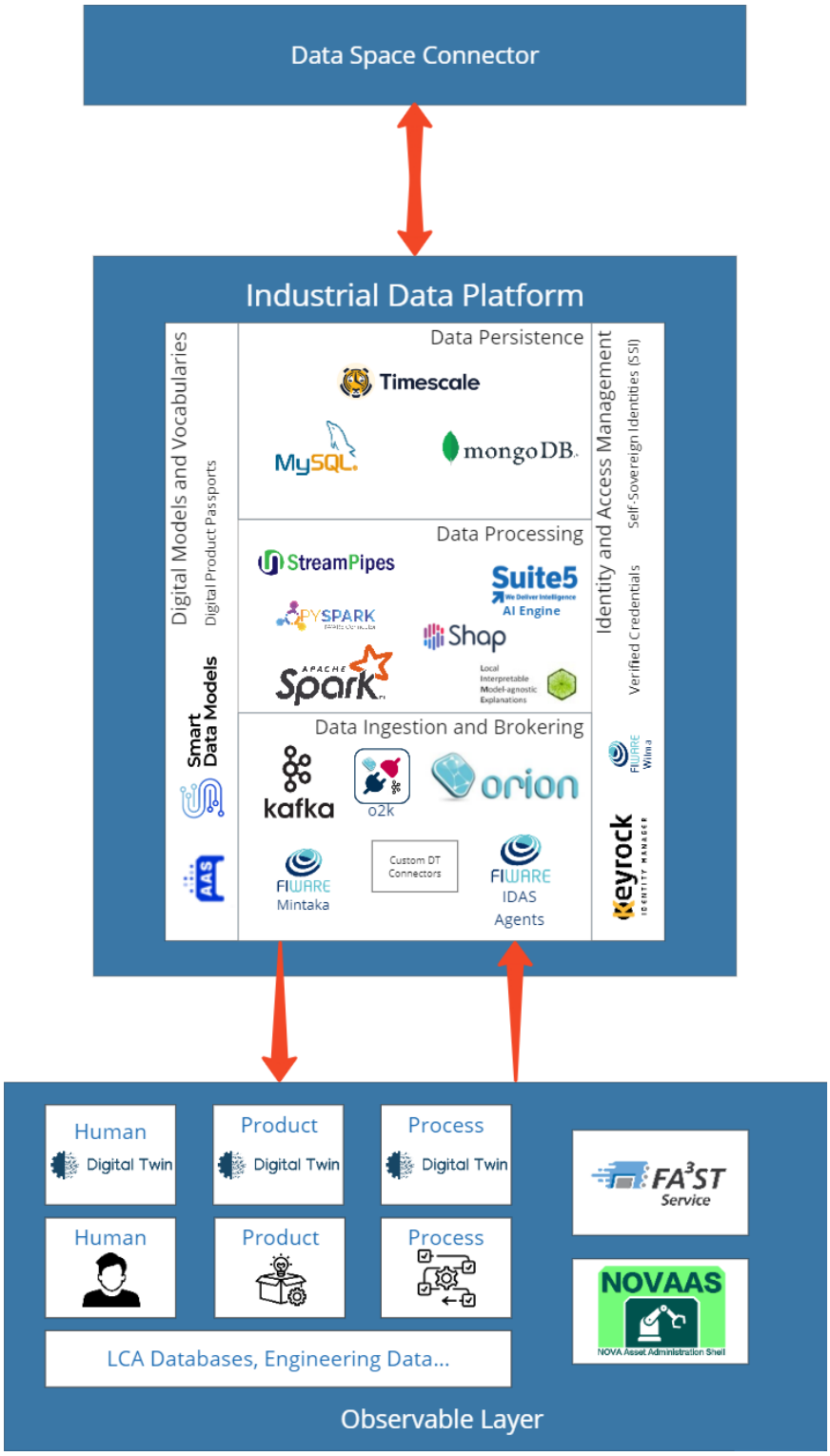


Figure 25: Industrial Data Platform Architecture

In this section it is provided a focused overview on the centralized IDP technological stack, explaining how each component fits with the other to implement the desired functionalities.

Data Ingestion and Brokering

- **FIWARE Orion Context Broker** implementing the NGSI-LD APIs, in the Industrial Data Platform context, allows users to query and update context information. By

implementing the NGSI-LD and JSON-LD standards, it is possible to link entities through relationships, provide property graphs and semantics, thus enabling the use of the Smart Data Model Initiative.

- **FIWARE Mintaka** is an Orion-LD component implementing NGSI-LD temporal API, hence allowing the retrieval of past data.
- Since **Apache Kafka** offers high throughput and low latency, resulting ideal for real-time analytics. It integrates well with other big data frameworks and systems such as Apache StreamPipes, enabling the first step to integrate data flows.
- **FIWARE IDAS Agents** include a set of FIWARE generic enablers acting as an interface to the IoT world. They are able to gather data from heterogeneous sources of data using different IoT protocols but also to send commands back to the observable layer, hence enabling a bidirectional communication with both digital twins and physical machines.
- The **FIWARE O2K-Connector** is used to ensure a seamless communication between Orion and Kafka.
- Other **Custom DT Connectors** can be deployed to connect with digital twins in a flexible way, allowing the integration of other DT systems and services such as NOVAAS or FA³ST.

Data Processing

- **Apache StreamPipes** is the core component for data flow integration: thanks to its wide range of pipeline components, it can merge data from several IoT devices, data brokers, databases and web services, performing data extraction from sources, transformation, cleaning and mapping.
- **Apache Spark** is a powerful engine suitable for processing batch and real time data in a scalable way. Thanks to its Python API, PySpark, it is possible to integrate efficient python algorithms to enable Digital Twin cognition.
- The **FIWARE PySpark Connector** acts as a bridge, allowing PySpark algorithms to receive data hosted on Orion Context Broker in near real-time.
- **XAI Open-Source Libraries** such as **LIME** or **SHAP** can be used to add explainability to AI algorithms. Explainability enhances human understanding of AI predictions and gives further insights on the analysed context.
- The **Suite5 AI Engine** facilitates the creation of complex AI and XAI pipelines connecting data processing algorithms encapsulated in modules. In the Industrial Data Platform environment, it can also enhance collaboration, allowing heterogeneously skilled teams to implement Digital Twins cognition.

Data Persistence

A set of databases are implemented in the data persistence layer both as support tools for existing components and to store data:

- **mongoDB** is used in the IDP to support Orion-LD component.

- **MySQL** is deployed to support the KeyRock component, storing users, applications, and policies.
- **TimescaleDB**: stores historical data for implementing NGSI-LD temporal APIs.
- **RDF Triplestore**: stores the ontologies and vocabulary for implementing the ontology library.

Digital Models and Vocabularies

- The **Circular TwAIn Ontology Library,** presented in 2.2, contains the information models and vocabularies to represent data provided by Circular TwAIn pilots.
- The **Smart Data Models,** presented in Section 4.4, allows a data producer referring to common structures in NGSI-LD standard for their Digital Twin, making available their schema (or a part of it) in Orion.
- The **Digital Product Passports (DPP)** are tools under development and their data models are not defined yet. The Industrial Data Platform, thanks to its flexible design, will be able to host Digital Product Passport data.
- **Asset Administration Shell (AAS)** is a widely used standard in Digital Twin systems. In order to make the platform extensible, it is also compliant with AAS, allowing external services (such as FA³ST or NOVAAS) to use some of the functionalities provided by the Industrial Data Platform, should they need it.

Identity and Access Management Layer

The identity and access management layer implements the security technological stack to control access to data. The following components are deployed in the Industrial Data Platform architecture:

- **FIWARE KeyRock** is the FIWARE component mainly responsible for Identity Management In the industrial data platform context, organization might want services such as the Orion-LD Context Broker to be protected from unauthorized access. By doing so, KeyRock would implement the Policy Decision Point (PDP) functionality with PDP Access Control Level 1 (Authentication Access).
- **FIWARE-PEP-Proxy Wilma** implements a PEP-proxy, that could be used in the platform in conjunction with KeyRock to intercept user requests directed to Orion so that it can redirect them to the Policy Decision Point.
- **Verifiable Credentials** consist of identifying information about the subject, issuing authority details, type of credential, asserted attributes, evidence, and constraints with added tamper-evidence and trustworthiness through digital signatures. Holders can share verifiable presentations to prove possession of credentials. Verifiable credentials and presentations can be transmitted rapidly, offering convenience in establishing trust remotely. Privacy considerations are addressed in the specification to balance the ease of expressing digital credentials. Verifiability refers to the ability to be verified, but it does not guarantee the evaluation of truth claims; evidence can be included to aid verifiers in assessing claim veracity.

- **Self-sovereign identity (SSI)** gives individuals full control of their digital identities, allowing them to decide who can access their data and revoke access at any time. Digital identity encompasses online data tied to individuals or organizations. SSI enables self-management of digital identities, eliminating the need for third-party providers. SSI is often synonymous with decentralized identity.

6 Conclusion and Future Outlook

This section summarises the main results of the deliverable and provides the outlook of the future activities that will be performed in the next iteration of Task 4.1 and Task 4.2.

Task 4.1 focuses on selecting ontologies and defining information models for the Circular TwAIn Ontology Library, with the aim of supporting data representation and harmonization in the Project's pilots. Additionally, the task aimed to facilitate semantic interoperability solutions in Digital Twin, DPP, and Data Spaces.

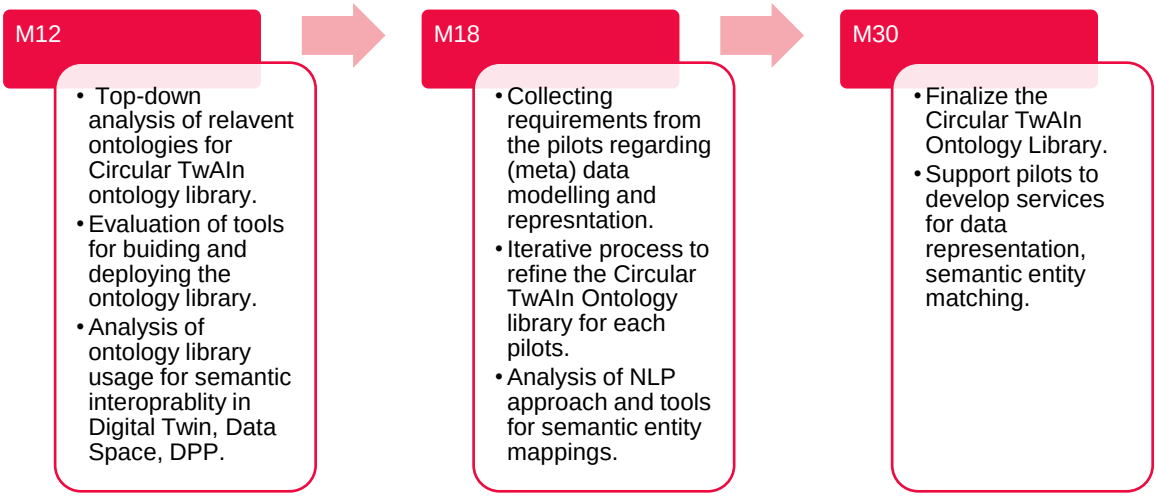


Figure 26: Task 4.1 Proposed Road Map

During the first iteration (M3-M12), relevant ontologies were reviewed based on the specification of the pilots. This process ensured that the Circular TwAIn Ontology Library aligned with the specific needs of the Project. Furthermore, a thorough evaluation of tools for applying ontologies, including reasoning and querying, was conducted. Additionally, libraries for programmatically manipulating ontologies were explored, enhancing the development process.

Furthermore, this task also discussed the use of ontologies to support semantic interoperability in Digital Twin (AAS) and Data Space (Vocabulary Provider). These investigations laid the foundation for integrating the Circular TwAIn Ontology Library into these crucial components of the Project. Moreover, the task addressed information modelling in the DPPs, emphasizing the importance of harmonized information models to ensure interoperability.

Moving forward to the next iteration (M15-M30), Task 4.1 envisions several key areas of focus and opportunities for further advancement. These include:

1. Continuing the development of the Circular TwAIn Ontology Library: The task will continue to expand and refine the ontology library based on the dataset and new requirements regarding data modelling and representation provided by the Project's pilots. Incorporating additional data and refining the ontology models will enable a more comprehensive representation of materials, products, processes, and assets within the Circular Economy context.

2. Exploring Large Pretrained Language Models for Entity Matching and Mappings: Leveraging state-of-the-art techniques, such as GPT-3 and BERT models, presents a new avenue for improving the efficiency and accuracy of entity matching and mappings between ontology classes and data properties. This exploration will involve investigating how these language models can assist in automating and streamlining ontology development processes.
3. Supporting Tool Development for Semantic Interoperability: Task 4.1 recognizes the significance of seamless data sharing and integration in the AAS and the IDS. As a result, efforts will be directed towards developing a dedicated tool that provides robust support for semantic interoperability.

Regarding **Task 4.2,** the main result is represented by the Industrial Data Platform defined according to the Circular TwAIn Reference Architecture described in D3.1. The Industrial Data Platform aims to support the data producer companies providing services for data ingestion, transformation, mapping, brokering, and processing. Provided data can be discovered and accessed from other companies through the Data Spaces Building Blocks described in D3.2 and D4.2.

Furthermore, the Reference Implementation has been depicted, in order to demonstrate how the entire Project (and the related Reference Architecture) can be implemented via mainly open-source solutions coming from FIWARE, Apache, Eclipse and IDSA communities.

References

[1] Studer, R., Benjamins, V. R., & Fensel, D. (1998). Knowledge engineering: Principles and methods. Data & knowledge engineering, 25(1-2), 161-197.

[2] Guarino, N., Oberle, D., & Staab, S. (2009). What is an ontology?. Handbook on ontologies, 1-17.

[3] Arp, R., Smith, B., & Spear, A. D. (2015). Building ontologies with basic formal ontology. Mit Press.

[4] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

[5] Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2019). Language models are unsupervised multitask learners. OpenAI blog, 1(8), 9.

[6] Martín-Recuerda, F., Walther, D., Eisinger, S., Moore, G., Andersen, P., Opdahl, P. O., & Hella, L. (2020). Revisiting ontologies of units of measure for harmonising quantity values–a use case. In The Semantic Web–ISWC 2020: 19th International Semantic Web Conference, Athens, Greece, November 2–6, 2020, Proceedings, Part II 19 (pp. 551-567). Springer International Publishing.

[7] Steigmiller, A., & Glimm, B. (2021, May). Parallelised abox reasoning and query answering with expressive description logics. In The Semantic Web: 18th International Conference, ESWC 2021, Virtual Event, June 6–10, 2021, Proceedings (pp. 23-39). Cham: Springer International Publishing.

[8] Jackson, R., Matentzoglu, N., Overton, J. A., Vita, R., Balhoff, J. P., Buttigieg, P. L., ... & Peters, B. (2021). OBO Foundry in 2021: operationalizing open data principles to evaluate ontologies. Database, 2021.

[9] Ceusters, W., & Smith, B. (2015). Aboutness: Towards foundations for the information artifact ontology.

[10] Baker, T., Bechhofer, S., Isaac, A., Miles, A., Schreiber, G., & Summers, E. (2013). Key choices in the design of Simple Knowledge Organization System (SKOS). Journal of Web Semantics, 20, 35-49.

[11] Keil, J. M., & Schindler, S. (2019). Comparison and evaluation of ontologies for units of measurement. Semantic Web, 10(1), 33-51.

[12] Janowicz, K., Haller, A., Cox, S. J., Le Phuoc, D., & Lefrançois, M. (2019). SOSA: A lightweight ontology for sensors, observations, samples, and actuators. Journal of Web Semantics, 56, 1-10.

[13] Lam, A. N., Elvesæter, B., & Martin-Recuerda, F. (2023, May). Evaluation of a Representative Selection of SPARQL Query Engines Using Wikidata. In The Semantic Web: 20th International Conference, ESWC 2023, Hersonissos, Crete, Greece, May 28–June 1, 2023, Proceedings (pp. 679-696). Cham: Springer Nature Switzerland.

[14] Lam, A. N., Elvesæter, B., & Martin-Recuerda, F. (2023). A Performance Evaluation of OWL 2 DL Reasoners using ORE 2015 and Very Large Bio Ontologies. In

Proceedings of the 1st International Workshop on Data Management for Knowledge Graphs (DMKG2023), May 28, 2023, Hersonissos, Greece.

[15] Rector, A., & Rogers, J. (2006). Ontological and practical issues in using a description logic to represent medical concept systems: Experience from GALEN. Reasoning Web: Second International Summer School 2006, Lisbon, Portugal, September 4-8, 2006, Tutorial Lectures 2, 197-231.

[16] Geraci, A. (1991). IEEE standard computer dictionary: Compilation of IEEE standard computer glossaries. IEEE Press.

[17] Rahman, H., & Hussain, M. I. (2020). A comprehensive survey on semantic interoperability for Internet of Things: State-of-the-art and research challenges. Transactions on Emerging Telecommunications Technologies, 31(12), e3902.

[18] Bader, S., Barnstedt, E., Bedenbender, H., Berres, B., Billmann, M., & Ristin, M. (2022). Details of the Asset Administration Shell-part 1: the exchange of information between partners in the value chain of Industrie 4.0 (Version 3.0 RC02).

[19] TNO. (2022, April). The Vocabulary Hub to configure data space connectors (Report No. TNO 04). Trusts Project. Retrieved from https://www.trusts-data.eu/wp-content/uploads/2022/06/04_TNO-april-2022-The-Vocabulary-Hub-to-configure-data-space-connectors.pdf

Co-funded by
the European Union

This Project has received funding from the European Union's Horizon Europe research and innovation programme under grant agreement No 101058585