Document title: D3.2 Major industrial data model semantics analysis

Version	Status	Date
1.3	final	2025.06.12
Editor	**Contact**	
Péter Olaszi	peter.olaszi@aitia.ai	

Deliverable D3.2
Major industrial data model semantics analysis

Work package and task leader: Nils Sandsmark
nils.sandsmark@posccaesar.org

Editor: Péter Olaszi
peter.olaszi@aitia.ai

Abstract

This document constitutes Deliverable 3.2 of the Arrowhead fPVN project. Work Package 3 is responsible for defining and selecting data model languages, providing guidelines and tools for the Use Cases. This deliverable analyzes the semantic web as an extension to existing major data models

EUROPEAN PARTNERSHIP

Chips.JU project 101111977 - Arrowhead FPVN
Project Coordinator: Professor Jerker Delsing | Luleå University of Technology

Grant agreement no.	**101111977**
Project acronym	**Arrowhead fPVN**
Project full title	**Arrowhead flexible Production Value Network**
Dissemination level	PU
Due Date	31-05-2025
Date of Delivery	12-06-2025
Deliverable Number	D3.2
Deliverable Name	Major industrial data model semantics analysis
AL / Task related	Tasks 3.1, 3.2
Author/s	Péter Olaszi (AITIA)
Contributors	Arrowhead fPVN WP3 partners
Reviewer	Jerker Delsing
Keywords	Industrial data models, semantics, interoperability, ontology
Abstract	This document constitutes Deliverable 3.2 of the Arrowhead fPVN project. Work Package 3 is responsible for defining and selecting data model languages, providing guidelines and tools for the Use Cases. This deliverable analyzes the semantic web as an extension to existing major data models.

Table of contents

1 Introduction 5
2 Synergy 6
2.1 In-person meetings with DISC and CFIHOS Semantics collaboration 6
2.2 Regular IDO Guidelines development meetings 6
2.3 Technical contributions and collaborative inputs 6
3 Ontology alignment and ontology mapping 7
3.1 Ontology scope and domain coverage 8
3.2 Evolution of ontologies 8
4 Basis for developing new OBI standards 9
5 Interoperability between major standards used in industry 10
5.1 Semantic interoperability 10
5.2 Machine reasoning in safety-critical applications 10
5.3 Cross-industry data integration 11
5.4 Integration with Digital Languages 11
5.5 Autonomous Adaptation 11
6 Mapping existing data models to IDO 12
6.1 ISO 10303-239/242 to IDO 12
6.2 CFIHOS Semantics to IDO 13
6.3 DEXPI to IDO 14
6.3.1 Process Modeling with IDO 15
6.4 ISO 15926-4 to IDO 17
7 Creating new ontologies with IDO as upper ontology 19
8 Arrowhead fPVN Use Case 2.9: Digital Twin 20
9 Arrowhead fPVN Use Case 3.9: Process industry 23
9.1 Balas 24
9.2 PLM Reference Data Library 25
9.3 DEXPI Process 27
9.4 Foundations for mappings 29
9.5 Balas to PLM RDL 29
9.6 Balas to DEXPI Process 34
10 Modeling and knowledge representation languages 37
10.1 Open-world and closed-world assumptions 37
10.2 SysML 37
11 A move towards open-sourcing ontology work 40

12 Discussion 41
12.1 List of patterns for IDO aligned ontologies 42
12.2 Levels of expressiveness 44
12.3 Example for a Centrifugal Pump Tag 48
12.4 IDO provides a unifying language 51
12.5 Classes and Properties as main language elements 52
12.5.1 Classes and Properties in IEC 61360-1:2017 52
12.5.2 Classes and Properties in ISO/DIS 23726-3 IDO 53
12.6 Property modeling defined in IDO 56
12.6.1 Two alternatives to model Properties based on IDO 58
12.7 Relation between lis:Object and lis:Quality and their subclasses 58
12.8 Relation between lis:Quality and lis:QualityDatum and their subclasses 60
12.9 Relation between lis:QualityDatum and lis:UnitOfMeasure and their subclasses 62
12.10 Data types which could be derived from IEC 61360-1 63
13 Conclusion 65
14 References 67
15 Revision history 69
15.1 Contributing and reviewing partners 70
15.2 Amendments 71
15.3 Quality assurance 71

1 Introduction

Work Package 3 selects and analyzes relevant standards and their data models, with focus on their common features and possibly synergies among them. Deliverable D3.1 “Major industrial data models”[1] has already accomplished its goals within the work package: selected major standardized data models relevant to the use cases, and identified the foundational properties, similarities and dissimilarities among of the selected data models. The current document explores how to achieve semantic interoperability among select standards. These standards will eventually form the basis for improving data model translation accuracy, in cooperation with WP4.

Task 3.2 focuses on digital language development and use. The contributors refine the common cross-industry interoperability framework of the selected industrial data models, making use of the semantic technology to reach semantic (and partly pragmatic) interoperability. A team of experts are compiling a White Book – a best practices and guidelines document.

In the current document the following standardized industrial data models are considered for evaluation:

- ISO 10303-239/242 STEP: Product data representation and exchange
- CFIHOS: Capital Facilities Information Handover Specification
- DEXPI: Data Exchange in the Process Industry
- ISO 15926-4: Industrial automation systems and integration – Integration of life-cycle data for process plants including oil and gas production facilities. Reference data.
- In the future further standards may be added, for instance S5000F.

ISO 23726-3 Industrial Data Ontology is selected as upper ontology.

This deliverable demonstrates semantic interoperability through collaborative efforts with DISC and CFIHOS Semantics, including regular technical meetings and working group sessions that have produced concrete modeling patterns addressing real-world engineering challenges. The work encompasses ontology alignment and mapping techniques, the development of Ontology-Based Interoperability (OBI) standards, and the establishment of different levels of expressiveness within IDO-based ontologies.

The analysis includes practical implementations through Arrowhead fPVN Use Cases 2.9 (Digital Twin) and 3.9 (Process industry), demonstrating how semantic technologies integrate with the Arrowhead framework to enable knowledge graph-based data integration across engineering, operations, and maintenance systems. We also examine modeling and knowledge representation languages, including the integration of SysML v2 with IDO-based ontologies for systems engineering governance.

Through detailed examination of property modeling patterns, class hierarchies, and data governance policies, this work establishes a foundation for widespread adoption of IDO-based approaches in industrial settings, supporting the vision of autonomous and evolvable interoperability across the entire industrial value chain.

2 Synergy

In this section we summarize briefly the major project events, activities and contributions leading to this deliverable.

2.1 In-person meetings with DISC and CFIHOS Semantics collaboration

Building on the initial cooperation established at the Arlanda meeting in January 2025, two significant in-person meetings were conducted in Oslo, Norway, hosted by Aker Solutions. These meetings brought together representatives from Arrowhead fPVN, DISC, and CFIHOS Semantics to advance the alignment work on IDO as an upper ontology for major industrial data models.

The first meeting (March 19-20, 2025) focused on reporting progress since the Arlanda meeting and reviewing plans for the five established working groups: Aligning ISO 10303-239/242 to IDO, Aligning CFIHOS Semantics to IDO, Aligning DEXPI to IDO, Establishing necessary reference data for Process Plant use-cases, and Guidelines for modelling according to IDO. The meeting included presentations on each working group's progress, status updates on ISO 23726 (Ontology based interoperability), and discussions on the Information Modelling Framework (IMF) and IDO integration.

The second meeting (May 6-7, 2025) continued this collaborative effort with a focus on aligning work between the working groups and agreeing on the path forward. Key outcomes included detailed discussions on reference data establishment, DEXPI-IDO alignment challenges, and the development of practical guidelines for IDO implementation. The meeting featured parallel working sessions that allowed for more focused technical discussions on specific alignment challenges.

2.2 Regular IDO Guidelines development meetings

Starting from 2025 February, regular online meetings have been conducted to develop guidelines for using IDO as an upper ontology. The sessions up to 2025 June involved key technical experts from multiple organizations and focused on practical implementation challenges.

The meetings evolved from initial conceptual discussions to detailed technical work on property modeling patterns, with emphasis on the pump lifecycle story as a concrete use case. Technical achievements included the development of activity profile patterns for property assignments, resolution of identifier handling approaches, and the establishment of domain ontology structures that bridge DEXPI classes to IDO-aligned semantic definitions.

A critical milestone was reached in the 2025 June 3 meeting where the team agreed to concentrate on plant-level modeling as the intersection of all approaches, using simplified examples to demonstrate practical IDO implementation patterns rather than attempting comprehensive theoretical coverage.

2.3 Technical contributions and collaborative inputs

The meetings generated technical contributions from partners, including property modeling patterns from Andreas Neumann (Siemens Energy), DEXPI transformation examples from Heiner Temmen, reference data harmonization work from Magne Valen-Sendstad (PCA), and practical implementation feedback from Onno Paap (Fluor/CFIHOS). These contributions are being systematically integrated into the envisioned IDO guidelines document.

3 Ontology alignment and ontology mapping

Ontology Alignment is the process of identifying correspondences between semantically related entities from different ontologies to achieve interoperability[2]. It involves determining which concepts, properties, or instances in one ontology are equivalent, similar, or related to concepts, properties, or instances in another ontology. Ontology alignment enables semantic integration between different data models. It may use automated, semi-automated, or manual techniques, often employing similarity measures (lexical, structural, semantic). The output is a set of correspondence assertions.

Ontology Mapping is the formal expression of relationships between elements in different ontologies. It refers to the actual set of correspondences or transformation rules that specify how concepts in one ontology relate to concepts in another ontology. It is the representation of the output of an alignment process; can be expressed through formal mapping languages or rules. It may include different types of relationships (equivalence, subsumption, transformation), and enables query translation and data integration across ontologies.

Ontology alignment and mapping are necessary to integrate heterogeneous data sources and knowledge representations. Ontology mapping is based on semantic information as a bridge between ontologies. The field of ontology mapping has developed sophisticated evaluation frameworks, notably the Ontology Alignment Evaluation Initiative (OAEI)[3], which aims to assess strengths and weaknesses of alignment systems, compare performance of techniques, and improve evaluation methodologies. Leading researchers like Jérôme Euzenat from INRIA Rhône-Alpes have been instrumental in advancing this field, with Euzenat's comprehensive survey "Ontology matching - Second edition"[4] reviewing more than 100 state-of-the-art matching systems and frameworks.

Modern ontology alignment systems have evolved to incorporate both logic-based and machine learning approaches. LogMap[5], developed by Ernesto Jiménez-Ruiz and colleagues[6], represents a highly scalable ontology matching system with built-in reasoning and inconsistency repair capabilities that extracts mappings between classes, properties, and instances. The system was awarded the SWSA Ten-Year Award for its significant impact on the field[7]. Recent advances have integrated large language models and embeddings, as demonstrated in OWL2Vec*[8], OWL2Vec4OA[9] and LogMap-ML[10], which augment traditional alignment techniques with semantic embeddings and distant supervision.

Industrial Data Ontology (IDO) exemplifies the practical application of ontology alignment in industrial settings. As described by Pål Rylandsholm[11], IDO includes mappings to widely used ontologies such as Semantic Sensor Networks, the Time Ontology in OWL, and GeoSPARQL, providing detailed modeling examples for industrial use cases. This approach is particularly relevant for the Arrowhead fPVN project, which aims to provide autonomous and evolvable interoperability through machine-interpretable content for industrial stakeholders.

The field continues to evolve with innovative approaches like Target Vocabulary Maps (TVM)[12], which offers a simpler, directed approach to vocabulary mapping, and tools like

STROMA[13] for semantic enrichment of ontology mappings. These developments reflect the ongoing need for efficient, scalable solutions to achieve semantic interoperability across diverse domains and applications.

A comprehensive survey of the evolution of ontologies is available at [14].

3.1 Ontology scope and domain coverage

There are numerous industrial ontologies. These are often competing in overlapping domains. One of the central problems is to align overlapping ontologies and then map data (i.e., individuals) represented in one ontology into another ontological representation.

Such mapping is generally not straightforward, because there is no direct mapping between classes in one ontology to classes in another one. This is anticipated, however. As the domains of two ontologies are not completely identical, one expects only a partial overlap between the two. It is foreseen that some class or concept in one ontology will not have direct corresponding class or concept in another ontology.

Consider an illustrative example taken from natural languages: Some northern languages may have evolved dozens of words to describe the variants and quality of *snow*. Naturally, some languages at a warm climate did not evolve to describe such nuances – these languages may have at best only a few words to describe snow.

At this point we need to make a distinction between the *domain scope* of an ontology, and the *domain coverage*. The scope (or extent) of the ontology measures how many concepts and relations the ontology is capable of representing. The overage of the ontology describes that for an actual data set what percentage of the available concepts and relations are actually used within the data set.

If a data set uses only a fraction of the available classes in the ontology, and the majority of those classes fall into a subset with an exact match to another ontology, then it may well be possible to map the particular data set from one ontology to another – even though the domain of the two ontologies do not align perfectly well.

With the natural language example, a person from a warm climate may exchange ideas without problem with a Nordic person who has a rich arsenal of word for snow, as long as the conversation involves concepts with which both of them are familiar. Only when the conversation drifts to the topics of various kinds of snow, arise problems of understanding each other.

3.2 Evolution of ontologies

Ontologies are naturally limited in their domain. Ontologies describing similar domains will contains overlapping concepts and relations. However, domains evolve: new concepts emerge through the advancement of science and technology; some concepts drift away from their original meaning; others split or merge; old concepts become obsolete. In order for an ontology to be relevant, it must follow the evolution of the world which it aims to describe.

On the one hand, ontologies must be standardized, so that they fulfill their role as a reliable platform for information exchange. This standardization takes a snapshot of the ontology, and consequently, takes a snapshot of the domain the ontology describes. As the domain evolves, so should the ontology in order to maintain its relevance over time.

Ontology versioning enables such changes in an ontology[15]. A newer version may (1) correct smaller errors in the current version, it may (2) introduce new concepts or relation without losing compatibility with the current version, or may (3) break compatibility with the current version in order to enable an evolutionary leap. All such changes help the evolution of the ontology.

Users of an ontology will be happy to receive version updates of kind (1) with typo corrections and fixes to annoying bugs. New features of kind (2) will be welcomed by some and possibly skipped by other users. Major version steps of kind (3) will break compatibility with previous versions. Anyone who are willing to take this evolutionary leap will need to transform all their data sets into the new standard. This is the price to pay in order to let the data reflect the contemporary domain.

Such versioning is used in programming languages (e.g., in Python). The approach is to continually collect feedback from the community; consolidate feature requests, bug reports and enhancement proposals; and release timely version updates. Such a community-driven, consensus-based, but centrally orchestrated version management approach would definitely be beneficial for the evolution of industrial data ontologies.

4 Basis for developing new OBI standards

Ontology-Based Interoperability (OBI) is a method for making different computer systems work together by using shared vocabularies and concepts. Instead of forcing each system to translate data formats directly, OBI creates a common understanding layer that all systems can reference. This approach treats interoperability as a *meaning* problem rather than just a *technical connection* problem.

The foundation of OBI rests on ontologies, which are structured representations of knowledge within specific domains. These ontologies define what concepts mean, how they relate to each other, and what rules govern them. All participating systems map their internal data to these shared definitions. This approach offers several practical advantages. When organizations need to integrate new systems, they only need to map the new system to the shared ontology rather than creating connections to every existing system. The semantic layer also enables automatic validation and reasoning about data quality and consistency. Systems can detect when data doesn't make sense according to the domain rules. OBI works particularly well in complex domains where data has rich meaning beyond simple values. The approach scales better than traditional point-to-point integration because adding new participants becomes easier as the network grows.

A key principle underpinning an OBI approach is that the semantic intent of information is captured and exchanged between humans and machines without loss. As with any language, this is achieved by committing to an agreed set of rules and practices. To achieve this in an OBI ecosystem, alignment is necessary to a top-level ontology (TLO). A TLO is a conceptualization of key concepts in a domain. The TLO ensures that information is represented in machine-readable formats while maintaining human intelligibility. This dual-faced nature of ontologies

– being both human-understandable and machine-processable – is what makes OBI standards particularly powerful for complex information management challenges.

Examples of artefacts in an ontology ecosystem include, but are not limited to, ontologies, vocabularies, modelling patterns, and asset models. To be compliant with an OBI standard they must adhere to the ontology commitments set by the TLO as well as to documentation and other management requirements.

5 Interoperability between major standards used in industry

Industrial interoperability operates at multiple levels. At its foundation, semantic interoperability establishes a common vocabulary that allows different systems to understand the same data consistently. This shared understanding enables machine reasoning in safety-critical applications, where systems can automatically infer risks and respond to hazards by understanding relationships between equipment and processes. With this reasoning capability cross-industry data integration helps information exchange between different industry sectors.

The next level involves integration with digital languages and platforms, where semantic models translate into practical implementations using existing technologies like OPC UA and AutomationML. This ensures that both legacy systems and modern applications can participate in semantic data exchange. At the highest level, autonomous adaptation emerges from combining all these capabilities, so that systems can automatically adjust to changing conditions, reconfigure processes, and optimize operations based on their semantic understanding of industrial relationships and contexts.

5.1 Semantic interoperability

The collaboration between ISO 23726-3 Industrial Data Ontology (IDO), ISO 10303-239/242 STEP, CFIHOS, DEXPI, and ISO 15926-4 creates a foundation for machines to understand industrial data across different systems. Each standard brings its own way of describing equipment, processes, and relationships. IDO provides the overarching structure for how industrial concepts connect, while STEP handles the geometric and product data that engineering systems need. CFIHOS focuses on the handover data between project phases, and DEXPI manages process plant information exchange. ISO 15926-4 adds the temporal aspects and lifecycle data management that industrial operations require.

When these standards work together, they enable different software systems to interpret the same piece of equipment or process in consistent ways. A pump described in a DEXPI file can be understood by a STEP-based CAD system because both reference the same underlying concepts defined in IDO. This shared understanding eliminates the translation errors that typically occur when data moves between engineering, operations, and maintenance systems.

5.2 Machine reasoning in safety-critical applications

Industrial systems demand more than simple data exchange - they need intelligent interpretation of complex relationships. The semantic technologies embedded in these standards allow machines to infer safety implications from equipment configurations and operational states. When a temperature sensor reading combines with pressure data and equipment specifications, reasoning engines can determine if conditions violate safety parameters without explicit programming for every possible scenario.

In safety-critical applications like chemical processing or nuclear facilities, this reasoning capability becomes essential. The standards provide formal logic structures that let systems understand cause-and-effect relationships. If a valve position changes in a way that could create unsafe conditions, the reasoning system can trace through the semantic relationships to identify potential hazards and trigger appropriate responses. This goes beyond simple rule-based systems to genuine understanding of industrial processes.

5.3 Cross-industry data integration

The framework created by these standards supports data exchange across industries through common descriptors and dialog patterns. Manufacturing data from automotive production can interface with process industry systems from chemical plants because both use the same fundamental concepts for describing equipment, materials, and operations. The standards define not just what data looks like, but how conversations between systems should happen.

This cross-industry capability becomes valuable in complex supply chains where different industries must coordinate. A pharmaceutical company working with chemical suppliers and packaging manufacturers can maintain data consistency across all partners because each uses compatible semantic models. The dialog descriptors ensure that when one system asks for equipment status, all other systems understand exactly what information to provide and in what format.

5.4 Integration with Digital Languages

These standards integrate smoothly with major digital languages and platforms used in industrial automation and engineering. They provide clear mappings to common programming interfaces and data formats, allowing existing systems to adopt semantic capabilities without complete rebuilds. The standards define how their semantic models translate into practical implementations using technologies like OPC UA, AutomationML, and web services.

This integration means that a control system programmed in ladder logic can still participate in the semantic data exchange, and a machine learning algorithm can access the rich semantic context it needs for better decision-making. The standards bridge the gap between traditional industrial systems and modern digital technologies.

5.5 Autonomous Adaptation

Perhaps most importantly, these standards enable systems to adapt autonomously to changing conditions and contexts. Rather than requiring manual reconfiguration when operational parameters change, systems can use the semantic relationships to understand new situations and adjust accordingly. When a production line switches from one product to another, the semantic models help systems understand what equipment reconfigurations are needed and what new safety considerations apply.

This autonomous adaptation capability extends to maintenance and optimization as well. Systems can recognize when equipment behavior deviates from normal patterns and automatically adjust monitoring parameters or maintenance schedules based on semantic understanding of equipment relationships and operational context. The result is more resilient and efficient industrial operations that require less human intervention for routine adaptations.

6 Mapping existing data models to IDO

This section explores mappings between existing industrial data models and the Industrial Data Ontology (IDO). We consider four standards: the ISO 10303 STEP family parts 239 and 242; CFIHOS Semantics; DEXPI (Data Exchange in the Process Industry); and ISO 15926-4 reference data for process plants.

We would like to emphasize the importance of semantic interoperability in modern industrial environments, where systems must communicate in a way which preserves the meaning and context of shared data. The mapping work described here forms a foundation for enabling industrial standards to work together through common semantic frameworks.

6.1 ISO 10303-239/242 to IDO

ISO 10303-239 and ISO 10303-242 are both parts of the STEP (Standard for the Exchange of Product Data) family, but they serve different purposes. Industrial Data Ontology (IDO) creates semantic frameworks for understanding and connecting industrial concepts.

ISO 10303-239, known as Product Life Cycle Support (PLCS) has already been described in the Arrowhead fPVN Deliverable 3.1, Section 6.12 “ISO 10303-239 Product Life Cycle Support”[1], including its data modeling language and exchange formats. It focuses on product lifecycle management and support information. This standard covers the entire product lifecycle from initial conception through final disposal, including maintenance and support data, configuration management, reliability and safety information, and service history documentation. It is primarily utilized in aerospace, defense, and complex manufacturing industries where detailed lifecycle tracking and support documentation are critical for operational success and regulatory compliance. IDO can build upon these structured relationships to create richer semantic models that help different industrial systems understand and share lifecycle information.

On the other hand, ISO 10303-242, is called Managed Model-based 3D Engineering (Arrowhead fPVN Deliverable 3.1, Section 6.15 “ISO 10303-242”[1]). This standard specializes in the exchange of 3D CAD data along with metadata, including 3D geometric models with manufacturing information, product and manufacturing information (PMI), annotations and dimensions, and material specifications. It serves general manufacturing, automotive, and aerospace industries that require precise 3D model communication and collaboration. IDO creates ontologies that connect design intent with manufacturing processes and quality requirements.

The differences between ISO 10303-239 and ISO 10303-242 lie in their data focus and application scope. While ISO 10303-239 handles broader organizational and lifecycle data management, ISO 10303-242 specializes in 3D geometric and manufacturing data exchange. ISO 10303-242 also represents a newer evolution toward model-based enterprise approaches, whereas ISO 10303-239 addresses the organizational aspects of product support throughout its entire lifecycle. These two standards can complement each other in digital manufacturing environments where both 3D modeling and lifecycle management are essential.

The relationship between these standards and IDO centers on semantic interoperability. While the ISO standards provide the technical structure for data exchange, IDO adds the semantic layer that helps machines understand what the data means. IDO can reference the formal models from these standards to ensure its ontologies align with established industrial practices. This alignment helps organizations integrate data from different sources while maintaining consistent meaning across their industrial systems.

6.2 CFIHOS Semantics to IDO

CFIHOS is an association of 82 companies which made a standard for data handover of engineering projects, by EPC companies, to owner/operators. However, there are many other initiative projects for semantic software development. They are working on digital twins or engineering databases central on projects. To support those projects, CFIHOS has started a semantics team. The CFIHOS Semantics project takes the CFIHOS RDL and makes it available in semantic web format. This will contain Plant Item classes (> 2000); Property (relationships and attributes) classes (> 10,000); Object models (~100); Document types (350); Grouping (e.g., property groups); Picklists with values; and Units of measures.

The data model will be compliant to IDO (ISO 23726-3). The status of the CFIHOS Semantics project is early start. The first release of the semantic database is in a couple of months, probably at address https://data.cfihos.org/.

In the following use case example of the IDO / Arrowhead implementation we show how the two ontologies work together in an instance example. Figure 1 shows the simplest data structure anything industrial has: an object's name. For example, process plant items use tag name; for suppliers, maybe a serial number.

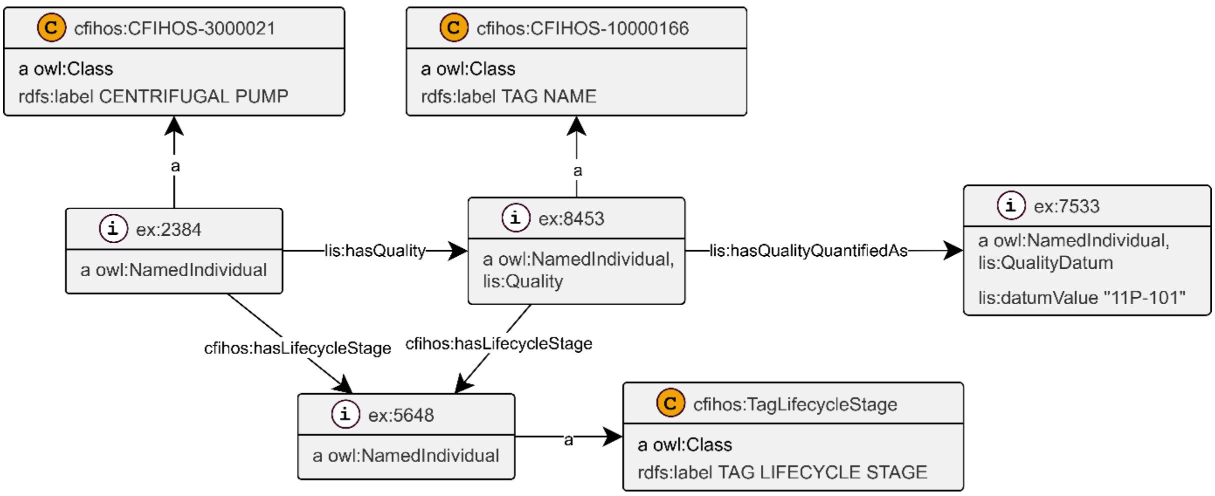

Figure 1. Illustration for IDO interoperability: an instance with a name

The combination IDO / Arrowhead ontologies is this:

- IDO classes and relationships for all data structures
- CFIHOS / DEXPI / SysML / STEP classes and properties
- Lifecycle Stages.

Lifecycle stages can be handled from different perspectives, like design/technological vs. equipment/procurement. We also need to consider a semantic solution using restrictions and lifecycle stage classifications. This addresses the challenge of having different permissible properties for the same object class in different lifecycle phases.

6.3 DEXPI to IDO

The main goal is to achieve interoperability between DEXPI and the semantic world. This involves creating a specification that allows DEXPI to communicate with the semantic world without adding an additional semantic layer. The project scope includes defining how to go from DEXPI to IDO, focusing on process and plant modules of DEXPI. The aim is to create a common solution for DEXPI 2.0, integrating process and plant specifications into one interface for interoperability.

The scope of DEXPI in the Arrowhead fPVN project includes engineering content, topology, and core elements from both process and plant. This includes process structure, unit operations, streams, properties, plant structure, apparatus, machines, piping, and instrumentation classes. On the other hand, graphical representation, such as block flow diagrams, process flow diagrams, and administration information (e.g., designer details, release information), are not included in the scope of DEXPI part in the Arrowhead fPVN project. It is important to map top classes in DEXPI, focusing on the relationship concepts between process and plant. This involves defining how elements from the process and plant domains relate to each other in the IDO world. This includes mapping the relationships between different elements, such as unit operations, streams, and plant structures, to ensure interoperability in the IDO world. For instance, a pump lifecycle includes the pumping activity, process streams, and properties. These elements must be modelled accurately.

The transformation from DEXPI to IDO should involve RDF, RDFS, OWL, and SKOS. These technologies are essential for defining the interface specification and ensuring interoperability between DEXPI and the semantic world. Unique identifiers ensure accurate mapping and interoperability between ontologies. It is necessary to rely on all clauses in the pnb (plants & bytes GmbH) ontology. These clauses provide the necessary framework for ensuring that DEXPI aligns with IDO's standards. Besides unique identifiers, clear and consistent labels, and corresponding classes ensure that all data is easily understood and accurately represented and integrated into the semantic world.

Due to restrictions in IDO – properties can only be assigned to activity profiles, not directly to activities –, activity profiles are required for property assignments. The introduction of activity profiles helps resolve the IDO restrictions which require directly linking properties to activities. The upcoming IDO guidelines also need to address the restrictions in IDO in order to find a solution that works within these constraints. It is clear that some elements belong in a domain ontology rather than in the DEXPI ontology itself. For instance, it is worth exploring the pattern: Activity → Activity Profile → Quality → Quantity Datum (with value, unit, quantity kind).

During our debates, a consensus emerged that it is better to create separate domain ontologies rather than mixing everything with DEXPI process ontology. The proposed domain ontologies are: Process, Properties, and Identification. In order for this to work, proper prefix conventions and namespace organization must be formalized.

Physical and calculated properties can be modelled using IDO as the upper ontology. A new *shortcut* modelling approach was demonstrated for the pumping head use case, with focus on simplifying structures while preserving semantic accuracy. There are two modelling variants: a full “explicit” model and a shortcut pattern for differential head in a pumping process. It is important to separate instances and ontology patterns in Activity Profiles and Process Activity. We also agreed that complex cases like delta values need to be addressed. An open question is where to draw the boundary between general IDO elements, DEXPI specifics, and domain ontologies.

Going forward we agreed that there is a need for graphical diagrams to complement the Excel-based structure and improve clarity, better naming strategies, and clearer documentation for SHACL and XML integration.

6.3.1 Process Modeling with IDO

This section illustrates a reduced scope of the plant lifecycle story, forming a small part of the Plant model.

The example of Figure 2 covers a pump as an excerpt from the pump lifecycle. The pump is part of the designed plant. The objects and relations within the lifecycle – like processes and plant objects – are not covered in this example. They should be added in a later phase of this project. In the graph –

- pale blue bubbles represent classes of DEXPI plant;
- the yellow bubbles at the bottom are classes of ISO 15926-4;
- the upper row contains four lis (IDO) bubbles and one OWL bubble;
- the row with the 5 other colored bubbles represent instances of a DEXPI export file.

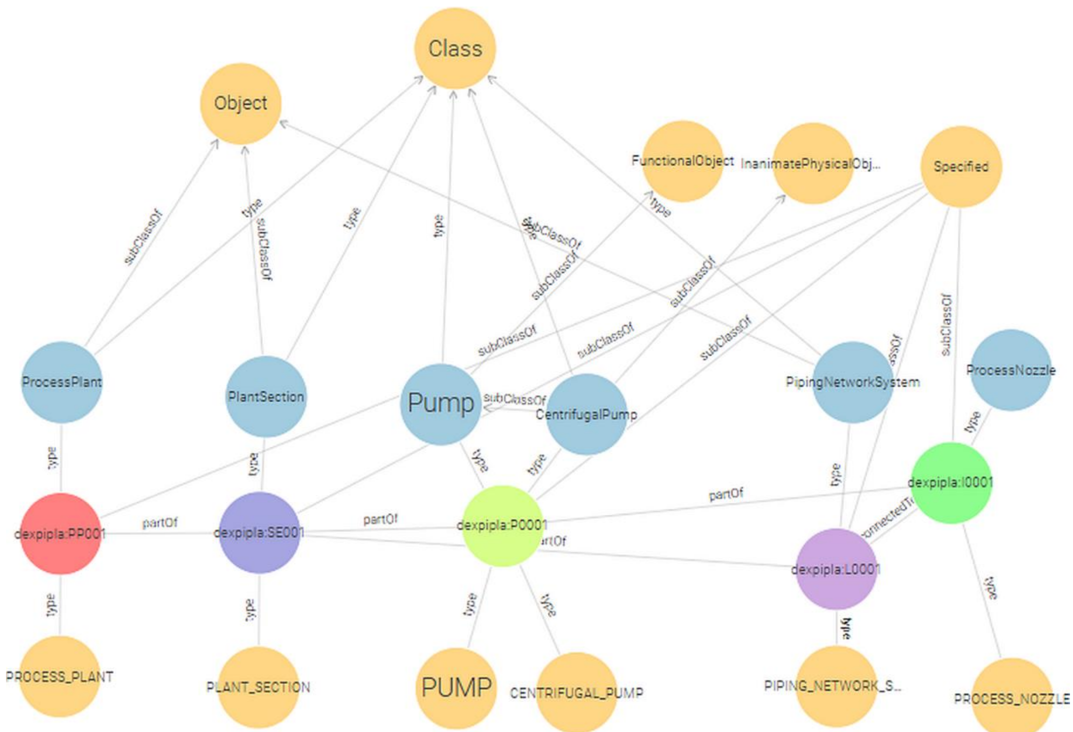

Figure 2. Reduced use case Pump lifecycle story: a part of the Plant model

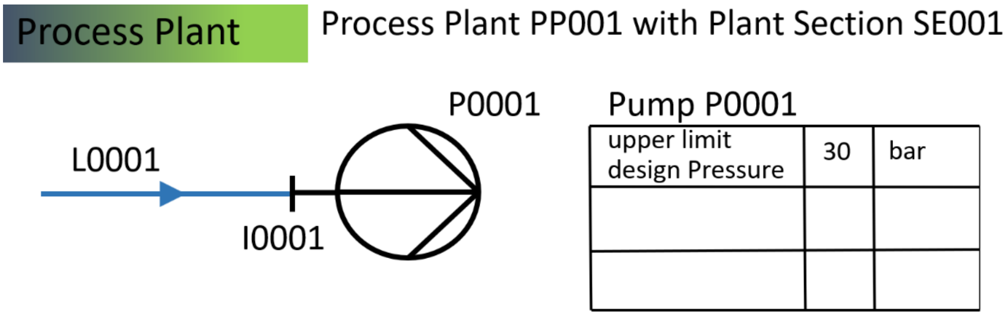

Figure 3. Process Plant PP001 with Plant Section SE001

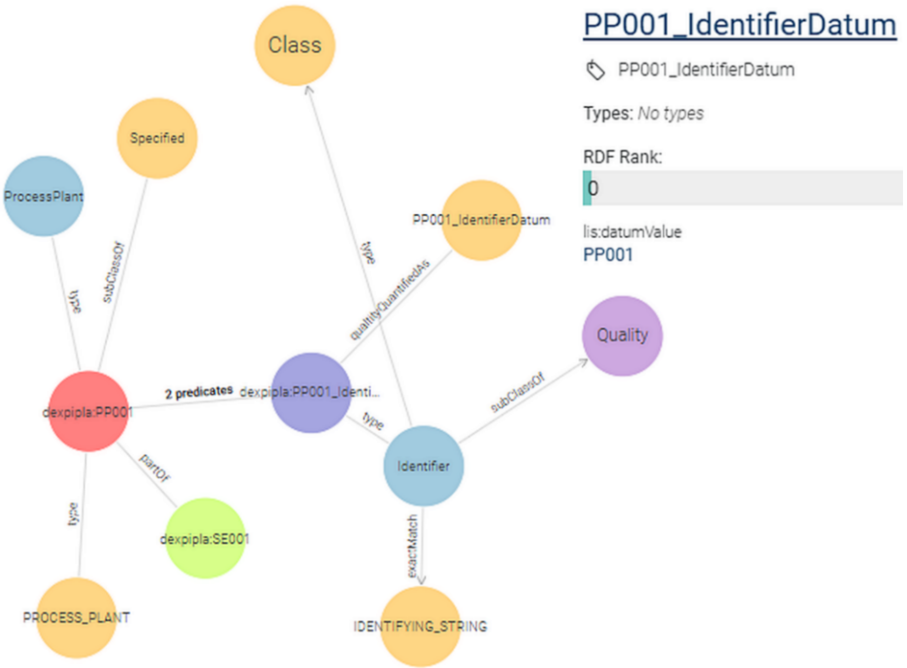

Figure 4. Identification of the Process Plant PP001

This example could be used to develop some typical modelling pattern for:

- What has to be on class or instance level?
- How to relate classes and / or instances to the lifecycle?
- How to model part of relations, e.g., SE001 is part of PP001?
- How to model identifier, like the tag number, in this case of the pump?
- How to model properties with physical quantity relation?
  (In this case upper limit design pressure.)
- How to model the process nozzle I0001 as part of the pump P0001?
- How to model the connection of the piping network system to the process nozzle?

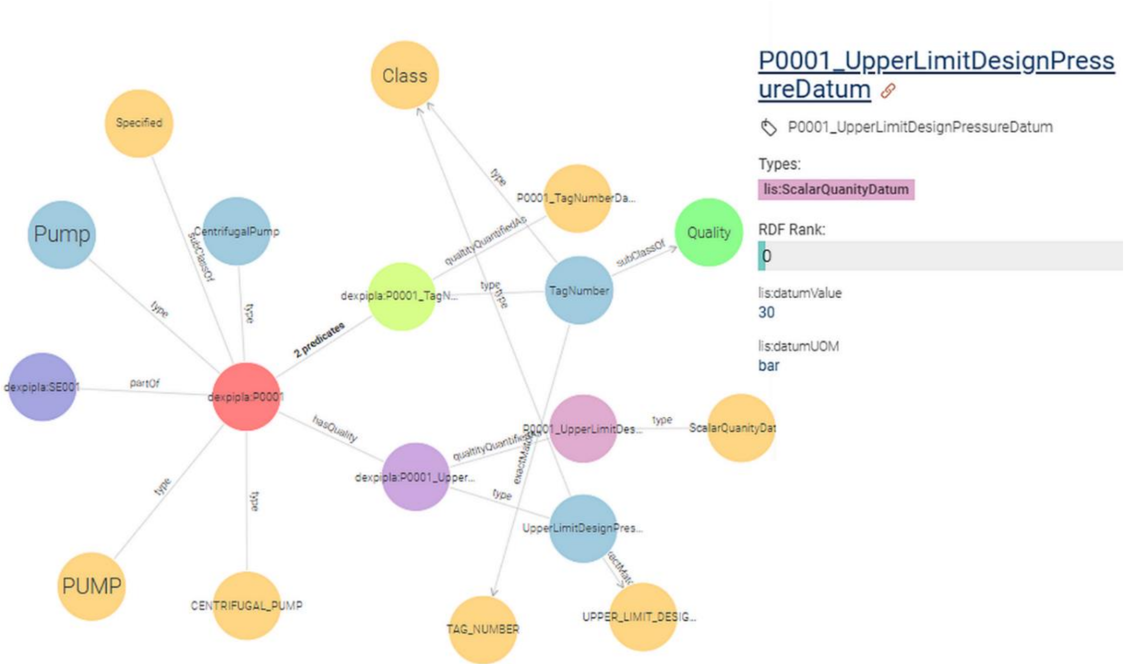

Figure 5. Pump P0001 with TagNumber and UpperLimitDesignPressure

6.4 ISO 15926-4 to IDO

This section examines how to map reference data from ISO 15926-4 (Reference data for Process Plant use-cases) to the Industrial Data Ontology (IDO). The goal is to create a bridge between existing industrial standards and the IDO framework for better data interoperability. ISO 15926-4 contains reference data that serves as a foundation for process plant applications.

The “ISO 15926-4 Ed. 4 All” spreadsheet (Figure 6) acts as a “lower upper ontology” – it extends the basic ISO 15926-2 framework with domain-specific classes that remain independent of particular standards or suppliers. These classes, called "core classes" in ISO 15926-1 terminology, provide a neutral foundation that different organizations can build upon.

ISO 15926-6 establishes a basic rule for writing class definitions. Every definition must follow this format:

```
A “Class in focus” is a “Name of superclass” that/where “description of what makes the “Class in focus” a subclass of “Name of superclass”.
```

This structure is essential for building taxonomies and helps users understand what each class represents. For example, a "centrifugal pump" would be defined as "a pump that uses a rotating impeller to increase fluid pressure."

Another fundamental rule requires all class names within the taxonomy to be unique. This uniqueness enables the creation of Uniform Resource Identifiers (URIs) that serve as primary identifiers for accessing the Reference Data Library (RDL). As the RDL grows, some existing names may need updates to maintain uniqueness and clarity.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
UniqueNumber	**UniqueName**	**Synonym1**	**Synonym2**	**TextDefinition**	**Source**	**Notes**	**SuperClass1**	**SuperClass2**	**SuperClass5**	**ISO15926-2Entity**	**Classification1**	**Classification2**	**Name in Source 1**	**Name in Source 2**
24655	2 CONCECUTIVE CONCENTRIC PIPE REDUCERS			A <2 CONCECUTIVE CONCENTRIC PIPE REDUCERS> is a <PIPE REDUCER ASSEMBLY> consisting of 2 concecutive concentric	NORSOK CAD symbol libraries, Z-CR-004,		PIPE REDUCER ASSEMBLY			ClassOfInanimatePhysicalObject	PIPING			
24656	2 CONCECUTIVE ECCENTRIC PIPE REDUCERS			A <2 CONCECUTIVE ECCENTRIC PIPE REDUCERS> is a <PIPE REDUCER ASSEMBLY> consisting of 2 concecutive eccentric	NORSOK CAD symbol libraries, Z-CR-004,		PIPE REDUCER ASSEMBLY			ClassOfInanimatePhysicalObject	PIPING			
24659	2 HOURS TEST RUN			<2 HOURS TEST RUN> is the <Activity> whereby the pump shall be run on a test stand at the rated flow until oil			Activity			ClassOfActivity	ACTIVITY OR EVENT			
24765	20-CYLINDER ENGINE			A <20-CYLINDER ENGINE> is a <RECIPROCATING PISTON			RECIPROCATING PISTON			ClassOfInanimatePhysicalObject	ROTATING EQUIPMENT			
24708	2-CYLINDER ENGINE			A <2-CYLINDER ENGINE> is a <RECIPROCATING PISTON			RECIPROCATING PISTON			ClassOfInanimatePhysicalObject	ROTATING EQUIPMENT			
24866	2D COORDINATES			<2D COORDINATES> is a <Property> that entails the distance from a common point			Property			MultidimensionalProperty	PROPERTY AND STATUS			

Figure 6. Excerpt from the "ISO 15926-4 Ed. 4 All" spreadsheet

The mapping process uses a simplified spreadsheet format based on ISO 15926-4 Edition 4. Each organization receives separate spreadsheets to avoid conflicts during mapping. The key columns include:

1. UniqueNumber: We will have to add a prefix to the new proposals.
2. UniqueName: In UK English where possible to comply with the ISO rules. This is the name that represents the unique key for humans to understand what the class is about.
3. Synonym1: An alternative name for the class. The same rules apply as for UniqueName.
4. Synonym2: Same as for Synonym1
5. TextDefinition: Together with UniqueName, this is the most difficult part of adding reference data, and will normally require a lot of work. PCA will assist here.
6. Source: Add this if applicable
7. Notes
8. SuperClass1: Add one from the existing set, or your proposal for extensions
9. SuperClass2: As for SuperClass1
10. ISO15926-2Entity: This will be proposed by PCA and reviewed with the proposer.
11. Classification1: This is the subject area to which the class belongs. This will be proposed by PCA
12. Classification2: As for Classification1 as there in special cases will be required to have multiple subject areas.
13. Name in Source 1: This is to be added by the proposing organization. By assigning this as a new identifier you confirm that this is the same class. Reading the TextDefinition from Column 5 is vital to get this right.
14. Name in Source 2: As for Name in Source 1

The current focus includes six main areas: piping (ISO and DIN standards), manual valves, instrumentation (loops and block diagrams), pumps, process equipment, and properties. Several categories from the full ISO 15926-4 spreadsheet are excluded from the fPVN deliverables: geometry data from ISO 15926-3, currencies, citizenships, and EXPRESS format type data.

Organizations should only create mappings when they are confident the definitions match exactly. Reading the TextDefinition is important – this is where the real semantic meaning lives. If no suitable match exists in the current reference data, organizations can propose new classes at the bottom of their spreadsheet. The process prefers definition accuracy to simple name matching. Many concepts exist in industrial systems under different names, but what matters is whether they represent the same underlying reality. A "control valve" in one system might include an actuator, while another system treats the valve body and actuator as separate components. These definitional differences must be identified clearly.

Recent analysis shows that harmonizing different industrial standards is not straightforward. A comparison between IEC 61987 and ISO 15926-4 instrumentation classes found only 6% exact matches. This low overlap isn't necessarily problematic – it reflects genuine differences in how different communities conceptualize industrial equipment.

The key insight is that different standards serve different purposes. IEC standards often focus on device and product markets, while ISO 15926 represents functional relationships in process applications. These different scopes create legitimate variations in how equipment is classified and defined.

The reference data mapping represents the first step toward broader interoperability. This approach recognizes that complete standardization may not be achievable or even desirable. Instead, the goal is to create sufficient common ground for practical data exchange while preserving the specialized knowledge embedded in different industrial communities.

7 Creating new ontologies with IDO as upper ontology

When creating new ontologies, the must observe a few key principles. Any ontology in the OBI series must fully comply with OWL 2 Direct Semantics, ensuring that its constructs can be interpreted and reasoned over using standard DL reasoners. All models and patterns are required to follow the W3C Linked Data approach by representing their graphs in RDF, helping uniform data exchange. To ensure semantic harmony, each new ontology must also build on the Industrial Data Ontology (IDO) by subclassing or sub-propertying from its classes and properties wherever appropriate. Any additional alignments (e.g., to external vocabularies) must be captured in dedicated OWL alignment modules with clear versioning, namespaces, diagrams and mappings.

Quality assurance is enforced through the lifecycle of ontology evaluation and rule-checking. Every OBI ontology must undergo both syntactic validation and logical consistency tests – catching issues like unintended synonyms, orphaned terms, or missing human-readable annotations. Where domain constraints or template-driven transformations are used (SHACL, SWRL, OTTR, R2RML, etc.), evidence must be provided that they preserve IDO-compatible consistency and soundness. To keep reasoning tractable, OBI ontologies are encouraged to offer “light profiles” calibrated for efficient performance on common DL reasoners.

Modularity is a core design principle: vocabularies are split into reusable components (such as core, scheduling, metadata, provenance), stored under well-defined namespaces and subdirectory layouts (OWL, SHACL, SWRL, property-chains). A uniform naming convention mandates UpperCamelCase for classes and lowerCamelCase for properties (with boolean properties or data-value properties clearly suffixed). This consistency simplifies both human understanding and automated processing.

Finally, ontologies in this series are intended to align seamlessly with engineering information models across the lifecycle of industrial systems. They must be driven by concrete use cases and competency questions, ensuring that each term and pattern addresses real-world requirements. All artifacts (OWL files, shapes, rules) are versioned and maintained in Git repositories under the PCA namespace, enabling transparent evolution, traceability, and collaboration.

8 Arrowhead fPVN Use Case 2.9: Digital Twin

Use Case 2.9 is a key use case for the technological development of the transformation of the DEXPI and the STEP standard models. The technology and its usage is described below.

The scope of this use case is a Digital Twin of a Paper Mill Subsystem at Stora Ensos Paper Mills. The Digital Twin is a digital representation of the Physical Twin. Ideally it represents many different viewpoints and the information of interest in these, all in a pre-consolidated and integrated manner. Data interpretation, consolidation and integration may be done at run time utilizing the semantic web technology relations between different data elements. Due to limitation of processor and network performance, it is most often preferable to carry out some pre-compilation of the information. The pre-compiled data is stored and managed in the Digital Twin database.

In the interpretation and consolidation work, the main pillars of the Arrowhead fPVN project are naturally considered and, in this case, the “Utilization of major industrially accepted data models” such as ISO 10303-STEP Core, CFIHOS, ISO 15926-4, DEXPI, IEC62541 and OPC-UA is in scope.

To enable the presentation of graphic and geometric information in combination with data from different data sources, an evaluation has been made of potential software called eShare from the software provider Cadmatic Oy. This Software will be used as a 3D graphical user interface to the Digital Twin content. The evaluation has been done over year 1 on installed software at one of the mills in Sweden to test different formats. Cadmatic will use the endpoint of the used graph database and the W3C SPARQL definition to search and extract data from the knowledge graph managed in the graph database.

During year 1 of the fPVN project the following very high-level architecture was defined and it has been the starting point of the further work carried out during year 2 of the UC2.9.

Figure 7 explains the interaction within the digital twin concept:

P1 Physical resources used for construction (changed functionality)
P2 Physical resources used for maintenance (maintained functionality)

I1 Design and Construction information that is used in combination with P1 to construct the system.
I2 State of the constructed system during construction, including non-conformance records.
I3 Requirements, conceptual solutions, chosen detailed designs and components, installed physical components, justifications, verifications and validation information etc.
I4 Machine readings of data originating from sensors. May include readings classified as issues/alarms.
I5 Out of scope, since the Digital Twin only used for monitoring

I6 Identification of maintenance components of interest for CMMS/MRO systems, as well as work requests and other triggers.
I7 Work Order information used in combination with P2 to maintain the system functionality.
I8 Feedback and confirmation of carried out work and exchanged components.
I9 Feedback and confirmation of carried out work and exchanged components.
I10 As Is configuration of system in scope for rebuild used as a starting point for a redesign/construction project.

I11 System configuration (historic, current, investigated or planned) for systems in scope for analysis/simulation.
I12 Analysis/simulation results with identification information to support context identification.

E1 Requirements, conceptual solutions, components and documents.
E2 Chosen detailed designs, components and documents.
E3 Purchase Order, needed technical information from WO etc.
E4 Feedback and confirmation of carried out work and exchanged components.
E5 Equipment behavior information related to Functional Location.

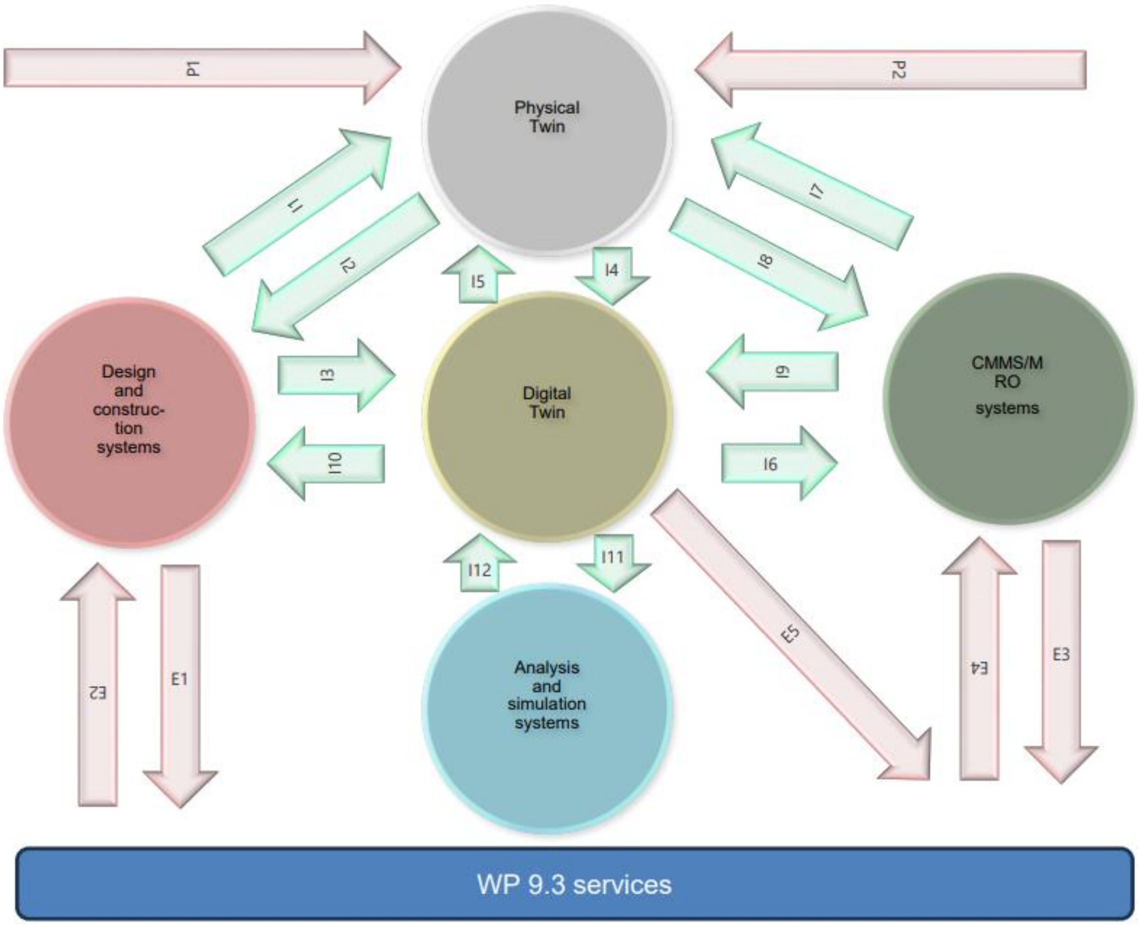

Figure 7. Proposed systems landscape and interaction points of Year 1 of the Arrowhead fPVN project

The system architecture of UC 2.9 has matured during the second year of Arrowhead fPVN. This architecture is verified using a proof of concept based on laboratory equipment from Luleå Technical University. The architecture is already utilizing results from fPVN WP2 and WP3, and it has been evident that the combination between knowledge graphs / formal semantic technology and the Arrowhead framework technology gives possibilities that are very hard or impossible to achieve with traditional implementation technologies.

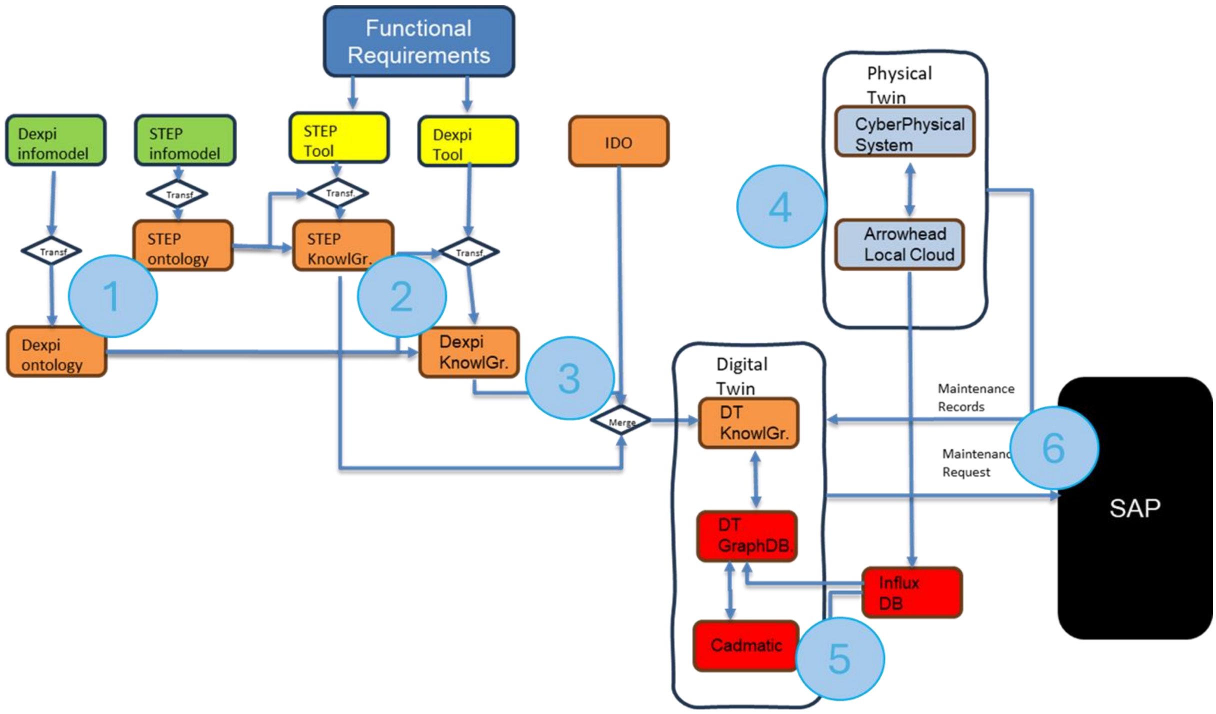

Figure 8. System architecture during the POC from Year 2 of the Arrowhead fPVN project

The utilization of Year 2 WP3 (and WP2) results is put in context of the UC 2.9 in Table 1. The activities are defined in Figure 8.

Table 1. The utilization of year 2 results of Arrowhead fPVN WP2 and WP3

Activity	Utilizing Results	Details	Involved Partners	Benefits
Activity 1	WP3	Transformation of Industrial standard models (DEXPI and STEP) to W3C OWL 2 DL using the XMI representation of the standards and Copilot LLM	Stora Enso, LTU, TBH Konsult, Semantum, Plants&Bytes	Uniform technology (OWL 2 DL) and IDO technology compliance
Activity 2	WP3 and WP9	Transformation of Industrial standard data to W3C OWL 2 DL individuals using XMI and Proteus format	Stora Enso, LTU, TBH Konsult, Semantum, Plants&Bytes	Creates Knowledge Graphs including both ontology and individuals
Activity 3	WP2 (Luleå Arrowhead Go implementation)	Aligning DEXPI and STEP ontology using IDO as upper ontology and alignment methods, importing into GraphDB	Stora Enso, LTU, TBH Konsult	Gives DB functionality to the Knowledge Graph using W3C SPARQL

Activity 4	WP2 (Luleå Arrowhead Go implementation) and WP4	Establish Arrowhead functionality into the Physical Twin, use FluxDB to monitor sensor readings	Stora Enso, LTU	Use of existing OPC-UA connectivity in the Arrowhead framework
Activity 5	WP2 (Luleå Arrowhead Go implementation)	Connect Cadmatic e-Share for 3D visualization, integration through GraphDB SPARQL endpoint	Stora Enso, LTU, TBH Konsult, Cadmatic	Visualization of search results and selections in the DT in GraphDB
Activity 6	WP2 (Sinetiq Arrowhead implementation)	Creation on Work Requests sent to SAP and Records sent as answers from SAP	Stora Enso, Sinetiq, LTU, TBH Konsult	Capturing changes impacting not Arrowhead monitored components

The result of the WP3 efforts during year 2 has been the base for the implementation of UC 2.9 proof-of-concept as well the ongoing work of the project.

9 Arrowhead fPVN Use Case 3.9: Process industry

The aim of Arrowhead fPVN Use Case 3.9 "Interoperability for technical information exchange in process industry"[16] is to improve pulp and paper process industry information exchange through interoperability of data, digital processes, and system integrations. The current section presents a Process Flow Diagram (PFD) standardization case study, whose objective is to evaluate means to transform process flow diagrams into a standardized data model form. The work is related to Arrowhead fPVN D4.2, where the technical adapter design and implementation is reported in Section 5.4.2 "Balas to DEXPI"[17]. Balas is a steady-state simulation software developed at VTT, used for modeling chemical processes, particularly in the pulp and paper, food processing, and biochemical industries.

Before transforming the Balas simulation model to a standard form, a mapping analysis and identification of similarities between concepts of data models must be explored. Here, we focus on POSC Caesar Association (PCA) Product Lifecycle Management (PLM) Reference Data Library (RDL) ontology and DEXPI Process standard representations. After a short introduction to Balas steady-state simulation software, we present the PLM RDL ontology, the DEXPI Process standard, and the mapping possibilities between Balas and the standardized data models. We conclude our study with proposals for additions to the presented standardized data models.

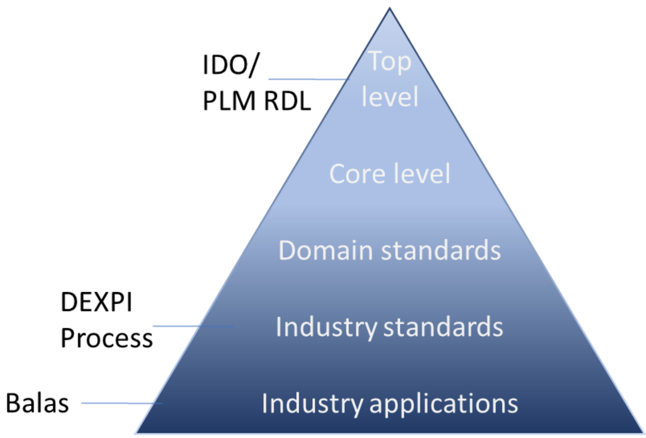

Figure 9. Semantic positions of the standardized data models and Balas.

Semantic positions of the standardized data models and Balas are presented in Figure 9. Balas is a very specific simulation tool; the DEXPI Process is designed for data exchange in the process industry; and IDO/PLM RDL is applied in several industries as a top-level ontology. These generic detail differences also influence our mapping possibilities and later conclusions.

9.1 Balas

Balas[18] is a steady-state simulation software developed by VTT. It is used for modelling chemical processes, especially in the pulp and paper, food processing, and biochemical industries. Balas has over 40 years of history in analyzing complex processes, energy, and mass balances. The current version used in this study is Balas 3.3.

Balas models are built with Microsoft Visio using Balas symbols, and streams connecting the symbols. The symbols represent the actual process unit operations (e.g., pump, heat exchanger, or evaporator). In total, about 220 different symbols can be found in the Balas library. Each symbol can be linked to only one acceptable calculation module. The calculation module describes the operation of the process unit. Typically, each symbol contains a few possible calculations modules – at a maximum of ten. The Balas database contains 118 different calculation modules for describing specific unit operations. Thus, it is possible to have multiple symbols with the same calculation module in operation. For this reason, the symbols are only important for the visualization of the process flowsheet. The transformation from the Balas model to standardized form will only be based on calculation modules. All calculation modules are listed in Table 2.

Table 2. Balas calculation modules.

Aqua heat recovery	Element flow calculator	Mix and fix outlet temperature
Air flotation dryer	Electric generator	COD and HHV monitor
Approach temperature air heater	Electrostatic precipitator	Black-liquor evaporator
Approach temperature boiler surface	Evaporator	Multistage screen
Approach temperature heat exchanger	Evaporator 2	OptiDry Vertical drying unit
Aeration lagoon with reactions	Flow and consistency controller	Electrostatic precipitator
Boiler	Counter-current hex with flow calculation on one	Press
Burner	Saveall filtrate tank	Pump/compressor
Separation with known filtrate composition	An adiabatic flash tank/mixer	Pulper
Indirect heating with steam	A flash tank/mixer with given vapour-to-feed frac	Reactor
Slaking/causticizing	Flotation shell	Recovery boiler
Caustification reactor	Flotation shell 2	TMP-refiner
Recovery boiler	Flash dryer	Repulping drum
Cooling tower with two fillings	Former	Reduction valve
Condensor	Lime kiln	Reduction valve with flow set
Condenser for producing vacuum	Flow controller	Steam box
Consistency controller	Gas compressor	Scrubber
Cooling tower	Gas IR-dryer	Steam heater
Cooling tower with fan	Gas turbine	Steam heated cyl. and vac. roll drying unit
Counter-current heat exchanger with phase change	Grinder	Tank where lime reacts with H2O to produce Ca(OH)
Cooling tower with volumetric heat transfer coeff	Headbox	Steam consumer
Centrifugal cleaning	Heater/cooler, define thermal duty	Smelt dissolver
Centrifugal cleaning, parameter inlet massflow	Heater/cooler, reverse calculated flow	Separation with accept ratio- and K-values
Cylinder dryer	Heat source	Source unit
Dual consistency splitting module	Absorption heat pump	Combined mixer and divider
Dry debarking drum	Counter-current heat exchanger with no phase chan	Split massflow
Deculator	Impingement dryer#2	Steam turbine
Desuperheater	Impingement dryer	Steam turbine with outlet pressure and temperatur
Dewatering element	Source unit (can be set by destination unit)	Direct heating with steam
Disk filter	Isothermic reactor	Sub-process terminal module
Saveall disc filter	Washing filter	Sub-process module
Saveall disc filter with shower waters	Batch digester	Super batch digester
Discfsorp	Lime kiln	Supplementary firing
Smelt dissolver	Measurement point	Storage tank
Divider	Membrane unit	Screening module with thickening ratio
Stream divider	Mixer with phase equilibrium calculation	Screening module with reject ratios
Steam heated cyl. drying unit	Mixing chest with sweetener stock	Simple separation module
Dry content controller	Storage tank with several inflows and outflows	Heater/cooler, define outlet temperature
Split a stream into two outlet streams	Mix and fix outlet massflow	Wire pit
Electric IR-dryer		

9.2 PLM Reference Data Library

The POSC Caesar Association initially published PLM RDL in 2022. PLM RDL is an outcome of two development projects, READI JIP and Krafla (Equinor)/NOA (AkerBP). The referred version is 0.9.1[19]. The detailed PLM RDL analysis was presented in the Arrowhead fPVN Deliverable 3.1 “Major industrial data models”[1]. In PLM RDL, there are ontology modules for “Equipment”, “Process”, “UoM”, “ChEBI”, etc. The dependencies of these ontology modules are presented in Figure 10.

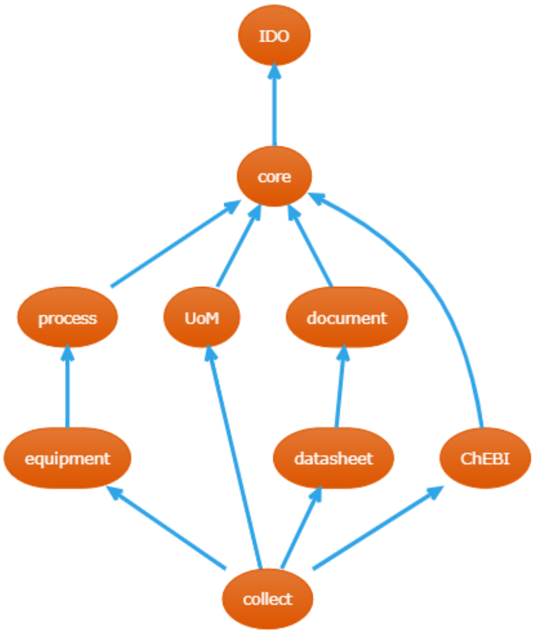

Figure 10. Ontology module dependencies in IDO/PLM RDL.
Source: https://rds-staging.posccaesar.org/ontology/plm/

The “Equipment” and “Process” ontology modules include relevant information related to the Balas symbols and calculation modules. PLM RDL has 234 equipment classes and 41 process classes. Since the model transformation must be done through the Balas calculation modules, we focus only on the PLM RDL processes. These 41 processes are organized in this document into three different levels (Level 0, Level 1 and Level 2) by their original classification (*sub-classOf* relations) in the PLM RDL ontology (Table 3). Level 0 represents the “Plant process”, which has 27 subclasses. These 27 PLM RDL processes are laid on Level 1. Six out of these 27 have further subclasses. These 14 PLM RDL processes are put on Level 2.

Table 3. PLM RDL processes.

Level 0	Level 1	Level 2
Plant process	Measuring	-
	Enthalpy Change	Heating
		Cooling
	Receiving	-
	Controlling	-
	Connecting	-
	Reaction	-
	Conducting	-
	Sending	-
	Closing	-
	Bifurcating	-
	Separating	Three Phase Separation
		Absorbing
		Desorbing
		Gas Liquid Separation
	Process Step	Unit Operation
	Deflecting	-
	Injecting	-
	Opening	-
	Reducing	-
	Sensing	-
	Containing	Tank Storage
	Displaying	-
	Fixing	-
	Transmitting	Transmitting Radio Frequency
		Transmitting Analog Signal
		Transmitting Digital Signal
		Transmitting Optical
	Expanding	-
	Weighing	-
	Transporting	-
	Entropy Change	Compression
		Pumping
	Barring	-
	Guiding	-

9.3 DEXPI Process

The DEXPI Process presentation format[20] was defined in 2023. It is a process addition to DEXPI information model, covering block flow diagrams (BFD) and process flow diagrams (PFD). The DEXPI Process 1.0 version was studied for the Balas model transformation purposes. In DEXPI Process, processes are organized into five super-subtype hierarchy levels. In total, 104 processes are identified on different levels of details (Table 4).

Table 4. The processes of DEXPI Process specification

Level 0	Level 1	Level 2	Level 3	Level 4
Process Step	Emitting	-	-	-
	ExchangingThermalEnergy	-	-	-
	Flaring	-	-	-
	FormingSolidMaterial	Extruding	-	-
		Pelletizing	-	-
	GeneratingFlow	Compressing	-	-
		Pumping	-	-
	IncreasingParticleSize	Agglomerating	-	-
		Crystallize	-	-
		Flocculating	-	-
	Mixing	Humidifying	-	-
		Kneading	-	-
		MixingSimple	-	-
		RotaryMixing	-	-
		StaticMixing	-	-
	Packaging	-	-	-
	ReactingChemicals	-	-	-
	ReducingParticleSize	Crushing	-	-
		CustomMilling	-	-
		Cutting	-	-
		Grinding	-	-
		Milling	-	-
	RemovingThermalEnergy	Cooling	-	-
	Separating	SeparatingByElectromagneticForce	SeparatingByElectrostaticForce	-
			SeparatingByMagneticForce	-
		SeparatingByFlash	-	-
		SeparatingByPhaseSeparation	SeparatingByCentrifugalForce	-
			SeparatingByCyclonicMotion	-
			SeparatingByGravity	-
		SeparatingByPhysicalProcess	Absorbing	-
			Adsorbing	-
			SeparatingByContact	-
			SeparatingByIonExchange	-
			SeparatingBySurfaceTension	-
		SeparatingByThermalProcess	Distilling	StabilizingDistilling
				StrippingDistilling
				VacuumDistilling
			Drying	-
			Evaporating	-
		SeparatingMechanically	Filtering	-
			Sieving	-
			Skimming	-
	Sink	-	-	-
	Source	-	-	-
	Splitting	SplittingEnergy	-	-
		SplittingMaterial	-	-
	SteeringFlow	BlowingDown	-	-
		Draining	-	-
		FeedingMaterial	-	-
		LimitingFlow	-	-
		PreventingBackflow	-	-
		RegulatingFlow	-	-
		RelievingOverpressure	-	-
		RelievingVacuum	-	-
		RelievingVacuumAndOverpressure	-	-
		ShuttingOffFlow	-	-
	Storing	StoringFluids	StoringInPressureVessel	-
			StoringInTank	-
		StoringSolids	StoringInSilo	-
	StoringElectricalEnergy	StoringInBattery	-	-
	SupplyFluids	-	-	-
	SupplyingElectricalEnergy	GeneratingACPower	-	-
		GeneratingCustom	-	-
		GeneratingDCPower	-	-
		GeneratingInFuelCell	-	-
	SupplyingMechanicalEnergy	DrivingByEngine	-	-
		DrivingByMotor	-	-
		DrivingByTurbine	-	-
	SupplyingSolids	-	-	-
	SupplyingThermalEnergy	Boiling	-	-
		GeneratingSteam	-	-
		HeatingElectrical	-	-
		HeatingInFurnace	-	-
	TransportingElectricalEnergy	-	-	-
	TransportingFluids	TransportingFluidsInChannel	-	-
		TransportingFluidsInHose	-	-
		TransportingFluidsInPipe	-	-
	TransportingSolids	TransportingSolidsContinuously	-	-
		TransportingSolidsDiscontinuously	-	-

9.4 Foundations for mappings

The information included in a Balas model is presented in Figure 11. The most relevant information for the mapping work consists of the name and calculation modules' descriptions. Besides these, other information is needed for a full model transformation.

Model			
	Symbols		
		Ports	
	Process/Unit operations		
		Calculation module	
			Properties
	Streams		
		Compounds (flow materials)	
			Compound attributes (e.g. pressure, temperature, total flow, consistency)

Figure 11. Information included in Balas models and the most relevant one for transformation (green)

To map the calculation modules to PLM RDL and DEXPI Process processes, we made a comparison. Some initial mapping might be possible to prepare with AI language models, but the little available information makes this approach impractical. Also, the exact use of calculation modules may differ slightly from what is in the description property. For these reasons, the mapping was done manually by Balas experts.

The mapping was prepared using Microsoft Excel sheets in which each Balas calculation module was listed in one column and the best-fitting PLM RDL process or DEXPI Process in the next column. As both ontologies contain generic processes, the aim was to find the most specific match for each calculation module, thus achieving the most accurate mapping, as all modules could just be mapped to the most generic one (Level 0 'Plant process' in PLM RDL or 'ProcessStep' in DEXPI Process).

9.5 Balas to PLM RDL

In the Balas to PLM RDL mapping we made the following observations. Only 13 (nine from Level 1 and four from Level 2) out of 41 process classes in PLM RDL were found usable for the calculation module mapping. Notably, Transporting does not have any use in the mapping. These nine Level 1 processes were matched with a total of 70 calculation modules, and the more detailed Level 2 processes were matched with 12 calculation modules.

The mapping had approximately one-to-one matches between processes and calculation modules. For example, the Balas Evaporator module can be described with the PLM Level 2 process heating. However, for the Balas model transformation to be complete and unambiguous, the transformed model should be able to reconstruct the complete Balas model. Problems with the mapping arose when either a single calculation module could be described with multiple processes, or multiple calculation modules were described with a single process.

When a single Balas calculation module can be mapped to multiple PLM processes, it might be necessary to create a combination mapping to all of these processes. For instance, "counter-current heat exchanger" with phase change can be mapped to the Level 1 process enthalpy change, and its sub-classes heating and cooling – as this depends on the Balas model. Unfortunately, a problem may arise when a PLM RDL-mapped model is used later in the project life cycle. A similar problem might arise when a single PLM process is mapped to multiple calculation modules, such as when mapping the PLM process ontology back to the Balas model. Figure 12 illustrates one possible case, which may occur when using PLM RDL encoding for Balas model back-and-forth transformation.

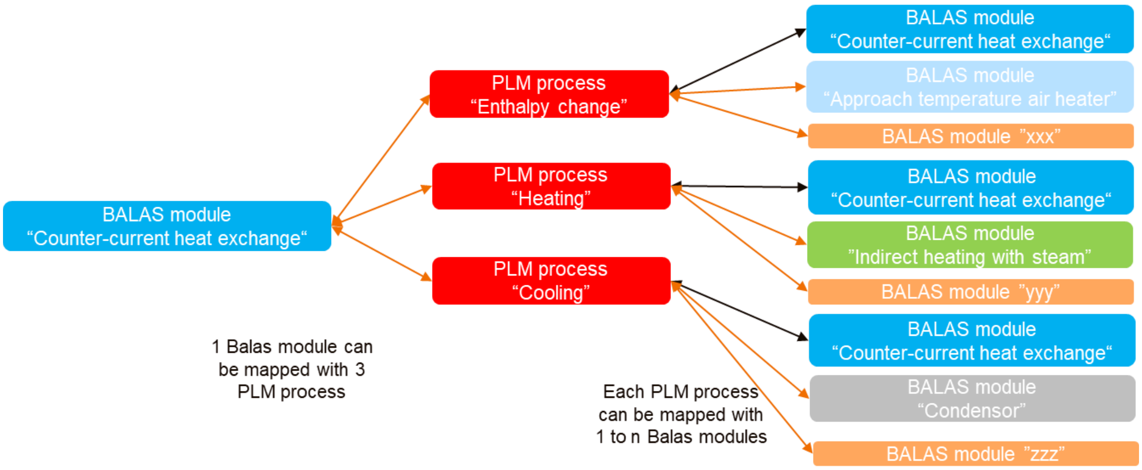

Figure 12. Example case of the Balas model mapping to PLM RDL ontology further.

We found three specific areas of Balas modelling which were either vague or unmappable to PLM RDL:

- First, drying modules are considered to be modules for water removal by means of heating. Drying modules could be described with a combination of existing PLM processes; for example, enthalpy change, heating, and separating. A more straightforward method would be to add drying processes to PLM RDL.
- Second, power production modules could be combined from several existing PLM processes, but it would be more straightforward to add power production-specific processes to PLM RDL.
- Third, pulp and paper industry-specific calculation modules cannot be mapped to any current PLM processes, and thus a section specific to such an industry should be added to PLM RDL.

The full list of Balas calculation modules and their mapping to PLM RDL along with proposals for new sections to PLM RDL are shown in Table 5.

Table 5. PLM RDL mapping to Balas modules along with proposed additions

	Mapped PLM process		
Balas module	**Main**	**Secondary 1**	**Secondary 2**
Aqua heat recovery	Enthalpy Change	Heating	Cooling
Air flotation dryer	drying specific		
Approach temperature air heater	Enthalpy Change	Heating	Cooling
Approach temperature boiler surface	Enthalpy Change		
Approach temperature heat exchanger	Enthalpy Change	Heating	Cooling
Aeration lagoon with reactions	Reaction		
Boiler	power production specific		
Burner			
Separation with known filtrate composition	Separating		
Indirect heating with steam	Heating		
Slaking/causticizing	p&p specific		
Caustification reactor			
Recovery boiler			
Cooling tower with two fillings	Cooling		
Condensor	Cooling		
Condenser for producing vacuum	Cooling		
Consistency controller	Controlling		
Cooling tower	Cooling		
Cooling tower with fan	Cooling		
Counter-current heat exchanger with phase change	Enthalpy Change	Heating	Cooling
Cooling tower with volumetric heat transfer coeff	Cooling		
Centrifugal cleaning	Separating		
Centrifugal cleaning, parameter inlet massflow	Separating		
Cylinder dryer	drying specific		
Dual consistency splitting module	Bifurcating		
Dry debarking drum	p&p specific		
Deculator			
Desuperheater	power production specific		
Dewatering element	drying specific		
Disk filter	p&p specific		
Saveall disc filter			
Saveall disc filter with shower waters			
Discfsorp			
Smelt dissolver			
Divider	Separating		
Stream divider	Controlling		
Steam heated cyl. drying unit	p&p specific		
Dry content controller	Controlling		
Split a stream into two outlet streams	Bifurcating		
Electric IR-dryer	drying specific		

Table 5 (Continued). PLM RDL mapping to Balas modules along with proposed additions

	Mapped PLM process		
Balas module	**Main**	**Secondary 1**	**Secondary 2**
Element flow calculator	Measuring		
Electric generator	power production specific		
Electrostatic precipitator	Separating		
Evaporator	Heating		
Evaporator 2	Heating		
Flow and consistency controller	Controlling		
Counter-current hex with flow calculation on one	Enthalpy Change	Heating	Cooling
Saveall filtrate tank	p&p specific		
An adiabatic flash tank/mixer	Gas Liquid Separation		
A flash tank/mixer with given vapour-to-feed frac	Gas Liquid Separation		
Flotation shell	Separating		
Flotation shell 2	Separating		
Flash dryer	drying specific		
Former	p&p specific		
Lime kiln			
Flow controller	Controlling		
Gas compressor	Compression		
Gas IR-dryer	drying specific		
Gas turbine	Expanding		
Grinder	p&p specific		
Headbox			
Heater/cooler, define thermal duty	Enthalpy Change	Heating	Cooling
Heater/cooler, reverse calculated flow	Enthalpy Change	Heating	Cooling
Heat source	Cooling		
Absorption heat pump	Enthalpy Change	Heating	Cooling
Counter-current heat exchanger with no phase chan	Enthalpy Change	Heating	Cooling
Impingement dryer#2	p&p specific		
Impingement dryer			
Source unit (can be set by destination unit)			
Isothermic reactor	Reaction		
Washing filter	p&p specific		
Batch digester			
Lime kiln			
Measurement point	Measuring		
Membrane unit	Separating		
Mixer with phase equilibrium calculation			
Mixing chest with sweetener stock	p&p specific		
Storage tank with several inflows and outflows	Tank Storage		
Mix and fix outlet massflow	Controlling		

Table 5 (Continued). PLM RDL mapping to Balas modules along with proposed additions

	Mapped PLM process		
Balas module	**Main**	**Secondary 1**	**Secondary 2**
Mix and fix outlet temperature	Controlling		
COD and HHV monitor	Measuring		
Black-liquor evaporator	Heating		
Multistage screen	Separating		
OptiDry Vertical drying unit	p&p specific		
Electrostatic precipitator	Separating		
Press	p&p specific		
Pump/compressor	Entropy change	Pumping	Compression
Pulper	p&p specific		
Reactor	Reaction		
Recovery boiler	p&p specific		
TMP-refiner	p&p specific		
Repulping drum	p&p specific		
Reduction valve	Reducing		
Reduction valve with flow set	Reducing		
Steam box	Heating		
Scrubber	Enthalpy Change	Heating	Cooling
Steam heater	Heating		
Steam heated cyl. and vac. roll drying unit	p&p specific		
Tank where lime reacts with H2O to produce Ca(OH)	p&p specific		
Steam consumer	Controlling		
Smelt dissolver	p&p specific		
Separation with accept ratio- and K-values	Separating		
Source unit			
Combined mixer and divider	Bifurcating		
Split massflow	Bifurcating		
Steam turbine	Expanding		
Steam turbine with outlet pressure and temperatur	Expanding		
Direct heating with steam	Heating		
Sub-process terminal module			
Sub-process module			
Super batch digester	p&p specific		
Supplementary firing	power production specific		
Storage tank	Tank Storage		
Screening module with thickening ratio	Separating		
Screening module with reject ratios	Separating		
Simple separation module	Separating		
Heater/cooler, define outlet temperature	Enthalpy change	Heating	Cooling
Wire pit	p&p specific		

9.6 Balas to DEXPI Process

In the Balas to DEXPI Process mapping we found that 46 processes out of the 104 DEXPI Process entries were usable for Balas calculation module mapping. With these 46 processes, 91 calculation modules were mapped to at least Level 1 DEXPI Process entries; additionally, 65 calculation modules were mapped further to Level 2, and 31 to Level 3 processes.

DEXPI Process has more specific descriptions of processes than PLM RDL. This allows more exact pairing of the calculation modules to the processes. Difficulties arise when a particular process in the Balas model is modeled by a combination of calculation modules. For example, distillation is modeled with four calculation modules, but in DEXPI Process, distilling is a single process. At the same time, these four modules are already mapped to their specific processes (Figure 13). Again, it is unclear how well the data exchange will function between different tools through DEXPI Process during a project lifecycle.

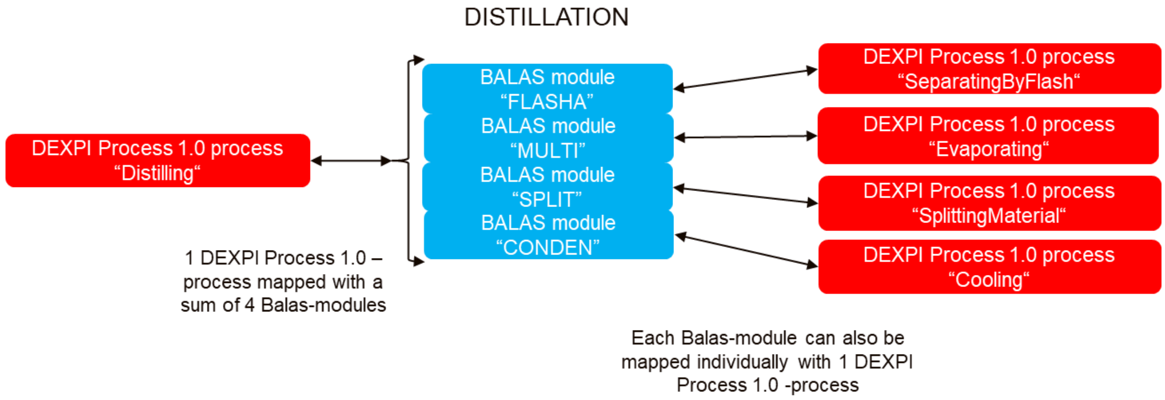

Figure 13. An example case: Distillation modelling versus Balas to DEXPI Process mapping.

It should also be noted that a calculation module might need to be mapped to multiple DEXPI Process entries. One such case is the boiler calculation module, which has combined properties of ReactingChemicals, SupplyingThermalEnergy, and Emitting processes in DEXPI Process. Similarly, other power production specific calculation modules could also be mapped by a combination of processes in DEXPI Process. This mapping can be seen in Table 5. The drying calculation modules can be mapped to the Drying process in DEXPI Process. Thus, new class for drying is not needed – as is the case in PLM RDL. For pulp and paper specific calculation modules, DEXPI Process does not have fitting processes either; these should be added to the standard. The full list of calculation module mappings to DEXPI Process processes with pulp and paper specific modules are given in Table 6.

Table 6. Balas calculation module mappings to processes of DEXPI Process specification.

Balas module	DEXPI PROCESS, Level 1	DEXPI PROCESS, Level 2	DEXPI PROCESS, Level 3
Aqua heat recovery	ExchangingThermalEnergy		
Air flotation dryer	Separating	SeparatingByThermalProcess	Drying
Approach temperature air heater	SupplyingThermalEnergy	EUROPEAN	
Approach temperature boiler surface	ExchangingThermalEnergy	PARTNERSHIP	
Approach temperature heat exchanger	ExchangingThermalEnergy		
Aeration lagoon with reactions	ReactingChemicals, Emitting		
Boiler	ReactingChemicals, SupplyingThermalEnergy, Emitting		
Burner	ReactingChemicals, SupplyingThermalEnergy, Emitting		
Separation with known filtrate composition	Separating	SeparatingMechanically	Filtering
Indirect heating with steam	SupplyingThermalEnergy		
Slaking/causticizing	p&p specific		
Caustification reactor	p&p specific		
Recovery boiler	p&p specific		
Cooling tower with two fillings	RemovingThermalEnergy	Cooling	
Condensor	RemovingThermalEnergy	Cooling	
Condenser for producing vacuum	RemovingThermalEnergy	Cooling	
Consistency controller	Mixing	Humidifying	
Cooling tower	RemovingThermalEnergy	Cooling	
Cooling tower with fan	RemovingThermalEnergy	Cooling	
Counter-current heat exchanger with phase change	ExchangingThermalEnergy		
Cooling tower with volumetric heat transfer coeff	RemovingThermalEnergy	Cooling	
Centrifugal cleaning	Separating	SeparatingByPhaseSeparation	SeparatingByCyclonicMotion
Centrifugal cleaning, parameter inlet massflow	Separating	SeparatingByPhaseSeparation	SeparatingByCyclonicMotion
Cylinder dryer	Separating	SeparatingByThermalProcess	Drying
Dual consistency splitting module	Separating	SeparatingByPhysicalProcess	
Dry debarking drum	Separating, ReducingParticleSize	SeparatingByPhysicalProcess, Crushing	
Deculator	Separating	SeparatingByPhaseSeparation	
Desuperheater	RemovingThermalEnergy, Mixing	Cooling	
Dewatering element	Separating	SeparatingByPhysicalProcess	
Disk filter	Separating	SeparatingMechanically	Filtering
Saveall disc filter	Separating	SeparatingMechanically	Filtering
Saveall disc filter with shower waters	Separating	SeparatingMechanically	Filtering
Discfsorp	Separating	SeparatingMechanically	Filtering
Smelt dissolver	p&p specific		
Divider	Separating		
Stream divider	Splitting	SplittingMaterial	
Steam heated cyl. drying unit	Separating	SeparatingByThermalProcess	Drying
Dry content controller	Mixing	Humidifying	
Split a stream into two outlet streams	Splitting	SplittingMaterial	
Electric IR-dryer	Separating, SupplyingThermalEnergy	SeparatingByThermalProcess, HeatingElectrical	Drying
Electric generator	SupplyingElectricalEnergy		
Electrostatic precipitator	Separating	SeparatingByElectromagneticForce	SeparatingByElectrostaticForce
Evaporator	Separating	SeparatingByThermalProcess	Evaporating
Evaporator 2	Separating	SeparatingByThermalProcess	Evaporating
Counter-current hex with flow calculation on one	RemovingThermalEnergy, SupplyingThermalEnergy, ExchangingThermalEnergy	Cooling	
Saveall filtrate tank	Storing, Separating	StoringFluids, SeparatingMechanically	StoringInTank, Filtering
An adiabatic flash tank/mixer	Separating	SeparatingByFlash	
A flash tank/mixer with given vapour-to-feed frac	Separating	SeparatingByFlash	
Flotation shell	Separating	SeparatingByPhaseSeparation	SeparatingByGravity
Flotation shell 2	Separating	SeparatingByPhaseSeparation	SeparatingByGravity
Flash dryer	Separating	SeparatingByThermalProcess	Drying
Former	p&p specific		
Lime kiln	p&p specific		
Flow controller	SteeringFlow	FeedingMaterial, LimitingFlow, RegulatingFlow, ShuttingOffFlow	
Set solids flow	SteeringFlow	RegulatingFlow	
Set total flow	SteeringFlow	RegulatingFlow	
Feed water tank	Storing, Separating	StoringFluids, SeparatingByThermalProcess	StoringInTank
Gas compressor	GeneratingFlow	Compressing	
Gas IR-dryer	Separating	SeparatingByThermalProcess	Drying

Table 6 (Continued). Balas calculation module mappings to processes of DEXPI Process specification.

Gas turbine	SupplyingMechanicalEnergy	DrivingByTurbine	
Grinder	p&p specific		
Headbox			
Heater/cooler, define thermal duty	RemovingThermalEnergy, SupplyingThermalEnergy	Cooling	
Heater/cooler, reverse calculated flow	RemovingThermalEnergy, SupplyingThermalEnergy	Cooling	
Heat source	SupplyingThermalEnergy		
Absorption heat pump	ExchangingThermalEnergy		
Counter-current heat exchanger with no phase chan	ExchangingThermalEnergy		
Impingement dryer#2	Separating	SeparatingByThermalProcess	Drying
Impingement dryer	Separating	SeparatingByThermalProcess	Drying
Source unit (can be set by destination unit)	Source, SupplyFluids, SupplyingSolids		
Isothermic reactor	ReactingChemicals		
Washing filter	p&p specific		
Batch digester			
Lime kiln			
Measurement point			
Membrane unit	Separating	SeparatingMechanically	Filtering
Mixer with phase equilibrium calculation	Mixing		
Mixing chest with sweetener stock	p&p specific		
Storage tank with several inflows and outflows	Storing	StoringFluids	StoringInTank
Mix and fix outlet massflow	Mixing		
Mix and fix outlet temperature	Mixing		
COD and HHV monitor			
Black-liquor evaporator	Separating	SeparatingByThermalProcess	Evaporating
Multistage screen	Separating	SeparatingMechanically	Sieving
OptiDry Vertical drying unit	Separating	SeparatingByThermalProcess	Drying
Electrostatic precipitator	Separating	SeparatingByElectromagneticForce	SeparatingByElectrostaticForce
Press	Separating	SeparatingByPhysicalProcess	
Pump/compressor	GeneratingFlow	Compressing, Pumping	
Pulper	p&p specific		
Reactor	ReactingChemicals		
TMP-refiner	p&p specific		
Repulping drum			
Reduction valve	SteeringFlow	RelievingOverpressure	
Reduction valve with flow set	SteeringFlow	RelievingOverpressure, RegulatingFlow	
Scrubber	ExchangingThermalEnergy		
Steam heater	SupplyingThermalEnergy		
Steam heated cyl. and vac. roll drying unit	Separating	SeparatingByThermalProcess	Drying
Tank where lime reacts with H2O to produce Ca(OH)	p&p specific		
Steam consumer			
Smelt dissolver	p&p specific		
Separation with accept ratio- and K-values	Separating		
Source unit	Source, SupplyFluids, SupplyingSolids		
Combined mixer and divider	Splitting	SplittingMaterial	
Split massflow	Splitting	SplittingMaterial	
Steam box	p&p specific		
Steam turbine	SupplyingMechanicalEnergy	DrivingByTurbine	
Steam turbine with outlet pressure and temperatur	SupplyingMechanicalEnergy	DrivingByTurbine	
Direct heating with steam	SupplyingThermalEnergy		
Sub-process terminal module	Sink		
Sub-process module			
Super batch digester	Separating		
Supplementary firing	SupplyingThermalEnergy, ReactingChemicals		
Storage tank	Mixing, Emitting		
Screening module with thickening ratio	Separating	SeparatingMechanically	Sieving
Screening module with reject ratios	Separating	SeparatingMechanically	Sieving
Simple separation module	Separating	SeparatingMechanically	Sieving
Heater/cooler, define outlet temperature	RemovingThermalEnergy, SupplyingThermalEnergy	Cooling	
Wire pit	Separating		

10 Modeling and knowledge representation languages

The Arrowhead fPVN deliverable D3.1 Section 9[1] has already explored the modeling and knowledge representation languages. In this section we would like to revisit two aspects: the implications of open-world vs. closed-words assumptions, and the role of the SysML modelling language in the context of Ontology Based Interoperability and Industrial Data Ontologies.

10.1 Open-world and closed-world assumptions

The open-world assumption and closed-world assumption represent two different ways of handling missing information when making logical conclusions. Under the open-world assumption, if something cannot be proven true, then we don't know whether it's true or false. This approach treats missing information as genuinely unknown. Under the closed-world assumption, if something cannot be proven true, then it is assumed false. The world is "closed" because the current knowledge is treated as complete. This approach treats missing information as evidence of absence.

These assumptions lead to very different reasoning outcomes. Open-world reasoning is more cautious and acknowledges uncertainty, which makes it suitable for situations where information might be incomplete or distributed across multiple sources. The semantic web and many artificial intelligence systems use this approach. Closed-world reasoning is more decisive and efficient, making it practical for databases and systems where one can reasonably assume the information is complete. The choice between these assumptions fundamentally changes how a system interprets gaps in knowledge and affects every conclusion it draws from incomplete information.

The IDO Guidelines rely on OWL 2 Direct Semantics and ontology consistency principles. These operate under open-world assumptions. During our expert discussions we explored the implications of open-world reasoning.

Property constraints, SHACL validation logic, and handling missing data require decisions that would be different under closed-world assumptions. For instance, when certain properties should be mandatory versus optional based on context, or when validating incomplete data from XML sources, we need to make design choices about how to handle the open-world assumption that missing information cannot be assumed to be false.

This gap between theoretical foundations and practical implementation lead us to focus our debates on solving immediate engineering problems rather than examining the logical underpinnings of their semantic framework. We emphasize creating useable patterns and tools for industrial engineers. Understanding open versus closed-world assumptions helps clarify many of the validation and constraint modeling challenges encountered in the engineering work.

10.2 SysML

Modern engineering domains – particularly those involving large-scale, service-oriented Systems of Systems (SoS) – face a fundamental tension between the dynamic flexibility of distributed architectures and the structured rigor demanded by safety, security, and lifecycle traceability requirements. This challenge is especially prominent in domains such as aerospace, manufacturing, and process industries, where complexity is paired with strict verification, validation, and compliance needs.

To address these needs, engineering governance has emerged as a cross-cutting meta-discipline that aims to coordinate and align heterogeneous models, tools, and processes across organizational and technological boundaries. Governance, in this context, encompasses the identification, formalization, and operationalization of the structures and relationships that make engineering workflows coherent, interpretable, and auditable. It is not merely about oversight – it is about enabling traceable and evolvable engineering across abstraction layers, domains, and system lifecycles.

These innovation activities are backed by and intertwined with a new generation of languages and standards emerging in various segments of the engineering universe. Not only IDO, but also SysML v2 fit into this meta-pattern.

The Systems Modeling Language (SysML) is a general-purpose modeling language tailored for systems engineering applications. It extends a subset of the Unified Modeling Language (UML) to support the specification, analysis, design, verification, and validation of complex systems. SysML (in its currently widespread version, here simply called v1) is an essentially diagrammatic language, addressing system architecture, behavioral models and parametrization of complex systems via visual modeling – which is, however, often used in its plainest form, as a mode of intuitive, semi-rigorous documentation. In the present vision we point out how to move forward from this stage in industrial practice.

Recognizing the limitations of SysML v1, the Object Management Group (OMG) initiated the development of SysML v2 to enhance precision, expressiveness, and usability. SysML v2 offers textual notation and standardized APIs in addition to graphical representations, largely contributing to next-level integration with various engineering tools and formats and, thus, promoting interoperability.

In the following, we demonstrate the potentials of SysML v2 – IDO co-modeling in a simplified, yet realistic fPVN scenario: we use a deliberately simple example of a thermal control component used in the process industry, aligned with the aims of the DEXPI initiative. First, consider a SysML v2 snippet:

```
package thermal.control {

    import dexpi::fluid::connections

    part def ThermometerSensor {
        // The physical sensing element
        part probe: TemperatureProbe
        // Expose the temperature as a measurement via a port
        port temperatureOut: out data port {
            feature temp: Real
        }
        // Define internal connection from probe to output port
        connect probe.temperature to temperatureOut.temp
    }
```

```
    part def TemperatureProbe {
        // Internal signal representing sensed value
        feature temperature: Real
    }
}
```

The scenario involves an information flow where the engineering data is being more refined and detailed as we move from high-level SysML v2 models to knowledge graph representations, where IDO provides a solid technical ground to formulate, e.g., the details on how and in what format thermal measurements are taken, as shown in the following IDO specification snippet in OWL format:

```
Prefix: lis: <http://rds.posccaesar.org/ontology/lis14/rdl/>
Prefix: ex: <http://example.org/thermo#>
Ontology: <http://example.org/thermo#>

# Define the sensor as a physical artefact
Class: ex:ThermometerSensor
    SubClassOf: lis:PhysicalArtefact,
                lis:hasPart some ex:TemperatureProbe,
                lis:hasFunction some ex:TemperatureSensing
                Function

# Define the internal sensing component
Class: ex:TemperatureProbe
    SubClassOf: lis:PhysicalArtefact,
                lis:hasFunction some ex:TemperatureSensing
                Function

# Define the function performed by the probe
Class: ex:TemperatureSensingFunction
    SubClassOf: lis:Function

# Define the output data representation
Individual: ex:SensorReading001
    Types: lis:ScalarQuantityDatum
    Facts: lis:datumValue "22.5"^^xsd:float,
           lis:datumUOM ex:degreeCelsius,
           lis:quantifiesQuality ex:TemperatureQuality001

# Define the temperature quality being measured
Individual: ex:TemperatureQuality001
    Types: lis:PhysicalQuantity
    Facts: lis:qualityOf ex:ThermometerSensor

# Define the unit (referencing OM or QUDT style, or IDO-defined)
Individual: ex:degreeCelsius
    Types: lis:UnitOfMeasure
    Annotations: rdfs:label "degree Celsius"

# Optional link from sensor to output data
Individual: ex:ThermometerSensor01
    Types: ex:ThermometerSensor
    Facts: lis:hasQuality ex:TemperatureQuality001
```

Particular attention is to be paid to the second half of the specification that, while remaining aligned with the SysML model, introduces the output representation and the measurement unit (degree Celsius).

As a result of fPVN, we envision a holistic governance framework, pillared on the knowledge models demonstrated above, digital twins of Arrowhead systems of systems, towards satisfying complex, inter-domain requirements, as summarized in Figure 14.

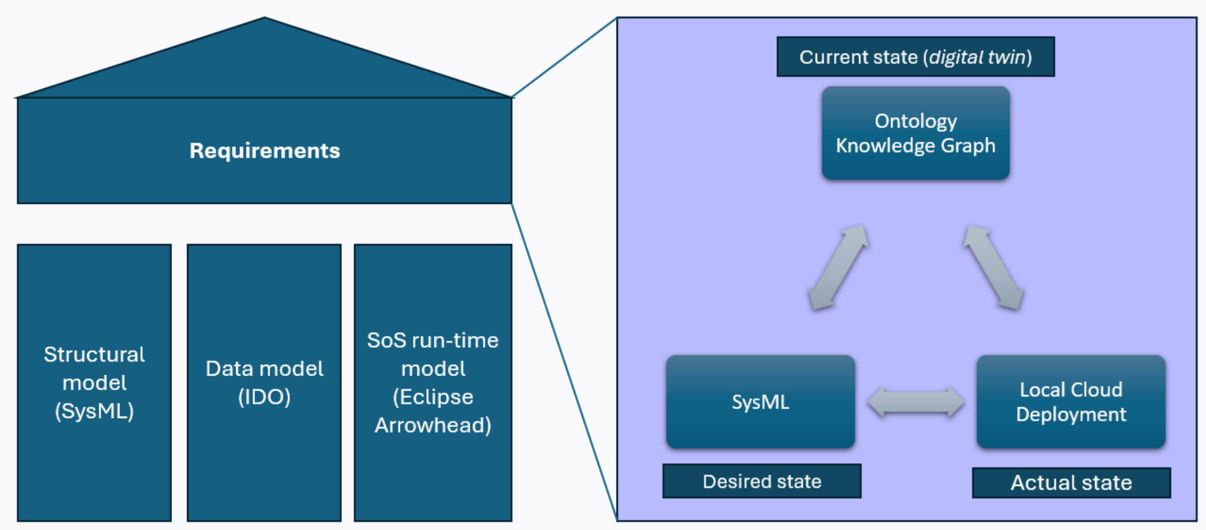

Figure 14. Inter-domain requirements within the Arrowhead fPVN systems of systems

11 A move towards open-sourcing ontology work

Based on our technical discussions involving industry experts, there are clear indications that major industrial players are considering moves toward open-sourcing their ontology work. This represents a shift in how industrial data modeling intellectual property is handled. The motivation behind this shift is multifaceted. For one, the standardized approaches to data modeling would benefit everyone in the ecosystem. When organizations work in isolation, they often duplicate efforts and create incompatible systems that make interoperability difficult. By open-sourcing ontology work, companies can use each other's expertise and avoid reinventing solutions that already exist.

The collaborative approach is also valuable in the context of industrial standards alignment. Currently there are challenges in harmonizing different standards like ISO 15926, DEXPI, CFIHOS, and others. For instance, definitional differences between IEC 61987 and ISO 15926-4 results in only 6% exact match between instrumentation classes. This creates barriers to interoperability. When companies share their ontology work openly, these alignment challenges become easier to address.

The technical benefits of this open approach are substantial. For instance, Siemens Energy's tool for property modeling could serve as a foundation for developing broader IDO editors that the entire community could use. Rather than each organization developing its own proprietary tools and methods, shared resources would allow smaller companies and research institutions to build upon proven foundations developed by industry leaders.

The initiative also has strategic advantages for the companies themselves. By contributing to open standards and sharing their work, large organizations can influence the direction of industry-wide standards development. They become key stakeholders in shaping how data interoperability evolves, rather than being forced to adapt to standards developed by others. This positioning can be more valuable than simply protecting proprietary methods that might become obsolete if the industry moves in different directions.

The open-source approach also enables faster innovation cycles. When multiple organizations contribute to the same foundational tools and ontologies, improvements and bug fixes happen more rapidly than when development is confined to internal teams. Companies benefit from community testing, feedback, and enhancement of their original contributions, often receiving more value back than they initially invested.

Our discussions revealed that this trend toward openness is happening across multiple organizations simultaneously. The collaboration between Arrowhead fPVN, DISC, and CFIHOS Semantics demonstrates that industry consortiums are actively working to align their efforts rather than compete in parallel. This coordination suggests that the movement toward open ontology sharing is gaining momentum across the industrial automation sector.

The practical implementation of this open-source approach requires careful consideration of intellectual property boundaries. Companies need to determine which aspects of their ontology work can be shared without compromising competitive advantages in other areas. Our discussions suggest that foundational data modeling patterns and interface specifications are good candidates for open sharing, while specific application implementations might remain proprietary.

The shift toward collaborative ontology development represents a recognition that data interoperability challenges are too complex and important for any single organization to solve effectively in isolation. By pooling resources and expertise, the industrial community can develop more robust, comprehensive solutions that benefit everyone involved while accelerating the overall pace of innovation in industrial digitalization.

12 Discussion

This section analyzes the IDO’s application for interoperability across industrial data standards. This work emerged from the need to establish consistent modeling patterns that enable seamless data exchange between industrial systems, organizations, and standards.

The IDO's abstract and philosophical foundation (while offering flexibility) presents challenges to ensure consistent implementation across applications and domains. This flexibility can lead to varying interpretations and implementations that may not be automatically interoperable, despite being based on the same underlying ontology. This challenge is addressed by examining the concept of "levels of expressiveness" within IDO-based ontologies. These levels range from highly detailed, semantically explicit representations to simplified shortcuts that enable direct mapping from database structures to ontological representations.

An important contribution of this analysis is the relationship between IDO and established industrial standards – particularly IEC 61360-1:2017 –, which defines the fundamental concepts for industrial data dictionaries. The following sections explore how IDO's language elements can bridge industrial standards – potentially reducing the engineering effort required to map properties and concepts across multiple standards.

This section also examines modeling patterns for representing properties, classes, and their relationships within IDO-aligned ontologies. These patterns address real-world engineering challenges such as defining mandatory versus optional properties, handling historical data, managing cardinality constraints, and ensuring proper data quality governance throughout the lifecycle of industrial assets.

Throughout this analysis, three key aspects are consistently evaluated: the available language elements and their capabilities, the knowledge and data governance policies needed to ensure consistent application, and the infrastructure requirements for knowledge-driven data integration systems that can support engineers and domain experts in creating and maintaining interoperable industrial ontologies.

The ultimate goal of this work is to establish a foundation for widespread adoption of IDO-based approaches in industrial settings. This will enable more efficient and reliable data exchange across the entire value chain from customers to solution providers to suppliers, and supporting the vision of truly interoperable industrial data ecosystems.

12.1 List of patterns for IDO aligned ontologies

IDO as upper ontology offers different possibilities to model the same case (one example are the different levels of expressiveness described in the next chapter). To achieve interoperability modeling principles and patters should be defined.

Modeling patterns for ontologies that are intended to be interoperable are important for several key reasons:

- **Consistency Across Ontologies**: Interoperable ontologies often need to work together or integrate seamlessly. Modeling patterns provide standardized ways to represent concepts and relationships, reducing ambiguity and ensuring that similar concepts are modeled similarly across different ontologies.
- **Improved Reusability**: When modeling patterns are followed, ontology components become more modular and reusable. This means that elements of one ontology can be more easily reused in another, reducing duplication of effort and promoting knowledge sharing.
- **Enhanced Interoperability**: Interoperability is the ability of different systems and organizations to work together. Standard modeling patterns ensure that ontologies can communicate effectively, enabling data integration, sharing, and semantic alignment across diverse domains.
- **Facilitated Maintenance and Evolution**: Consistent modeling makes it easier to update and maintain ontologies over time. Changes in one part of an ontology are less likely to cause unintended effects elsewhere when standard patterns are used.
- **Better Tool Support**: Ontologies designed using standard modeling patterns can be more easily supported by automated tools for reasoning, validation, and querying. This improves their usability and functionality in practical applications.

- **Reduced Learning Time**: For developers and users of ontologies, familiar modeling patterns reduce the time and effort needed to understand and work with new ontologies. This supports broader adoption and collaboration.

Some of the cases for patterns for IDO aligned ontologies which should be discussed, if needed reworked, agreed and released to be used in the industry community are:

- Property modeling (details see next chapters)
    - as class (highest level of expressiveness)
    - as object property (shortcut with middle level of expressiveness)
    - as datatype property (shortcut with lowest level of expressiveness)
- IDO does not allow to assign properties (lis:Quality) directly to all of the classes defined in IDO. Therefore, patterns are needed for the different case to express something with properties. Here some examples:
    - Maximum temperature which should be achieved in a heating process at the Time series data (e.g., of the current temperature of the water at the water out port of a boiler)
- Data quality rules for the property assignment to a class (independent from the level of expressiveness)
    - Is the property mandatory or optional maybe dependent on the lifecycle status of the class it is assigned to?
    - Is it allowed or needed to have more than one value? (E.g., Can a Person have more than one given name?)
- Definition of the expected datatype of the value of an instance of a property (details see Section 12.10 “Data types which could be derived from IEC 61360-1”).
- A property can have different characteristics that should only be used in certain context. For example, the property Voltage can be used as Nominal Voltage in the Context of the specification or Operation and Rated Voltage in the context of a product. These two properties can have a rule like Rated Voltage should be higher than Nominal Voltage to select products which fulfill the specification (or operational requirements). Unit and quantity kind modeling (unit without quantity kind is incomplete, e.g., the unit Volt is used for the quantity kind Voltage and Electrical Potential).
- Modeling of lists of values
- Use of lists of values in different contexts like properties and classes (including inheritance hierarchy – which means that the valid values are reduced more because the classes are becoming increasingly specific.)
- How to model the assignment of different classifications to the same resource?
- Modeling of partonomies including restrictions on the cardinality (e.g., Power transformers and Reactors have windings, therefore they should inherit this from a parent class, but the Power transformer must have exactly two or more windings and the Reactor must have exactly 1 winding).
- Synonyms
- Identification keys
- Polymorphism as defined in IEC 61360-1.
- Modeling of dependent cardinalities (e.g., the property serial number becomes mandatory latest at the point in time the name plate of a product should be produced).

12.2 Levels of expressiveness

IDO offers an abstract and philosophical foundation where the main elements of the language are predefined in a general manner. How to use the elements defined in IDO is more on a general level described and leaves room for interpretation. This means that different IDO based ontologies are not automatically interoperable.

This can be illustrated with Figure 15. It is used explains the two shortcuts lis:qualityQuantifiedValue and lis:hasQualityQuantifiedAs based on lis: hasQuality, lis:Quality, lis:qualityQuantifiedAs, lis:QualityDatum and lis:UnitOfMeasure. Please refer to the legend at Figure 16.

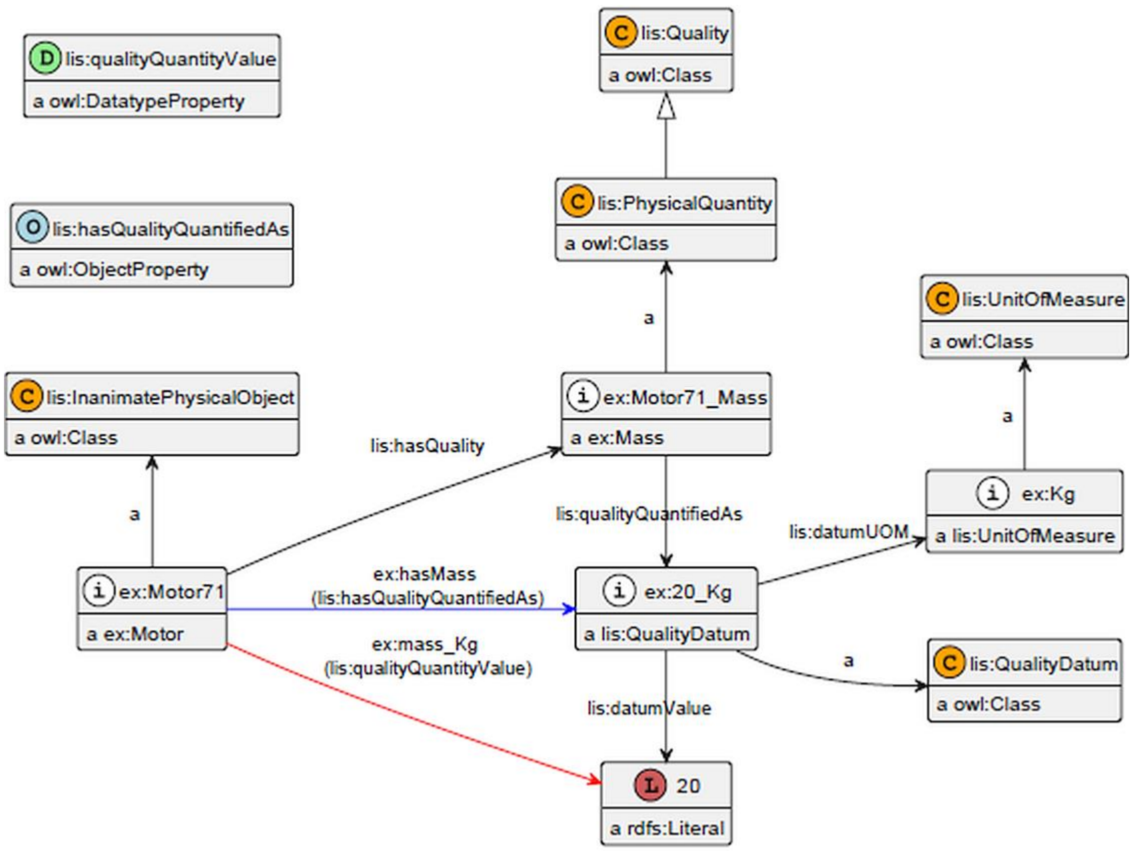

Figure 15: Representations of mass of a motor
Source: ISO/DIS 23726-3:2024 Industrial Data Ontology – DIS stage

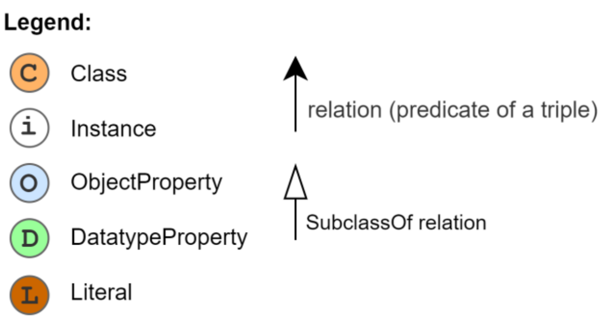

Figure 16. Legend for the representations

The prefix lis: is used as abbreviation of the IDO namespace, and it stands for "lifecycle information system"; This prefix was defined before the name Industrial Data Ontology (IDO) was defined.

The picture shows three patterns with different levels of expressiveness. Here expressiveness means to define knowledge as explicit as possible. The pattern with the highest expressiveness is shown in the following extract from Figure 17.

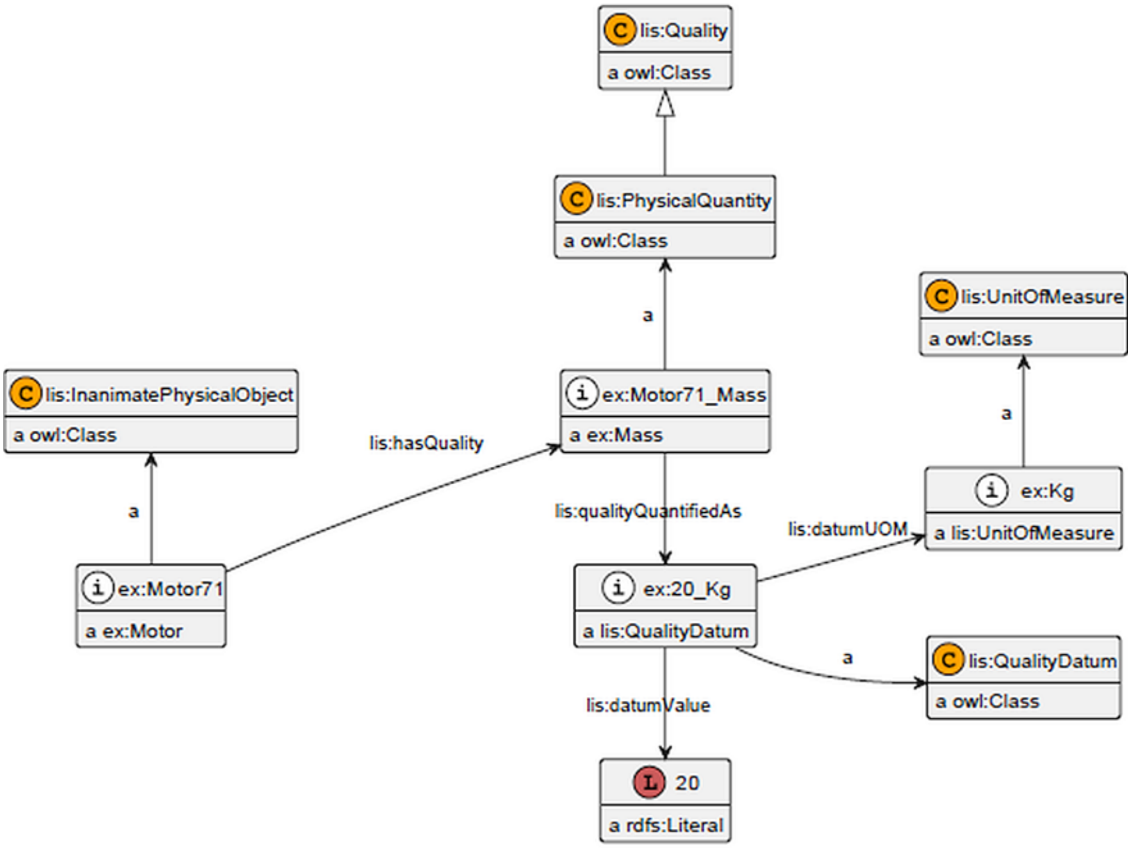

Figure 17. Representations of mass of a motor. Focus on the elements of the whole picture.

The pattern with less expressiveness is shown in the following extract of Figure 18.

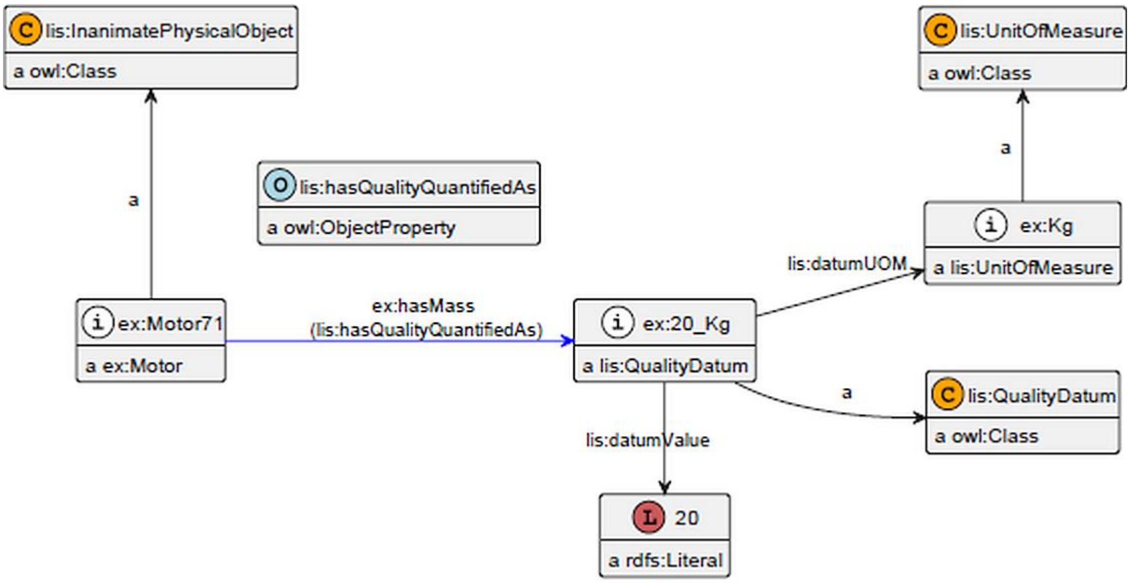

Figure 18. Representation of mass of a motor with focus on the lis:hasQualityQuantifiedAs short-cut

The pattern with the lowest expressiveness is shown in the next extract of Figure 19.

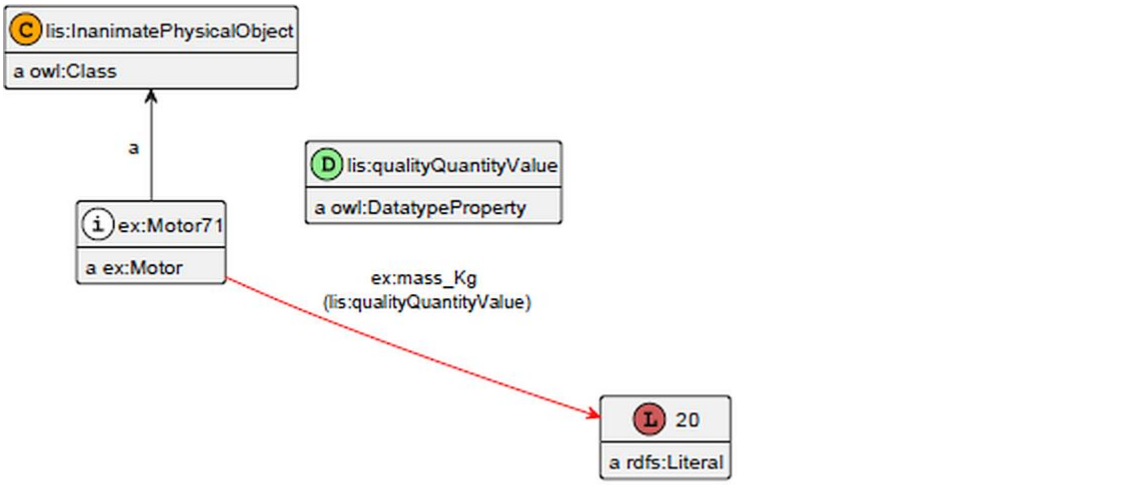

Figure 19. Representation of mass of a motor with focus on the lis:qualityQuantityValue short-cut

All three patterns are valid because they support different use cases:

- The pattern with the highest expressiveness supports the uses case to collect all the knowledge about a property (lis:Quality) in the most explicit manner to enable detailed querying for the different aspects of a property (e.g., like it is defined in IEC 6130-1 as well).
- The pattern with the lowest expressiveness support the uses case to map data to IDO based modeling directly (ex:Motor and ex:mass_kg could be columns in a database table and ex:Motor71 and 20 a dataset in this database table).
- The third pattern is something between the other both. This pattern is for example used for the RDF serialization of ECLASS (see "Technical Specification 110 - ECLASS Serialization as RDF - Part 1", chapter 6.6 "Property") or is used in QUDT (qudt:Quantity is a combination of lis:Quality and lis:QualityDatum; see Figure 20; more details about this are described later).

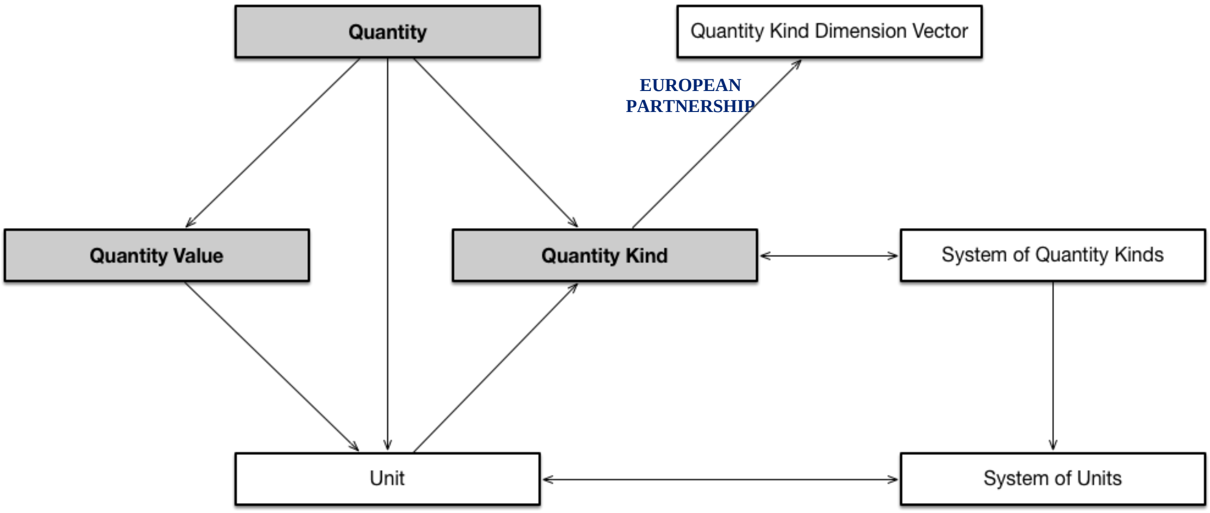

Figure 20. Main elements defined in QUDT (Source: https://www.qudt.org/)

Figure 21 shows different levels of expressiveness. These levels are not defined yet. One principle could be to annotate IDO based ontologies by a level of expressiveness. The highest level of expressiveness offers all information to automatically create the lower levels of expressiveness. This should be expressed as well maybe with a new AnnotationProperty "is derived from" like shown in the following picture.

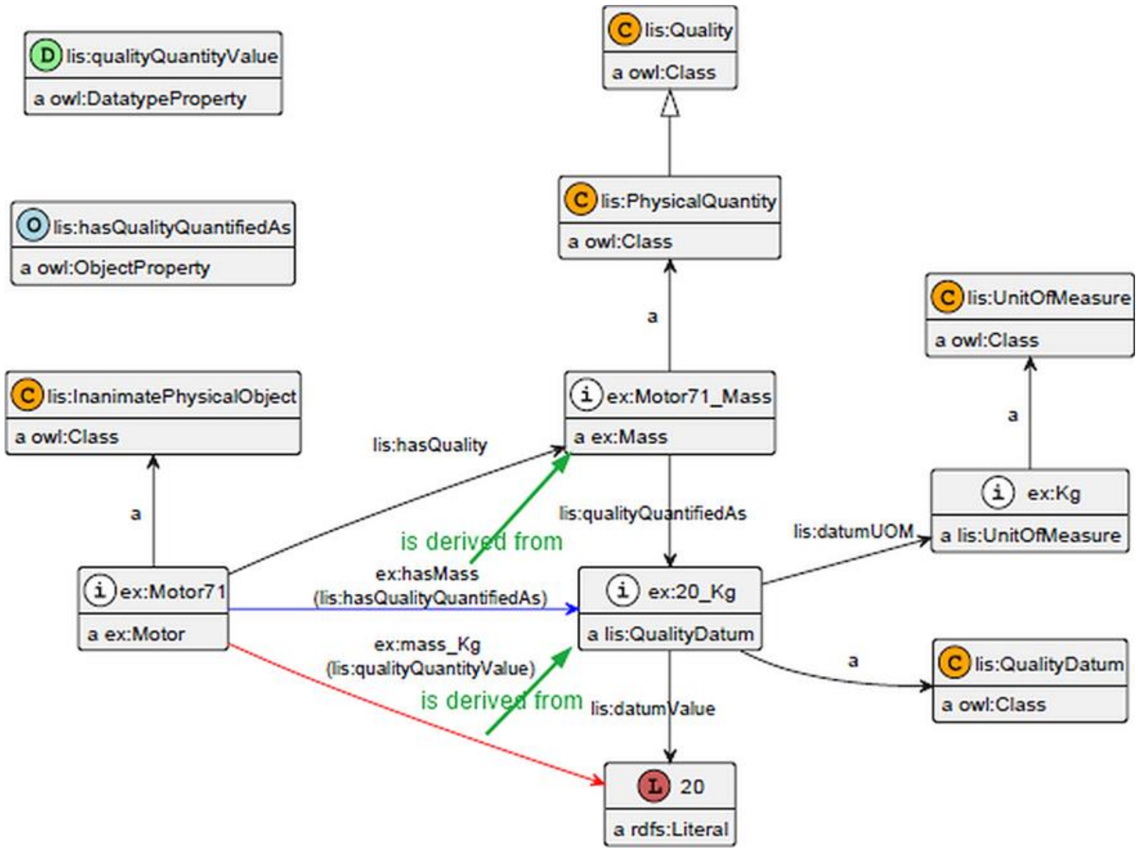

Figure 21. Representation of mass of a motor, extended with the "is derived from" relation

Table 7 offers a summary of the different aspects of expressivity.

Table 7. Summary of the levels of expressivity

Aspect	Available	To be discussed and introduced
Language	Basic elements for three levels of expressivity	• Levels of expressivity are not yet defined • Modelling elements which define how lower levels of expressivity can be derived automatically from the higher level
Knowledge/Data Governance Policies	Use cases for each level	• Guidelines which describe the levels of expressivity including the criteria which are relevant to identify the level of expressivity of an ontology • Modeling patterns for the different levels of expressivity • Rules on how to derive the lower levels of expressivity from the higher levels
Knowledge-/ Driven Data Integration Systems	–	• Tools which are implementing the guidelines, modeling patterns and rules by collection the relevant information from the user and creating the triples automatically in line with the guidelines based on the input of the user. • Vision: AI could support extracting the knowledge from data and to created already a first draft of the different levels of expressivity.

12.3 Example for a Centrifugal Pump Tag

Several drafts for patterns are mentioned in the document. How CFIHOS, STEP and DEXPI could communicate on ontology level is shown in this chapter by generalizing the model shown in Section 6.2 “CFIHOS Semantics to IDO” and combine it with the Process Plant part mentioned in Section 6.3 “DEXPI to IDO”. This is a first starting point for discussing the needed modeling patterns. In the next step this example should be completed and the modeling patterns used should be documented for reuse.
The following principles/patterns are part of the example:

- Separation between general ontologies and specific views on the general ontologies
- Inheritance hierarchy: Hierarchy from a general level down to a specific level (Note that, it is not a classification hierarchy; a picture of an inheritance hierarchy used at Siemens Energy is shown at the end of this chapter)
- Different patterns to model properties
- Synonyms
- ClassIdentifiers

The example shows three different possibilities to model the same case. Figure 22 shows the model where patterns for high expressivity are used; Figure 23 shows patterns with lower expressivity; and Figure 24 shows patterns with even lower expressivity.

An example of an inheritance hierarchy developed at Siemens Energy is illustrated in Figure 25.

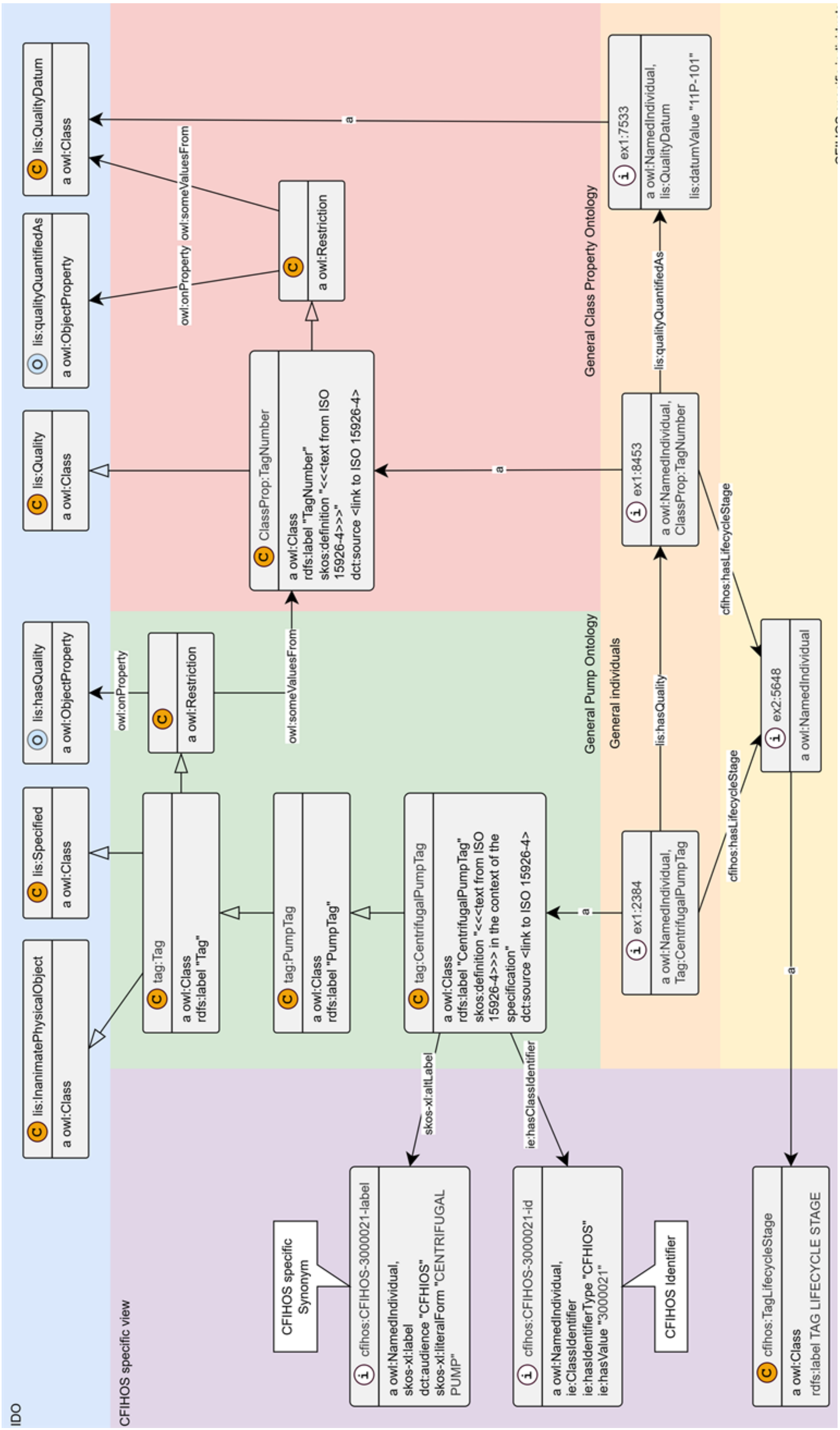

Figure 22. Model with patterns for high expressivity

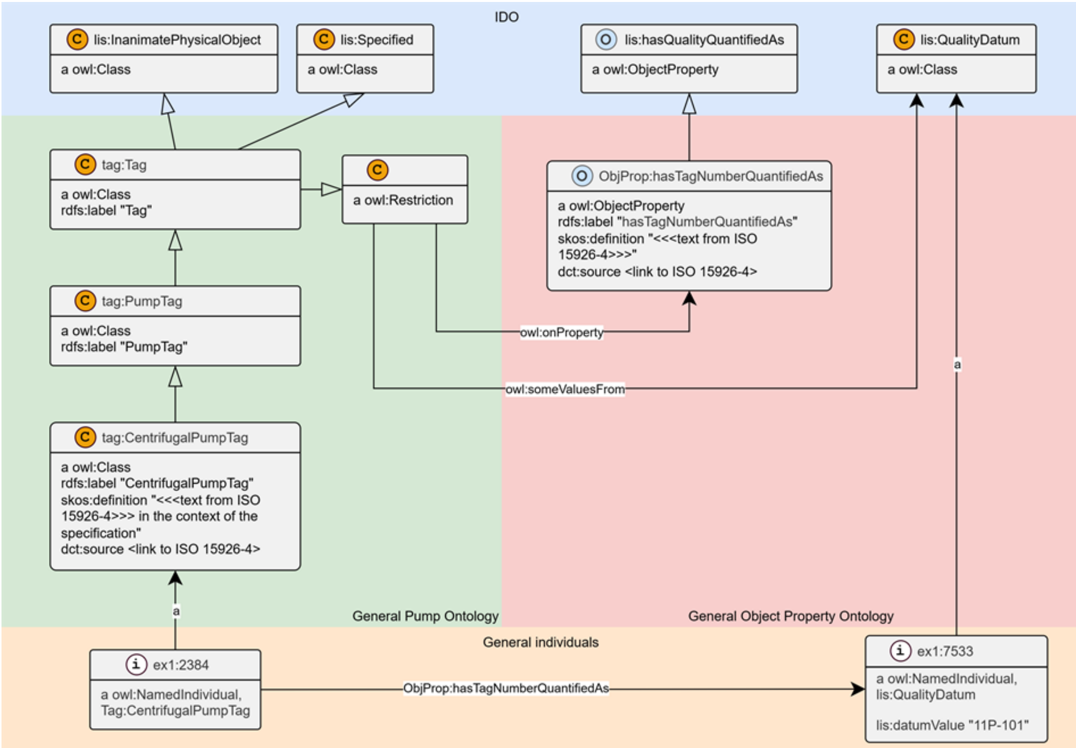

Figure 23. Model with patterns for medium expressivity

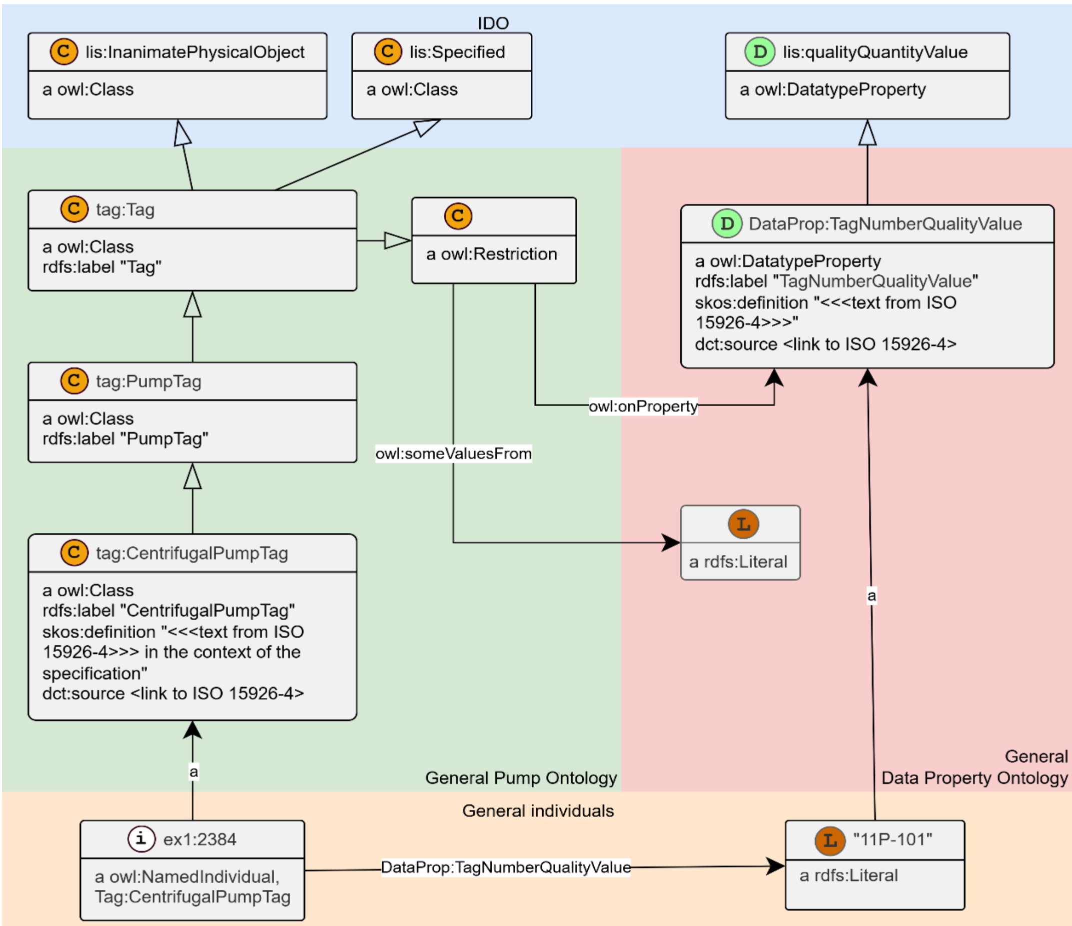

Figure 24. Model with patterns for lower expressivity

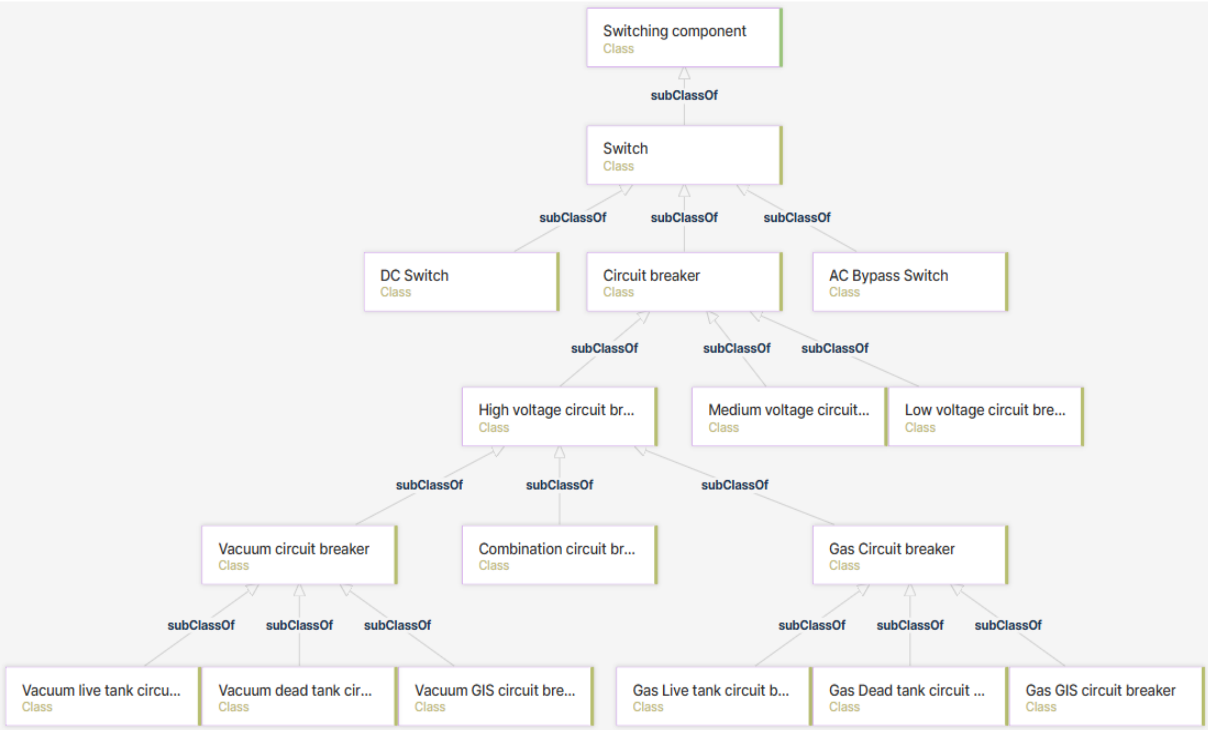

Figure 25. An example of an inheritance hierarchy developed at Siemens Energy

12.4 IDO provides a unifying language

We have seen different levels of expressiveness and realized that it is possible to automatically derive all other levels from the level with the highest expressiveness. We will now take a closer look at the highest level of expressiveness. This will be done from the perspective of engineering experts who have the knowledge which should be made human and machine readable and with this made interoperable.

Such engineering experts are already using existing standards like ECLASS or IEC CDD which are both based on IEC 61360-1:2017 (for now the focus is on these standards, other standards will be considered later). One task of such experts is to ensure that international standards are mapped to the data used in the company in order to enable trustworthy communication with internal and external partners who use the same standard. This includes the mapping between different standards.

Standards usually focus on a specific perspective in order to reduce complexity and focus. In the end, all perspectives belong together. Achieving this efficiently is a challenge for engineers. The thesis is that IDO has the essential language elements to depict information coherently from different perspectives. If this thesis is proven, then the IDO would be a candidate as the core of knowledge governance and could be the enabler for a seamless value-chain from the customer to the solution provider to the sub suppliers and the way back.

It is therefore of interest to develop initial ideas to answer the following questions:

- How does the language defined in IEC 61360-1:2017 match as a first example with the language defined in IDO?
- Which elements overlap?
- Which elements differ and how could they be merged to collect all information at one place?

Initial Ideas for answering these questions are developed step by step below. These ideas are intended to encourage the definition of uniform rules for the application of IDO, incorporating existing standards and with this enabling wide-ranging interoperability.

12.5 Classes and Properties as main language elements

12.5.1 Classes and Properties in IEC 61360-1:2017

To answer these questions step by step the main elements from IEC 61360-1:2017 will be compared with the one from IDO. The main elements defined in IEC 61360-1:2017 are:

- iec61360:Item_class,
- iec61360:Property,
- iec61360:Unit_of_Measure,
- iec61360:Value_list,
- iec61360:Value,
- iec61360:Relation
- and as root element iec61360:Dictionary_element

As shown in Figure 26 of IEC 61360-1:2017. The prefix iec61360: is not defined in IEC 61360-1:2027. It is introduced to clearly assign elements to the standards in which they are defined.

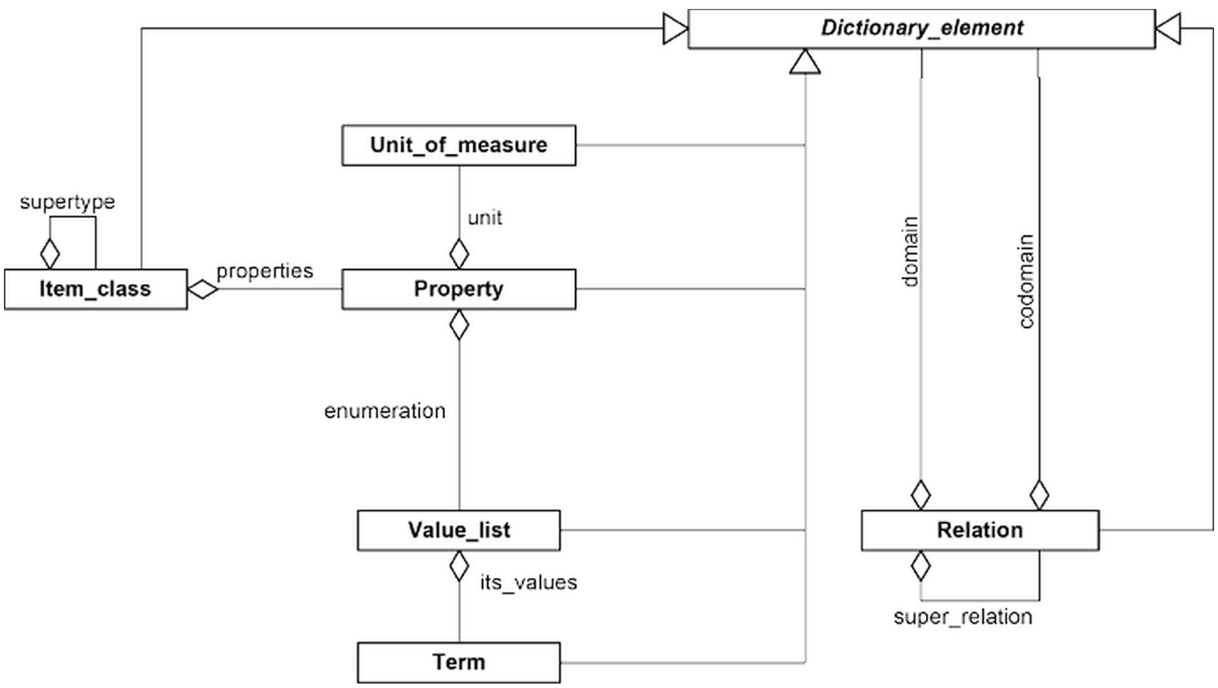

Figure 26. Main elements defined in IEC 61346-1

In addition, two general terms are introduced: Class and Property.

Class is used in this concept as it is defined in OWL Web Ontology Language Reference, Chapter 3. Classes: Classes provide an abstraction mechanism for grouping resources with similar characteristics. Like RDF classes, every OWL class is associated with a set of individuals, called the class extension. The individuals in the class extension are called the instances of the class. A class has an intentional meaning (the underlying concept) which is related but not equal to its class extension. Thus, two classes may have the same class extension, but still be different classes. The similar term of Class in IEC 61360-1:2017 would be entity (and not class): 3.1.12 entity: class of information defined by common properties class of information defined by common properties. (3.1.6 class: abstraction of a set of similar products.)

Property is used in this concept as it is defined in IEC 61360-1:2017, Chapter 6.2: information object that characterizes the Item_class(es) to which it is associated. A property, when used, always has an associated value, such as a numerical value together with a unit, or a qualitative value taken from a predefined list. It should be noted that Property is not used as generic term for owl:AnnotationProperty, owl:ObjectProperty or owl:DatatypeProperty. It is viewed from the perspective of an engineer who wants to use IDO. The term PROPERY as defined in IEC 613630-1:2017 is more common for engineers.

12.5.2 Classes and Properties in ISO/DIS 23726-3 IDO

The most important elements defined by IDO to be mapped to the Elements defined in IEC 61360-1:2017 are shown in the following list. The elements which are relevant for the mapping to IEC 61360-1:2017 are highlighted in **bold font**. The prefix lis: is for the elements defined in IDO. The indentations in the following list symbolize the parent-child relationship between the elements. The inverse owl:ObjectProperties are not listed.

- owl:Class
    - lis:Object
        - lis:InformationObject
            - lis:QualityDatum
            - lis:UnitOfMeasure
    - lis:Dependent
        - lis:Quality
- owl:ObjectProperty
    - lis:hasPart
        - lis:hasContentPart
            - lis:datumUOM
    - lis:hasQuality
    - lis:representedIn
        - lis:qualityQuantifiedAs
- owl:DatatypeProperty
    - lis:datumValue

Figure 27 shows the elements which are relevant for the mapping to IEC 61360-1:2017 graphically together with the parent elements.

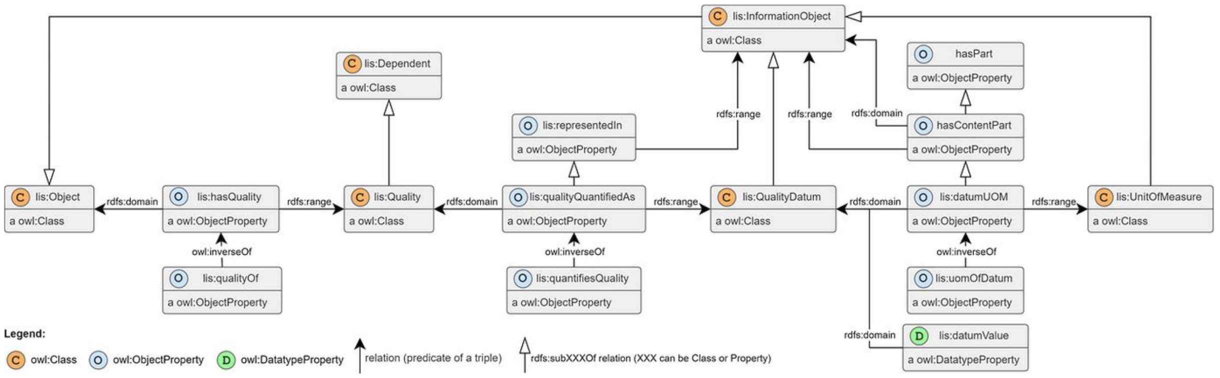

Figure 27. Model extraction of the elements from lis:Object to lis:UnitOfMeasure

The picture shows that IDO expresses the relationship between an object and its properties (here lis:Quality) with semantic precision. Expressed in words, the image shows the following:

- A lis:Object can have lis:Quality.
- lis:Quality is a subclass of lis:Dependent. lis:Dependent is defined as "An entity that is an inherent quality or realizable potential of an object". This means that a used lis:Quality always need a bearer which is lis:Object.
- The relationship between lis:Object and lis:Quality is defined as lis:hasQuality.
- The relationship between lis:Quality and lis:Object is defined as lis:qualityOf.
- Not shown in the picture, but in IDO defined, is that lis:Object and lis:Dependent are disjoint classes. This means that lis:Object and lis:Quality are also disjoint.
- A lis:Quality can have lis:QualityDatum.
- lis:QualityDatum is a subclass of lis:InformationObject which is a subclass of lis:Object.
- The relationship between lis:Quality and lis:QualityDatum is defined as lis:qualityQuantifiedAs.
- lis:qualityQuantifiedAs is a subPropertyOf lis:representedIn which is defined as relation pointing to a lis:InformationObject.
- The relationship between lis:QualityDatum and lis:Quality is defined as lis:quantifiesQuality.
- A lis:QualityDatum can have lis:UnitOfMeasureand lis:datumValue.
- lis:UnitOfMeasure is a subclass of lis:InformationObject which is a subclass of lis:Object.
- The relationship between lis:QualityDatum and lis:UnitOfMeasure is defined as lis:datumUOM.
- lis:datumUOM is a subPropertyOf lis:hasContentPart which is defined as relation between lis:InformationObject.
- lis: hasContentPart is a subPropertyOf lis:hasPart.
- The relationship between lis:UnitOfMeasure and lis:QualityDatum is defined as lis:uomOfDatum.

All this information is human and machine readable (which is the nature of the semantic web standards defined by Word Wide Web Consortium W3C where IDO is based on). Table 8 offers a mapping of the element defined in IDO and elements defined in IEC 61360-1:2017.

IDO is using the following in OWL defined elements as upper elements like iec61360:Dictionary_element in IEC 61360-1:2027:

- owl:Thing for owl:Classes
- owl:topObjectProperty for owl:ObjectProperties
- owl:topDataProperty for owl:DatatypeProperties

Table 9 shows an initial indicator for the thesis that IDO provides a unifying language together with the existing semantic web standards defined by W3C. It has been shown that there are differences in the precision of the elements defined in IDO and IEC 61360-1:2027. It therefore seems possible to use IDO as a higher-level language for integrating standards. If this were indeed the case, it would reduce the effort that engineers currently have to put into mapping Properties from different standards. It is therefore worth delving deeper into the details. The following table shows some points which are needed to set up a framework to enable Property mapping holistically. The core idea is: IDO makes it possible to create a place where information from different standards is brought together.

Table 8. Mapping between elements defined in IDO and IEC 61360-1:2017

Element in IDO	Element in IEC 61360-1	Comment
lis:Object	iec61360:Item_class	Deeper comparison will come later
lis:Quality	iec61360:Property	IDO is more precise than IEC 61360-1:2017
lis:QualityDatum	iec61360:Property	
lis:UnitOfMeasure	iec61360:Unit_of_Measure	similar
lis:hasQuality	iec61360:Relation	IDO is more precise than IEC 61360-1:2017
lis:qualityQuantifiedAs	iec61360:Relation	
lis:datumUOM	iec61360:Relation	
lis:datumValue	–	
–	iec61360:Value_list	Several concepts are defined in the W3C standards to define value lists. Therefore, no definition in IDO is needed
–	iec61360:Value	

Table 9. IDO property standards integration implementation aspects

Aspect	Available	To be discussed and introduced
Language	IDO: Basic elements to define Properties precisely. IEC 61360-1:2017: Additional elements for more details	EUROPEAN PARTNERSHIP • Investigation into whether other standards which are defining Properties can also be integrated. • Maybe missing modeling elements for the integration.
Knowledge/ Data Governance Policies	Use case to map Properties from different standards	• If it is not possible to convert existing standards to become IDO conform on the highest expressivity level, then agreements are needed who will add the missing information and how this will be done including a release workflow. • Guidelines that describe how IDO should be used to integrate existing standards which are defining Properties (maybe by defining extensions of IDO). • Guidelines that describe the principle how the origin format of the model defined in the standard can be generated out of the IDO conform model.
Knowledge-Driven Data Integration Systems	–	Infrastructure to support the integration of international Standards which are defining Properties. This should include: • editors which support the guidelines to collect the information from different standards and save the results at a common place, • workflow and versioning support, • generation of the format of origin standard out of the IDO conform model, • APIs to get the information about the Properties from different perspectives like: form the different expressiveness levels, from the perspective of a certain standard or from the quality aspect.

12.6 Property modeling defined in IDO

We have seen that a deep dive in IDO and other standards makes sense to find out how the integration of Properties could look like. We start with the IDO by looking deeper in the Figure B.1 mentioned already above. The following picture shows all elements from IDO and the elements defined in the Domain Ontology (prefix ex:) and Instance Data (Prefix ex: as well which should be discussed). The different parts of this picture will be described step by step.

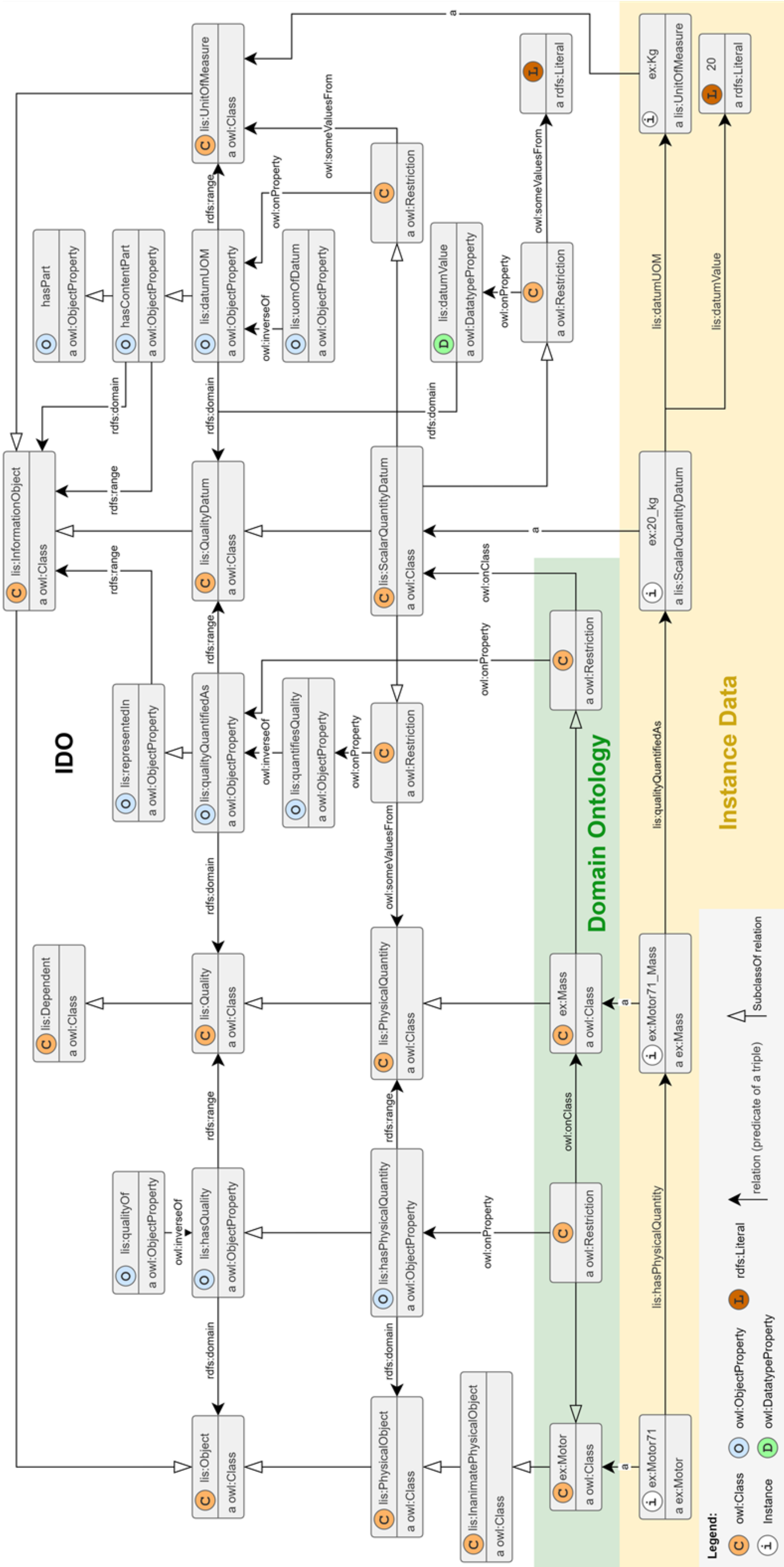

Figure 28. Model extraction of the elements from lis:PhysicalObject to lis:UnitOfMeasure including an example of how to use

12.6.1 Two alternatives to model Properties based on IDO

Figure 28 shows the following differences in addition to the fact that all elements are shown:

- the elements defined in the domain ontology are shown explicitly,
- lis:Quality is replaced by lis:hasPhysicalQuantity and
- lis:QualityDatum is replaced by lis:ScalarQuantityDatum.

IDO allows both modeling options. Although the same thing should be expressed, the two models are different in their precision and thus in their interoperability. This example alone shows that modeling rules are required to ensure that models can be interpreted consistently. Table 10 summarizes the findings of this section.

Table 10. Knowledge governance and data integration aspects for IDO Properties

Aspect	Available	To be discussed and introduced
Language	–	–
Knowledge/Data Governance Policies	Possibilities to model Properties based on IDO	Guidelines that define rules as to when which of the possible modeling patterns defined in IDO should be used.
Knowledge-Driven Data Integration Systems	–	Infrastructure to guide the users to select the suitable pattern in line with the guideline for creating and maintaining Properties based on IDO.

In the following, some suggestions for rules are worked out by going through the above picture from left to right in individual parts.

12.7 Relation between lis:Object and lis:Quality and their subclasses

The following picture focusses on the Relation between lis:Object and lis:Quality and its subclasses (left part of Figure 28). The green area shows what a Domain Ontology could look like:

- The Class ex:Motor is defined
- The Property ex:Mass is defined
- An owl:Restriction on the relation between the Class ex:Motor and the Property ex:Mass is defined (an alternative could have been a SHACL shape)

The corresponding instances (data represented in a knowledge graph) are shown in the yellow area .

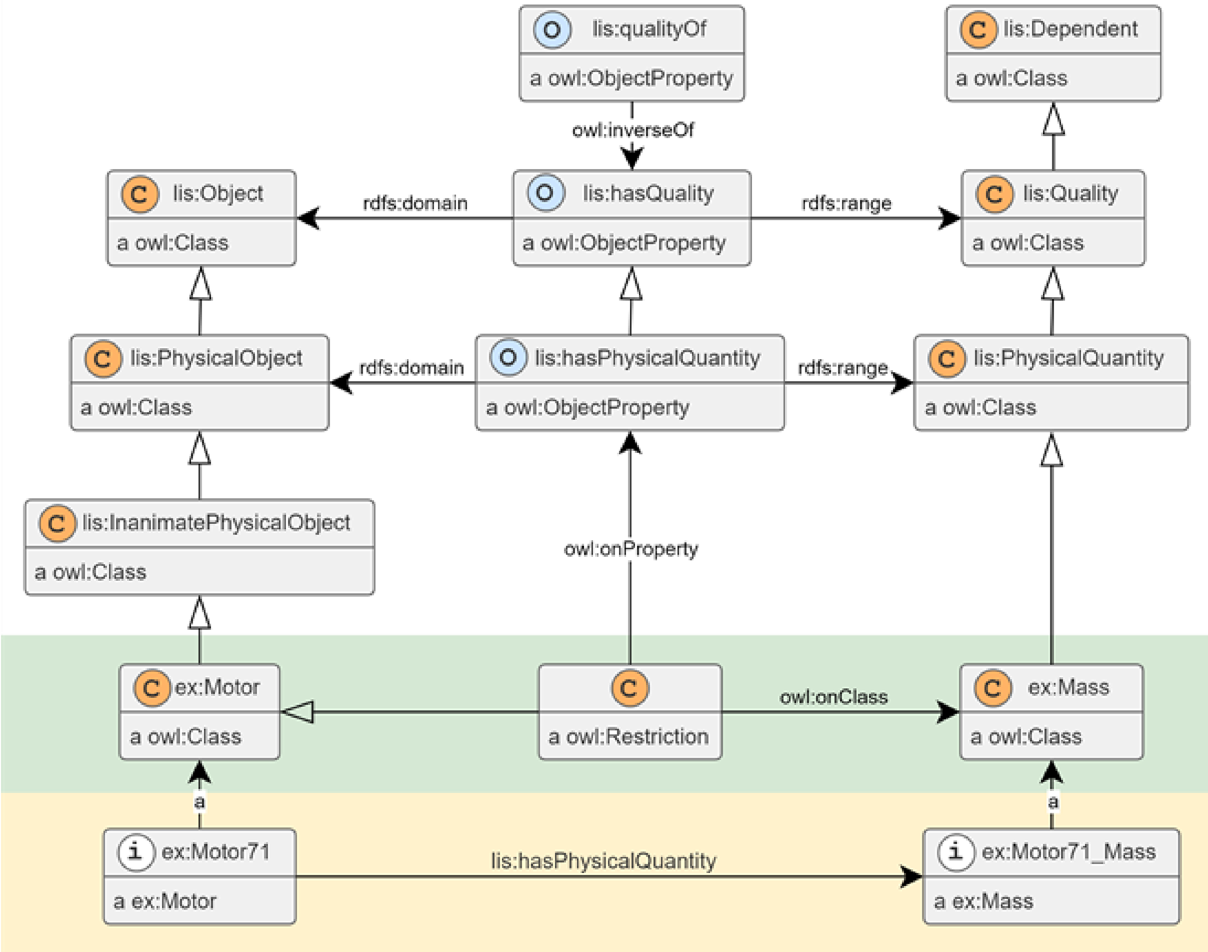

Figure 29. Focus on the elements lis:PhysicalObject on lis:Quality in IDO including an example of how to use them

The owl:Restriction is assigned to the Class ex:Motor and could be used to define the expected cardinality of the assigned Property ex:Mass. The cardinality defined in the W3C standards owl and SHACL can have a minimal value and a maximal value.

- the minimal value expresses if a Property is optional (value 0) or mandatory (value 1) in the context of a Class and
- the maximal value expresses if a Property can be used only once (value 1) or multiple times (no maximum value is defined). Example for the necessity of multiple values could be the Property “First name” (the Author of this concept has the First names Andreas and Christoph) or the Property “Rated frequency” (some power transformers are built for rated frequency 50 Hz and 60 Hz).

The minimal value may depend on the lifecycle of the Class. For example, the Property “Serial number” of the motor is at the beginning of the lifecycle not available and could not be mandatory. However, it must be mandatory at the latest by the time the motor nameplate is to be produced. Such a dependency could be added to the restriction (Table 11).

Table 11. Relation between lis:Object and lis:Quality and their subclasses

Aspect	Available	To be discussed and introduced
Language	–	– EUROPEAN PARTNERSHIP
Knowledge/ Data Governance Policies	• Possibilities to model cardinality for the relations between a Class and its Properties based on W3C standards • Cases what could be expressed with cardinality of the relations between a Class and its Properties	• Guidelines that define patterns for at least the above-mentioned cases. • The patterns should be available as owl:Restriction and as SHACL shape • If possible then conversion rules from owl:Restriction to SHACL shape and vice versa should be defined
Knowledge-Driven Data Integration Systems	–	Infrastructure • to support the users to define the cardinality of the relations between a Class and its Properties, • to create the defined owl:Restrictions or SHACL shapes in the background and • a corresponding converter from owl to SHACL and vice versa.

12.8 Relation between lis:Quality and lis:QualityDatum and their subclasses

The next part is the Relation between lis:Quality and lis:QualityDatum and its subclasses, shown in Figure 30. The green area shows how this part of a Domain Ontology could look like:

- The Property ex:Mass is repeated from the first part
- An owl:Restriction on the relation between the Property ex:Mass and lis:ScalarQuantityDatum is defined (an alternative could have been a SHACL shape)

The corresponding instances (data represented in a knowledge graph) are shown in the yellow area .

In addition, we see that IDO already has defined an owl:Restriction for the (inverse) relation between lis:ScalarQuantityDatum and lis:PhysicalQuantity. No cardinality is defined in this restriction.

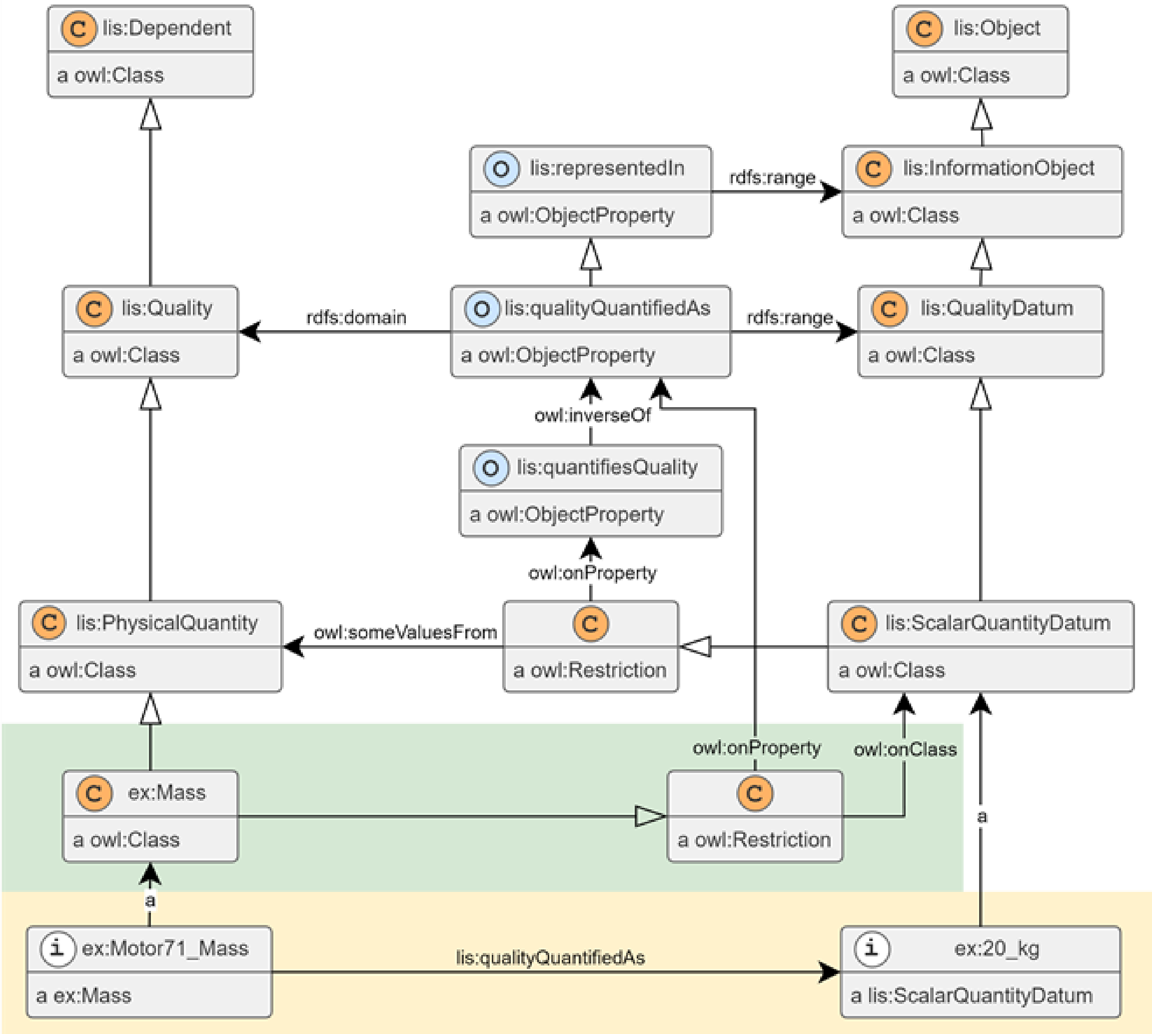

Figure 30. Focus on the elements lis:PhysicalQuality and lis:QualityDatum in IDO including a usage example

The missing owl:Restriction is assigned to the Property ex:Mass and could be used to define the expected cardinality of the assigned lis:ScalarQuantityDatum. The cardinality defined in the OWL and SHACL W3C standards can have a minimal value and a maximal value.

- The minimal value should always be 1, because an instance of a Property ex:Mass makes only sense if it has an instance of lis:ScalarQuantityDatum.
- The maximal value can be used to express if lis:ScalarQuantityDatum. It can have no historical values (value 1) or historical values (no maximum value is defined). Example for the necessity of history is the Property “Name”, which is assigned to an organizational unit because it may be interesting how the name of the same organizational unit changed over time.

The findings are summarized in Table 12.

Table 12. Relation between lis:Quality and lis:QualityDatum and their subclasses

Aspect	Available	To be discussed and introduced
Language	–	–
Knowledge/ Data Governance Policies	• Possibilities to model cardinality for the relations between a Property and its lis:QuantityDatum or lis:ScalarQuantityDatum based on W3C standards • Cases what could be expressed with cardinality of the relations between a Property and its lis:QuantityDatum or lis:ScalarQuantityDatum	• Guidelines that define patterns for at least the above-mentioned case. • The patterns should be available as owl:Restriction and as SHACL shape • If possible then conversion rules from owl:Restriction to SHACL shape and vice versa should be defined
Knowledge-Driven Data Integration Systems	–	Infrastructure • to support the users to define the cardinality of the relations between a Property and its lis:QuantityDatum or lis:ScalarQuantityDatum, • to create the defined owl:Restrictions or SHACL shapes in the background and • a corresponding converter from owl to SHACL and vice versa.

12.9 Relation between lis:QualityDatum and lis:UnitOfMeasure and their subclasses

The last part is the relation between lis:QualityDatum (including its subclasses) and lis:UnitOfMeasure. Figure 31 shows no area for the Domain Ontology. This part is missing in the picture, although restrictions – like defining the allowed units depending on the selected physical quantity or the data type of the literal – could be relevant here. These points are taken up again later in ICE 61360-1:2027.

For now, we are focusing on the restrictions defined in the IDO which are the following:

- an owl:Restriction on the relation between lis:ScalarQuantityDatum and lis:UnitOfMeasure is defined, and
- an owl:Restriction on the relation between lis:ScalarQuantityDatum and the datatype rdfs:Literal is defined.

Again, an alternative could have been a SHACL shape. These two restrictions show which kind of restrictions could be defined on Domain Ontology level. The restrictions defined in the IDO deliberately leave details open. This is different in standards like ICE 61360-1:2017 or QUDT. It therefore makes sense to take a closer look at these standards before delving deeper into the direction defined by the IDO.

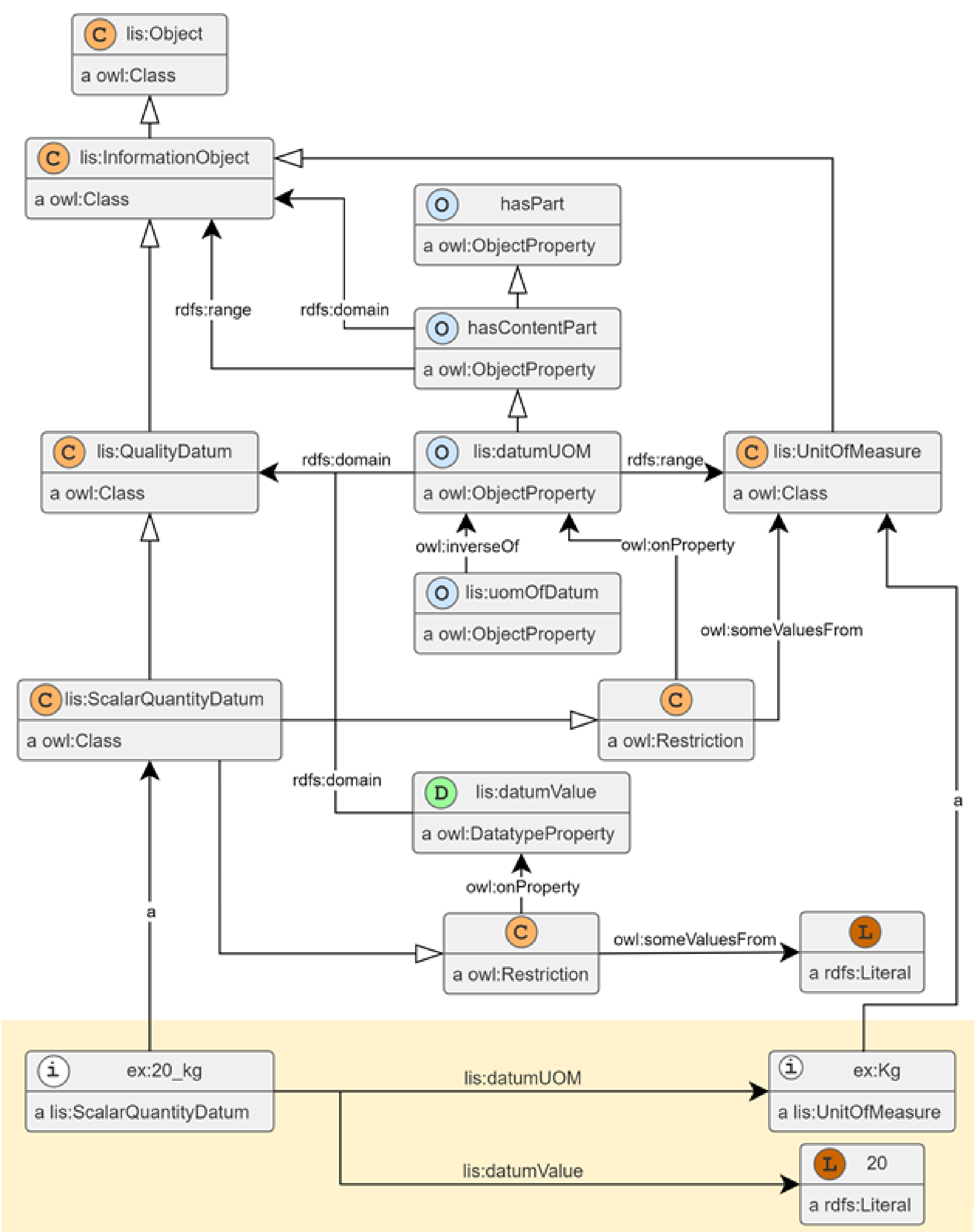

Figure 31. Focus on the elements lis:QualityDatum and lis:UnitOfMeasure in IDO including a usage example

12.10 Data types which could be derived from IEC 61360-1

Table 13 shows a list of data types which could be derived from IEC 61360-1. These types are already available in the RDF representation of ECLASS. A corresponding ontology is developed by Siemens AG and can be reused.

Table 13. Data types which could be derived from IEC 61360-1

Category	Simple Types	Specific Types	Class	xsd
Simple	Binary		BinaryTypeDatum	xsd:base64Binary
Simple	Boolean		BooleanTypeDatum	xsd:boolean
Simple	String		StringTypeDatum	xsd:string
Simple	String	Translatable	TranslatableStringTypeDatum	rdf:langString
Simple	String	Non-translatable	NonTranslatableStringTypeDatum	xsd:string
Simple	String	Date/Time	DateTimeTypeDatum	xsd:dateTime
Simple	String	Date	DateTypeDatum	xsd:date
Simple	String	Time	TimeTypeDatum	xsd:time
Simple			NumberTypeDatum	
Simple	Int		IntTypeDatum	xsd:integer
Simple	Int	Measure	IntMeasureTypeDatum	xsd:integer
Simple	Int	Currency	IntCurrencyTypeDatum	xsd:integer
Simple	Rational		RationalTypeDatum	
Simple	Rational	Measure	RationalMeasureTypeDatum	
Simple	Real		RealTypeDatum	xsd:double
Simple	Real	Measure	RealMeasureTypeDatum	xsd:double
Simple	Real	Currency	RealCurrencyTypeDatum	xsd:double
Class	Reference		ClassReferenceTypeDatum	
Enumeration			IEC61360EnumTypeDatum	
Enumeration	Boolean		EnumBooleanTypeDatum	xsd:boolean
Enumeration	String		EnumStringTypeDatum	
Enumeration	String	Translatable	NonTranslatableStringTypeDatum	rdf:langString
Enumeration	String	Non-translatable	EnumNonTranslatableStringTypeDatum	xsd:string
Enumeration	Int		EnumIntTypeDatum	xsd:integer
Enumeration	Int	Measure	EnumIntMeasureTypeDatum	xsd:integer
Enumeration	Int	Currency	EnumIntCurrencyTypeDatum	xsd:integer
Enumeration	Rational		EnumRationalTypeDatum	
Enumeration	Rational	Measure	EnumRationalMeasureTypeDatum	
Enumeration	Real		EnumRealTypeDatum	xsd:double
Enumeration	Real	Measure	EnumRealMeasureTypeDatum	xsd:double
Enumeration	Real	Currency	EnumRealCurrencyTypeDatum	xsd:double
Enumeration	Reference		EnumReferenceTypeDatum	
Enumeration	Code		EnumCodeTypeDatum	
Level			LevelTypeDatum	
Level	Int		LevelIntTypeDatum	
Level	Int	Measure	LevelIntMeasureTypeDatum	
Level	Real		LevelRealTypeDatum	
Level	Real	Measure	LevelRealMeasureTypeDatum	

13 Conclusion

This deliverable presents the outcomes of the WP3 semantic interoperability analysis among major industrial data models. We explored how the Industrial Data Ontology (IDO) can serve as an upper ontology to bridge different industrial standards, and enable seamless data exchange across organizations and systems. Our collaborative efforts with DISC and CFIHOS Semantics have shown that practical alignment between standards like ISO 10303-239/242 STEP, CFIHOS, DEXPI, and ISO 15926-4 is achievable through IDO-based semantic frameworks. The regular technical meetings and working group sessions have produced concrete modeling patterns that address real-world engineering challenges.

The key finding is that semantic interoperability requires more than just technical standards – it needs consistent modeling patterns, clear governance policies, and practical tools that engineers can use effectively. The different levels of expressiveness identified in IDO provide flexibility for various use cases, from simple database mappings to complex reasoning applications. However, this flexibility also demands careful attention to consistency across implementations. Our analysis revealed that detailed modeling patterns and governance policies are essential for achieving true interoperability between IDO-aligned ontologies.

The work on Use Case 2.9 at Stora Enso demonstrates the practical value of combining semantic technologies with the Arrowhead framework. The proof-of-concept implementation shows that knowledge graphs can effectively integrate design data, operational information, and maintenance records while maintaining semantic consistency across different data sources and formats. Similarly, Use Case 3.9's process industry mapping work between Balas simulation software and standardized data models (PLM RDL and DEXPI Process) revealed both the potential and challenges of semantic transformation, with only 13 out of 41 PLM RDL process classes proving usable for calculation module mapping.

Several important developments emerged from this analysis. The move toward open-sourcing ontology work by major industrial players suggests a shift in how the industry approaches data modeling intellectual property. Companies are recognizing that collaborative development of foundational semantic frameworks benefits everyone more than proprietary solutions that create integration barriers. The integration of SysML v2 with IDO-based ontologies opens new possibilities for systems engineering governance, particularly in complex Systems of Systems environments. This combination addresses the tension between dynamic flexibility and structured rigor that characterizes modern industrial automation.

The mapping analysis between existing standards revealed significant challenges. For instance, comparison between IEC 61987 and ISO 15926-4 instrumentation classes found only 6% exact matches, reflecting genuine differences in how different communities conceptualize industrial equipment. This highlights that complete standardization may not be achievable or even desirable – instead, the goal should be creating sufficient common ground for practical data exchange while preserving specialized knowledge embedded in different industrial communities.

However, challenges remain. The abstract nature of IDO, while providing valuable flexibility, can lead to inconsistent implementations without proper guidance. Our work identified the critical need for modeling patterns that address real-world engineering challenges such as defining mandatory versus optional properties, handling historical data, managing cardinality constraints, and ensuring proper data quality governance throughout the lifecycle of industrial assets.

Looking forward, the successful alignment of these industrial standards through IDO creates a foundation for more efficient and reliable data exchange across the entire industrial value chain. The patterns and guidelines developed through this work provide a roadmap for organizations seeking to implement semantic interoperability solutions. The ultimate goal of autonomous, evolvable interoperability through machine-interpretable content is becoming achievable through the combination of semantic web technologies with established industrial frameworks like Arrowhead.

While technical challenges remain, the collaborative approach demonstrated in this project shows that the industrial community can work together to solve complex interoperability problems that no single organization could address alone. The establishment of three distinct levels of expressiveness in IDO-based ontologies – from highly detailed semantic representations to simplified database mapping shortcuts – provides practical pathways for organizations at different stages of semantic technology adoption.

This analysis establishes IDO as a viable upper ontology for industrial applications and provides the groundwork for the next phase of semantic technology adoption in industrial settings. The success of this effort depends on continued collaboration between standards organizations, technology providers, and end users to refine these approaches and develop the supporting tools and infrastructure needed for widespread deployment.

14 References

[1] Arrowhead fPVN, Deliverable D3.1 “Major industrial data models” 2023 November. https://fpvn.arrowhead.eu/wp-content/uploads/sites/3/2024/10/Arrowhead-fPVN-D3.1-1.pdf
[2] Ontology alignment. https://en.wikipedia.org/wiki/Ontology_alignment
[3] Ontology Alignment Evaluation Initiative. https://oaei.ontologymatching.org/
[4] Euzenat, Jérôme & Shvaiko, Pavel. (2013). Ontology matching: Second edition. 10.1007/978-3-642-38721-0.
[5] Jiménez-Ruiz, E., Cuenca Grau, B. (2011). LogMap: Logic-Based and Scalable Ontology Matching. In: Aroyo, L., *et al.* The Semantic Web – ISWC 2011. Lecture Notes in Computer Science, vol 7031. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-25073-6_18
[6] Jiménez-Ruiz, Ernesto & Grau, B.C. & Zhou, Y. & Horrocks, I. (2012). Large-scale Interactive Ontology Matching: Algorithms and Implementation. Frontiers in Artificial Intelligence and Applications. 242. 444-449. 10.3233/978-1-61499-098-7-444. https://www.researchgate.net/publication/286532961_Large-scale_interactive_ontology_matching_Algorithms_and_implementation/citation/download
[7] SWSA Ten-Year Award. https://swsa.semanticweb.org/content/swsa-ten-year-award
[8] Chen, J., Hu, P., Jimenez-Ruiz, E., Holter, O. M., Antonyrajah, D., & Horrocks, I. (2021). OWL2Vec*: embedding of OWL ontologies. *Machine Learning*, *110*, 1813-1845. https://doi.org/10.1007/s10994-021-05997-6
[9] Teymurova S, Jiménez-Ruiz E, Weyde T, Chen J. OWL2Vec4OA: Tailoring Knowledge Graph Embeddings for Ontology Alignment. In International Knowledge Graph and Semantic Web Conference 2024 Dec 12 (pp. 168-182). Cham: Springer Nature Switzerland.
[10] Jiaoyan Chen, Ernesto Jimenez-Ruiz, Ian Horrocks, Denvar Antonyrajah, Ali Hadian, Jaehun Lee. Augmenting Ontology Alignment by Semantic Embedding and Distant Supervision. European Semantic Web Conference, ESWC 2021. https://github.com/KRR-Oxford/OntoAlign
[11] Pål Rylandsholm - Industrial automation systems and integration – Ontology-based Interoperability (OBI) (2023) https://rds.posccaesar.org/ontology/lis14/ https://rds.posccaesar.org/InformationAboutOBI.pdf
[12] Niklas Lindström - Target Vocabulary Maps. https://github.com/niklasl/ldtvm
[13] STROMA: Semantic Refinement of Ontology Mappings. https://dbs.uni-leipzig.de/research/projects/stroma
[14] Péter Olaszi (2025). A survey of ontology frameworks. https://github.com/d893a/ontology/blob/main/Olaszi%20P%C3%A9ter%20-%20A%20survey%20of%20ontology%20frameworks.2025.pdf
[15] What Software Engineering Can Teach Knowledge Engineers about Version Control. https://enterprise-knowledge.com/what-software-engineering-can-teach-knowledge-engineers-about-version-control/
[16] Arrowhead fPVN, Deliverable D9.2: Process industry use case first year progress and next step specification. 2024.05.31. https://fpvn.arrowhead.eu/deliverables/
[17] Arrowhead fPVN, Deliverable D4.2: Second generation of translation technology. 2024.11.25. https://fpvn.arrowhead.eu/deliverables/

[18] VTT BALAS – Conceptual process design to optimize resource efficiency. VTT Technical Research Centre of Finland Ltd. https://www.vttresearch.com/en/explore/vtt-balas-conceptual-process-design-optimise-resource-efficiency (accessed 28.10.2024)

[19] POSC Caesar Association, 2024. Reference Data Library: Industrial data ontology & Product Lifecycle Management. URL https://rds.posccaesar.org/doc/ (accessed 15.10.2024).

[20] DEXPI Initiative, 2023. DEXPI Process Information Model Version 1.0. DEXPI. https://dexpi.org/specifications/

15 Revision history

Version	Date	Contributor	Contributions
0.1	2025.03.14	Nils Sandsmark	Initial draft
0.2	2025.03.20	Oslo meeting participants	Review the draft
0.3	2025.05.06	Oslo meeting participants	Review the draft
0.4	2025.05.08	Melinda Hodkiewicz	Add notes to Section 5.1 “Basis for developing new OBI standards”
0.5	2025.05.13	Péter Olaszi	Add content to Sections 4 “Vocabulary (Terms and definitions)”, 6 “Creating new ontologies with IDO as upper ontology”, and 7 “Ontology alignment”.
0.6	2025.05.13	Onno Paap	Add content to Sections 8.1.2 “ISO 15926-2: Data model”, 8.1.3 “ISO/TS 15926-4: Core reference data”, 9 “Starting from the use-cases”
0.7	2025.05.13	Nils Sandsmark	Add content to Section 2 “WP3 Task 3.2 Objectives”
0.8	2025.05.13	All	Add 2025.05.13 meeting insights into Sections 1 “Introduction” and Section 3 “Requirements”
0.9	2025.05.15	All	Add 2025.05.15 meeting insights: Sections 5.2 “Interoperability between major standards used in industry” and 5.2.3 “DEXPI to IDO”
0.10	2025.05.20	All	Add 2025.05.20 meeting insights to Section 5.2.3 “DEXPI to IDO”
0.11	2025.05.21	Onno Paap	Add content to Section 5.2.2 “CFIHOS Semantics to IDO”
0.12	2025.05.22	Torbjörn Holm	Add content to Section 9 “Starting from the use-cases”
0.13	2025.05.22	All	Add 2025.05.22 meeting insights to Sections 3 “Requirements”, 5.2 “Interoperability between major standards used in industry”, 5.2.2 “CFIHOS Semantics to IDO”, 5.2.3 “DEXPI to IDO”.
0.14	2025.05.27	Magne Valen-Sendstad	Add content to Section 5.2.4 “Reference data for the Process Plant use-cases”
0.15	2025.05.27	All	Add 2025.05.27 meeting insights to Sections 1.1 “Ontology domain gamut vs. domain coverage”, 1.2 “Evolution of ontologies”, 5.2.3 “DEXPI to IDO”, 7 “Ontology alignment”.
0.16	2025.05.28	Andreas Neumann	Add proposal for Appendix 12.1 “Patterns for IDO aligned ontologies”
0.17	2025.05.28	Géza Kulcsár	Add Section 10.1 “SysML”
0.18	2025.06.03	Heiner Temmen	Amend Section 5.2.3.1 “Process Modeling with IDO”
0.19	2025.06.04	Péter Olaszi	Rename document to "D3.2 Major industrial data model semantics analysis - Draft report". Copy over content from “IR3.2.20250417100622.v0.1.docx”.
0.20	2025.06.04	Heiner Temmen	Amend Section 5.2.3.1 “Process Modeling with IDO”
0.21	2025.06.04	Onno Paap	Add example to Section 9.2.2 “CFIHOS Semantics to IDO”
0.22	2025.06.04	Péter Olaszi	Rewrite Section 1 “Introduction”
0.23	2025.06.04	Péter Olaszi	Add content to Section 6 “Ontology alignment and ontology mapping”. Amend Sections 6.1 “Ontology gamut and domain coverage”, 6.2 “Evolution of ontologies”.

Version	Date	Contributor	Contributions
0.24	2025.06.05	Andreas Neumann	Add content to Appendix
0.25	2025.06.05	Péter Olaszi	Review the complete Appendix
0.26	2025.06.05	Péter Olaszi	Add content to Section 4 “Artefacts”. Review the document.
0.27	2025.06.06	Péter Olaszi	Add content to Section 11 “A move towards open-sourcing ontology work”
0.28	2025.06.06	Péter Olaszi	Review the document. Edit, reorganize sections.
0.29	2025.06.06	Andreas Neumann	Add Section 14.3 “Example for a Centrifugal Pump Tag”
0.30	2025.06.06	Andreas Neumann	Add Section 14.10 “Data types which could be derived from IEC 61360-1”. Amend Section 14.1 “List of patterns for IDO aligned ontologies”.
0.31	2025.06.06	Péter Olaszi	Review the document. Edit, reorganize sections.
0.32	2025.06.06	Péter Olaszi	Add Conclusion
1.0	2025.06.06	Péter Olaszi	Document ready for external review
1.1	2025.06.10	Péter Olaszi	Address review comments: Rename Section 4 to “Synergy”. Move contents of Appendix to Section 11 “Discussion”. Rename Section 3.1 to “Ontology scope and domain coverage”. Review Section 6.4 “ISO 15926-4 to IDO”

15.1 Contributing and reviewing partners

Representing partner	Contributor	Contents	Review
PCA	Nils Sandsmark	X	
PCA	Magne Valen-Sendstad	X	
UWA	Melinda Hodkiewicz	X	
AITIA	Péter Olaszi	X	X
Fluor/CFIHOS	Onno Paap	X	
TBHK	Torbjörn Holm	X	
Independent/Siemens Energy	Andreas Neumann	X	
IncQuery	Géza Kulcsár	X	
Independent/DEXPI e.V.	Heiner Temmen	X	
VTT	Teemu Sihvonen	X	
VTT	Teemu Mätäsniemi	X	
VTT	Lotta Sorsamäki	X	

15.2 Amendments

No.	Date	Version	Subject of Amendments	Author
1	2025-06-10	1.1	Rename Section 4 to “Synergy”, update Section 11 “Discussion”, Rename Section 3.1 to “Ontology scope and domain coverage”, Review Section 6.4 “ISO 15926-4 to IDO”	Péter Olaszi
2	2025-06-10	1.2	Fix margins, references, figures, formatting, typography. No content change.	Péter Olaszi
2	2025-06-11	1.3	Add Section 9 “Arrowhead fPVN Use Case 3.9: Process industry”.	Teemu Sihvonen, Teemu Mätäsniemi, Lotta Sorsamäki

15.3 Quality assurance

No	Date	Version	Approved by
1	2025.06.12	1.3	Jerker Delsing