IDO training – CFIHOS Semantics

Implement the CFIHOS Semantics domain ontology in IDO upper ontology

Onno PAAP

Fellow Semantic Technologies and Data Interoperability

onno.paap@fluor.com

Former presentations

OBI - IDO training - session 12 - 31.1.2025

POSC Caesar Association • 162 views • 1 year ago

This session will focus on CFIHOS - Q/A and discissions. Onno Paap, project leader of CFIHOS Semantics and Peter Townson , Project Director CFIHOS (IOGP JIP36) participate and answer questions.

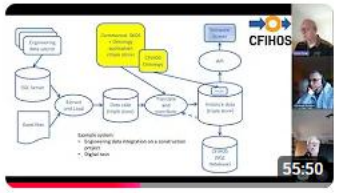
CFIHOS general

OBI - IDO - Training - session 18 - 21.04.2026

POSC Caesar Association • 46 views • 2 weeks ago

Life cycle for IDO – Aligned requirements with Heiner Temmen DEXPI e.V., Onno Paap Fluor and Torbjörn Holm TBH Konsult

Lifecycle



OBI - IDO training - session 11 - 24.1.2025

POSC Caesar Association • 188 views • 1 year ago

This session will focus on CFIHOS Semantics. CFIHOS Semantics will be presented by Onno Paap, project leader of CFIHOS Semantics.

Implementing semantic technology - overview

Today:

- Why ontology
- Choice of database
- IDO and lifecycle datasets
- Building up the ontology

Cheat sheet:

- RDF = graph data model, in triples.
- LPG = also graph data model, in nodes, edges and properties.
- Ontology = shared meaning
- CFIHOS = industry data standard for handover
- IDO = upper ontology
- SKOS = vocabulary structure

Why ontology.

- Problem with Traditional RDBMS
  - Fixed schemas → slow to evolve
  - Schema changes can take months to years
  - Becomes a bottleneck in engineering projects
- Semantic Approach (RDF + Ontology)
  - Flexible, schema-light model
  - Structure evolves rapidly (like a spreadsheet)
  - Ontology embeds meaning and relationships
- Key Advantage
  - *Data is no longer just stored. It becomes self-describing and adaptable*

Why this matters

From fragmented engineering data → to a usable digital twin

Benefits for EPC contractors during the project phase

- Handling Project Changes
  - Semantic technology separates meaning from schema, enabling flexible adaptation to evolving project requirements without database redesign.
- Reduced IT Dependence
  - Engineers can directly map new data attributes using ontologies, minimizing delays and reliance on IT teams during integration.
- Improved Data Consistency
  - Shared ontologies ensure consistent concepts across systems, improving data quality and tracking provenance during projects.

Benefits for Owner/Operators During the Operations Phase

- Challenges of Traditional Data
  - Traditional data delivery fragments asset information, making it hard to trust, connect, and use efficiently during operations.
- Semantic Digital Twin Solution
  - Semantic digital twins connect diverse data sources into a unified, queryable knowledge graph for complete asset insight.
- Operational Impact and Benefits
  - Faster trusted data access reduces downtime, improves maintenance quality, and lowers operational risks significantly.
- Future-Proofing Asset Information
  - Semantic technology evolves with new requirements, enabling continuous, practical, and trusted asset lifecycle management.

A central database on each project yes, but why not stick to relational?

Traditional Databases	Semantic Approach (Ontology + RDF)
Rigid schemas	Flexible, evolving structure
Costly schema changes	Adapt instantly without redesign
Hard to integrate	Designed for integration
Data lives in silos	Connected, contextual knowledge
Meaning in applications	Meaning embedded in the data

Relational databases store structure.
Semantic models store meaning.

Choose the right graph technology

LPG (e.g., Neo4j) Labeled Property Graph

- Nodes + edges + properties
- Optimized for graph traversal & performance
- Meaning is implicit in structure and code

Best for:

- Pathfinding, recommendations, network analytics
- Performance-heavy graph queries

RDF + Ontology (Semantic Web Graph)

- Subject–Predicate–Object triples
- Ontology defines explicit meaning and relationships
- Schema evolves without redesign

Best for:

- Data integration across different sources
- Complex domains requiring shared semantics
- Knowledge graphs with long lifecycle and reuse

LPG	RDF + Ontology
Fast traversal	Rich semantics
Flexible structure	Explicit meaning
App-driven logic	Model-driven logic
Local optimization	Cross-system integration

For engineering data integration

RDF + ontology is the right choice

Solving Real Engineering Challenges

- Data Integration at Scale
  - 40+ databases + hundreds of spreadsheets per project
  - Engineers define mappings directly via ontology (no long IT cycles)
- Unlocking Unstructured Data
  - PDFs, drawings, 3D models → traditionally hard to use
- Future-Proofing
  - Digital Product Passports / Asset Administration Shell → 100,000+ properties
  - LLMs + ontology → extract meaningful structured knowledge
  - Relational schemas don’t scale to this complexity
  - RDF handles massive, evolving relationships naturally
- Conclusion

For data integration → semantic technology is the better fit

Concrete Example: Pump Failure Analysis

Today

- Engineer checks multiple systems(maintenance, docs, vendor, engineering)
- Manual linking of data
- Takes hours to days
- Risk of missing information

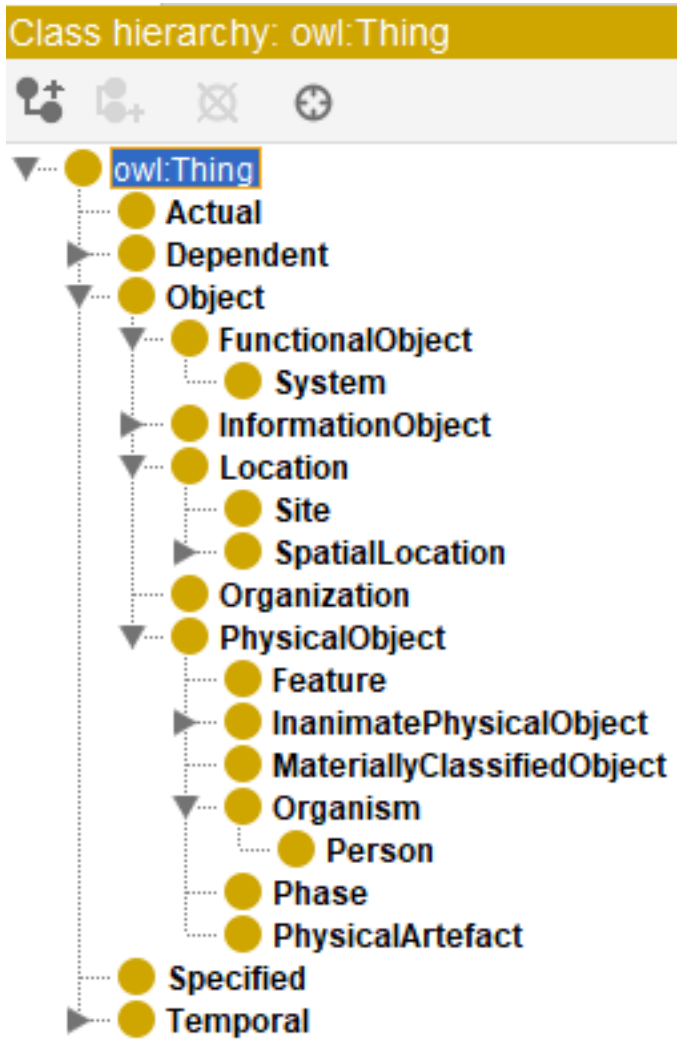
With semantic approach

- Query one system for Pump P-101
- All data automatically connected
- Full context instantly available
- Takes minutes

From searching & guessing → to instant insight

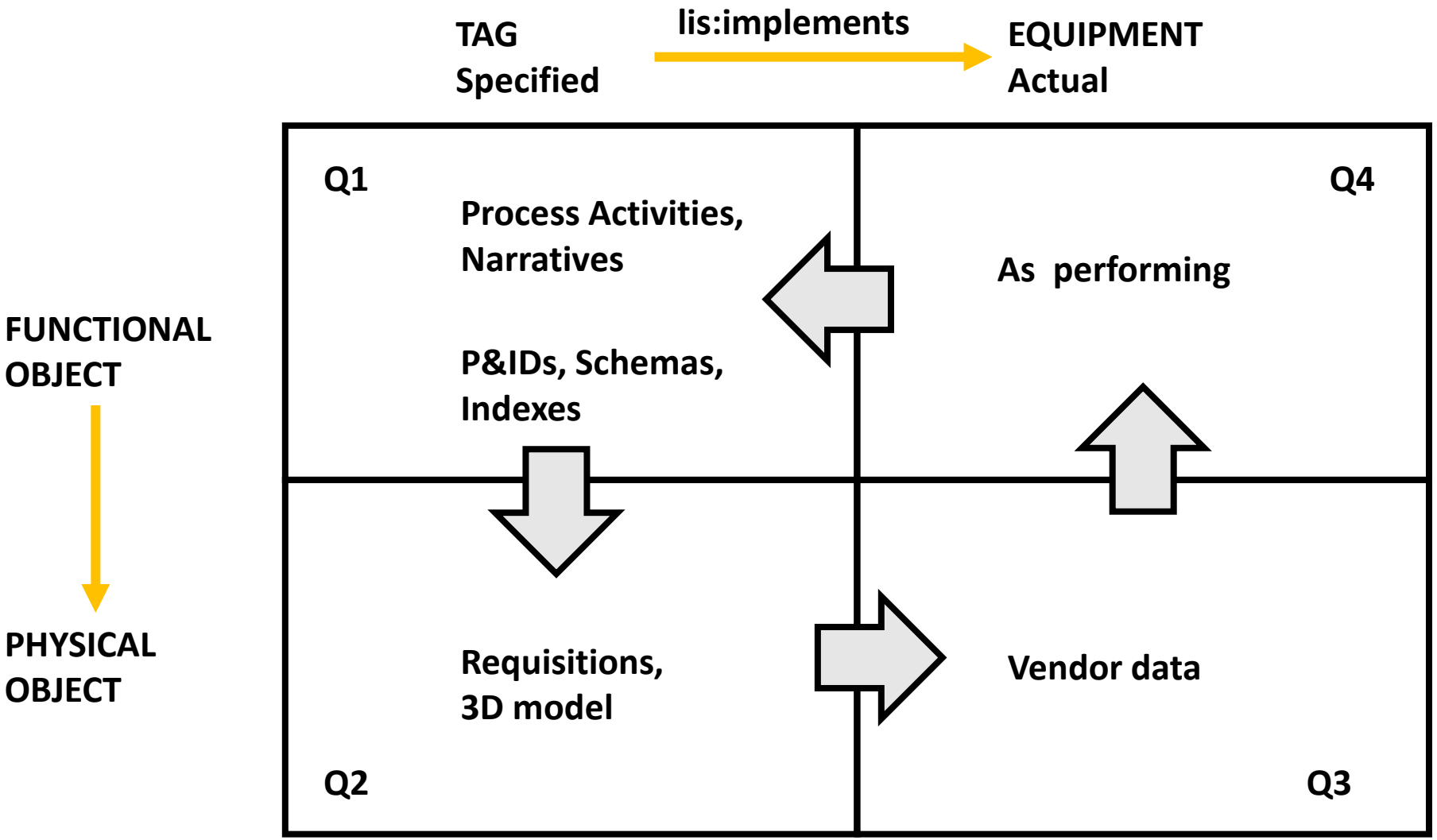
Get the IDO turtle: curl --insecure -H "Accept: text/turtle" -L http://rds.posccaesar.org/ontology/lis14/ont/core > ido.ttl

The IDO upper ontology mapping proposal

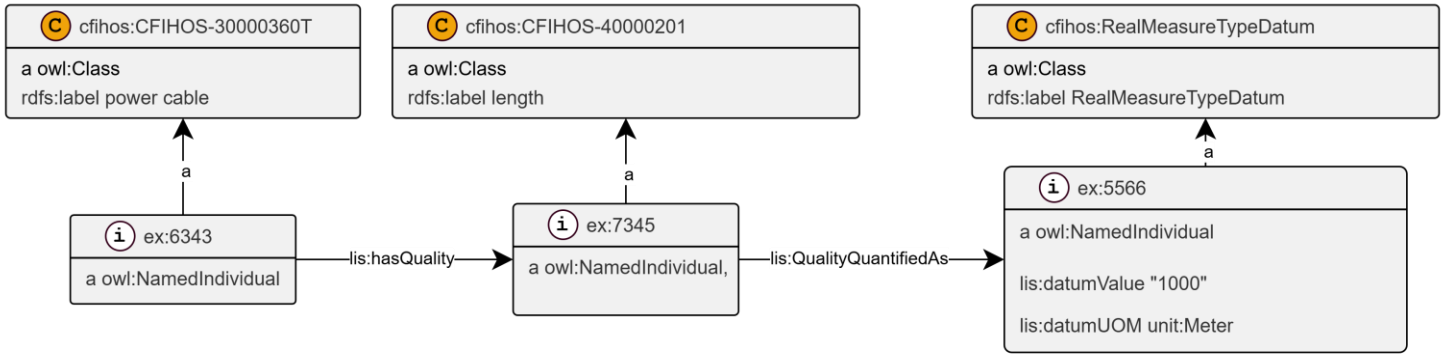


IDO	CFIHOS RDL
	Cover and index
	Guidance
	RDL master object
Dependent	data dictionary
	CFIHOS object equivalent mapping
Organization	discipline
InformationObject	document type
	discipline document type
PhysicalArtefact	equipment class
OWL restrictions	equipment class property
	handover event
Dependent	property
InformationObject	property picklists values
InformationObject	property groupings
	source standard
	IOGP JIP33 Information requirements sepecification
OWL restrictions	document requirement per class
FunctionalObject	tag class
OWL restrictions	tag class property
isRealizedBy	tag class equipment class relationship
	tag or equipment class source standard
	tag or equipment class property source standard
QUDT	unit of measure

Lifecycle datasets



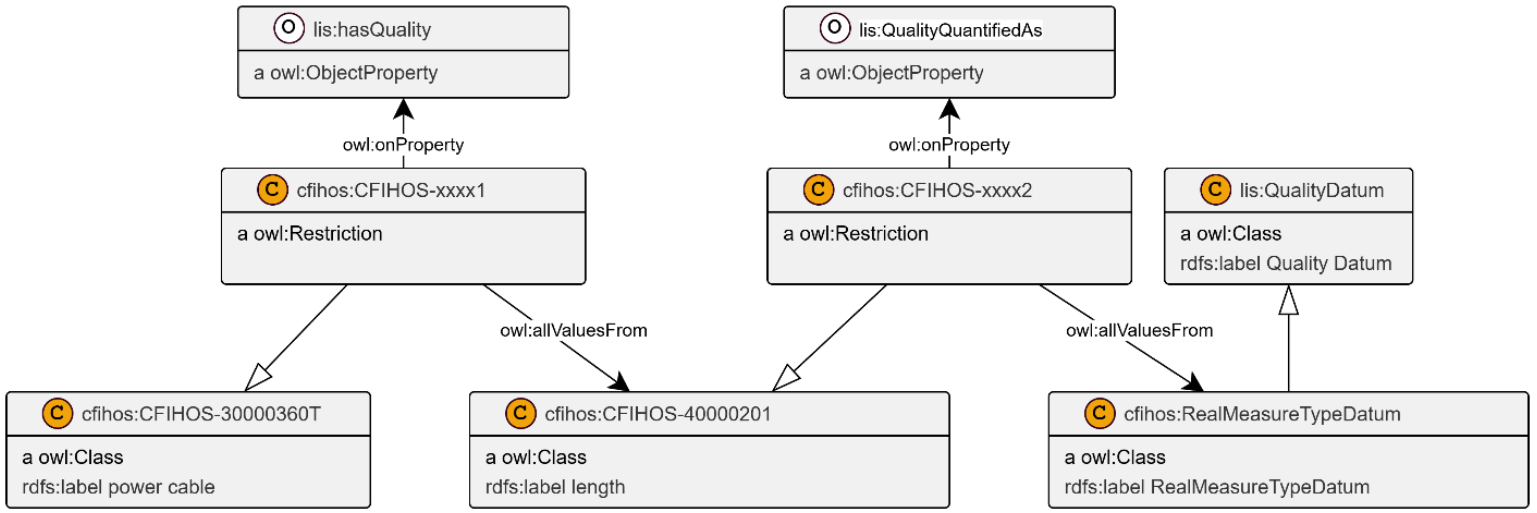
Starting with instances



Power Cable property length

```
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix lis: <http://rds.posccaesar.org/ontology/lis14/rdl/> .
@prefix unit: <http://qudt.org/vocab/unit/> .
@prefix cfihos: <http://data.cfihos.org/rdl/> .
@prefix ex: <http://example.com/> .

cfihos:CFIHOS-30000360T a owl:Class ;
    rdfs:label "power cable" .

cfihos:CFIHOS-40000201 a owl:Class ;
    rdfs:label "length" .

cfihos:RealMeasureTypeDatum a owl:Class ;
    rdfs:label "RealMeasureTypeDatum" .

unit:Meter a owl:Class ;
    rdfs:label "Meter" .

lis:hasQuality a owl:ObjectProperty .

lis:QualityQuantifiedAs a owl:ObjectProperty .

ex:6343 a owl:NamedIndividual ;
    a cfihos:CFIHOS-30000360T ;
    lis:hasQuality ex:7345 .

ex:7345 a owl:NamedIndividual ;
    a cfihos:CFIHOS-40000201 ;
    lis:QualityQuantifiedAs ex:5566 .

ex:5566 a owl:NamedIndividual ;
    a cfihos:RealMeasureTypeDatum ;
    lis:datumValue "1000"^^xsd:string ;
    lis:datumUOM unit:Meter .
```

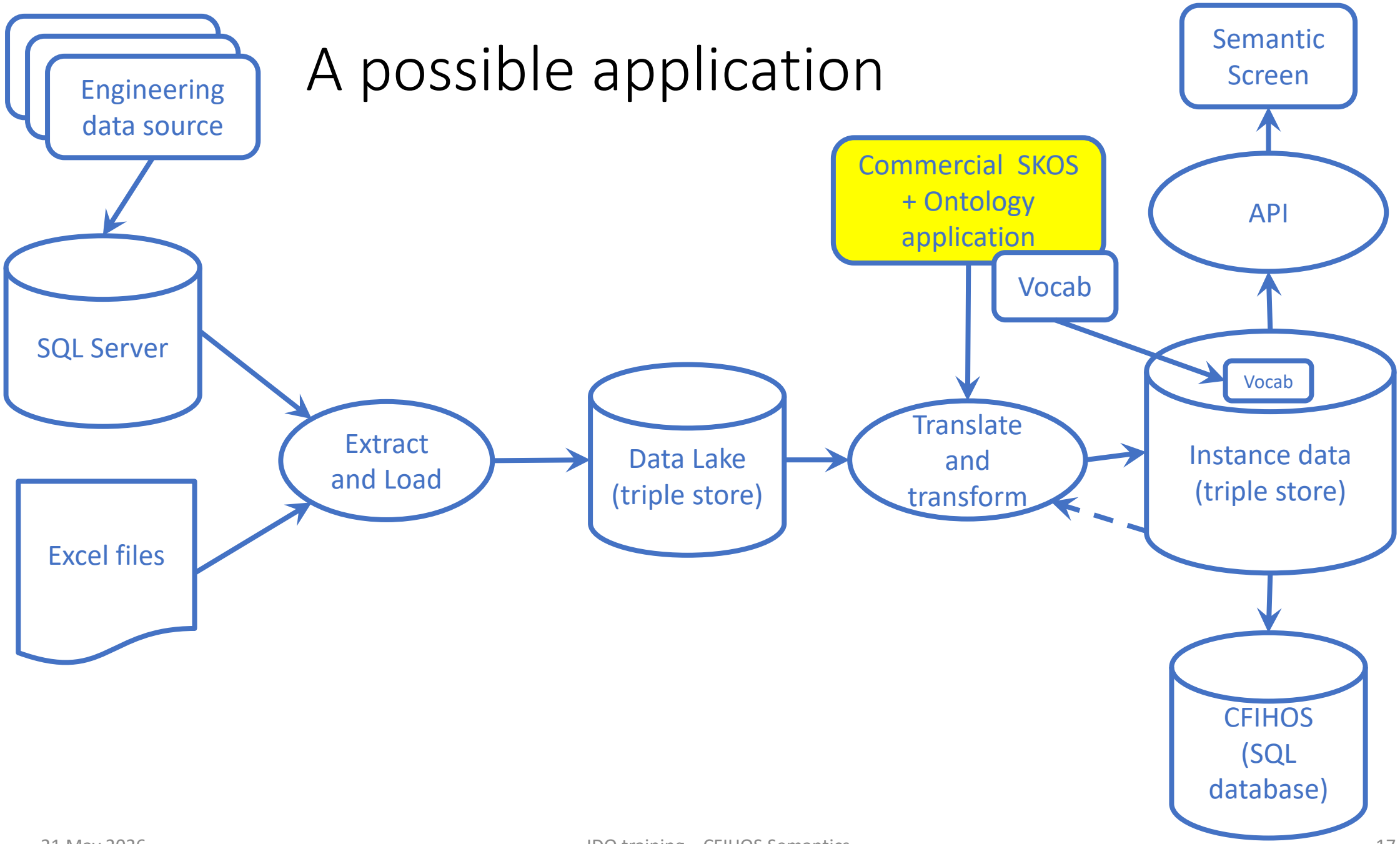
OWL restrictions for allowable properties



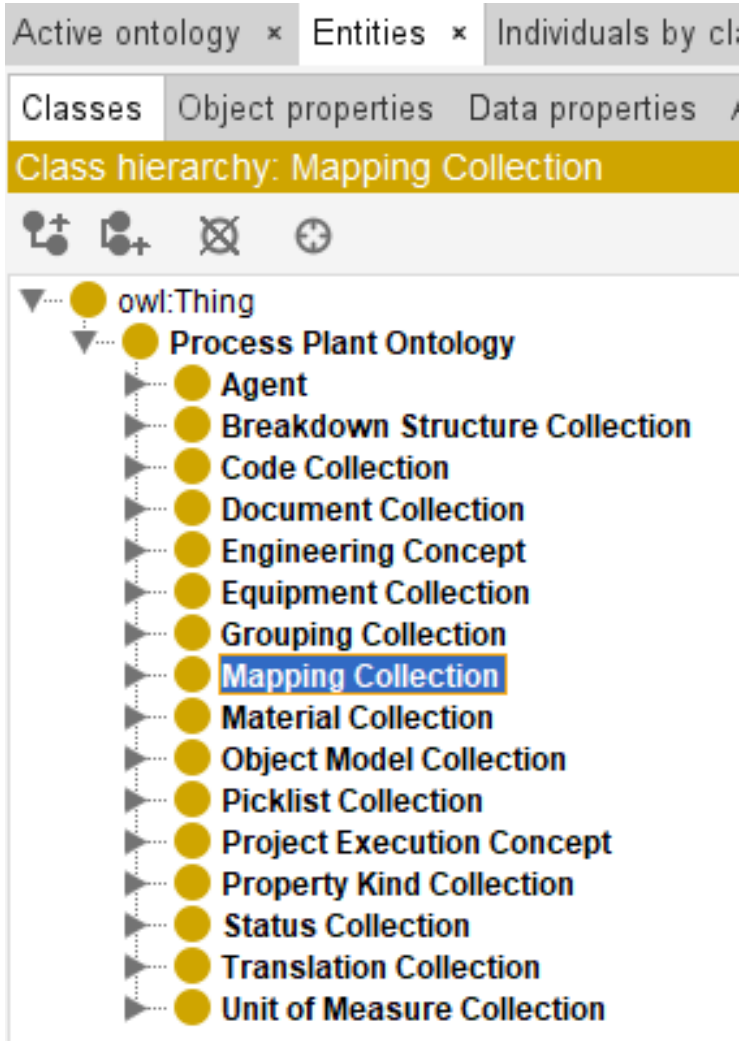
The Length as allowable Dependent property of Power Cable

```
@prefix : <http://example.org/> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix lis: <http://rds.posccaesar.org/ontology/lis14/rdl/> .
@prefix unit: <http://qudt.org/vocab/unit/> .
@prefix cfihos: <http://data.cfihos.org/rdl/> .
@prefix ex: <http://example.com/> .

cfihos:CFIHOS-30000360T a owl:Class ;
    rdfs:label "power cable" .

cfihos:CFIHOS-40000201 a owl:Class ;
    rdfs:label "length" .

cfihos:RealMeasureTypeDatum a owl:Class ;
    rdfs:label "RealMeasureTypeDatum" ;
    rdfs:subClassOf lis:QualityDatum .

lis:QualityDatum a owl:objectProperty, owl:Class ;
    rdfs:label "QualityDatum" .

lis:hasQuality a owl:objectProperty.

lis:QualityQuantifiedAs a owl:objectProperty.

cfihos:CFIHOS-xxxx1 a owl:Restriction ;
    rdfs:subClassOf cfihos:CFIHOS-30000360T ;
    owl:onProperty lis:hasQuality;
    owl:allValuesFrom cfihos:CFIHOS-40000201 .

cfihos:CFIHOS-xxxx2 a owl:Restriction ;
    rdfs:subClassOf cfihos:CFIHOS-40000201 ;
    owl:onProperty lis:QualityQuantifiedAs;
    owl:allValuesFrom cfihos:RealMeasureTypeDatum .
```

A possible application



The parts of a projects-proven domain ontology



The ontology is SKOS-based with links to an upper ontology.

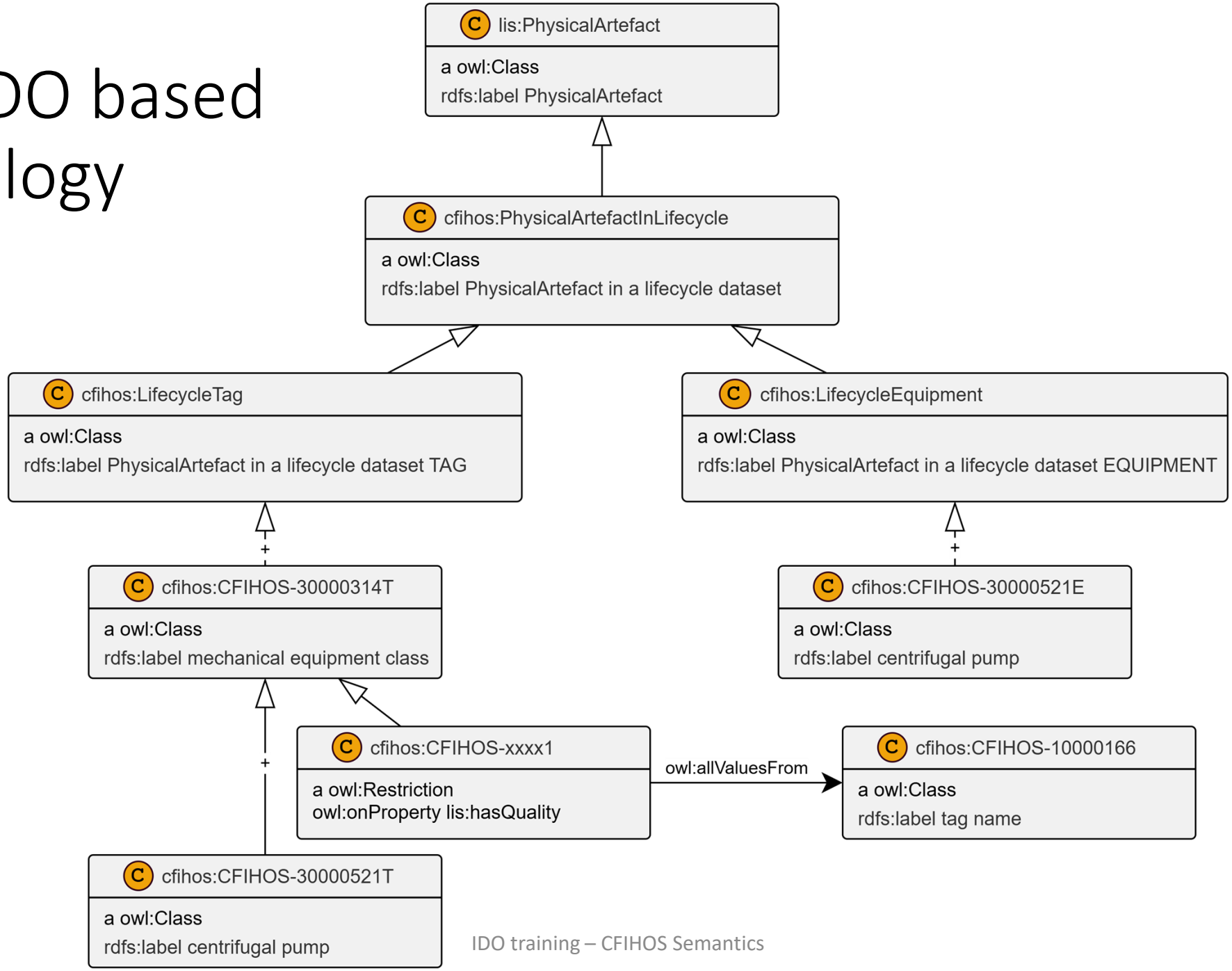
Core

- **Agent** is AI agents, persons and organizations.
- **Breakdown structure** is specific per project.
- **Code collection** is engineering codes. Works with Picklists .
- **Document collection** is document types.
- **Engineering concept** is things like Work Breakdown structure.
- **Equipment collection** is classes for plant items.
- **Material collection** is material classes.
- **Object Model collection** is all allowable properties per class, divided in TAG and EQUIPMENT
- **Project Execution Concept** is all terms on project planning
- **Property Kind collection** is the properties taxonomy.
- **Status collection** is all statuses.
- **Unit of Measure collection** is an uploaded QUDT (=UOM ontology)

Control

- **Grouping collection** is kinds of groups, like property groups.
- **Mapping collection** is control over all source data mapping.
- **Picklist collection** works with Code collection.
- **Translation collection** is used to translate all terms that have been typed in wrongly in the source data.

An IDO based ontology



The takeaway

- Engineering data is complex and fragmented
- Traditional (relational) systems struggle with:
  - Change
  - Integration
  - Scale
- Semantic technology (RDF + ontology):
  - Connects data across systems
  - Embeds meaning in the data
  - Adapts as data is added
- Result
  - From disconnected data → to a connected, usable digital twin

Finally

- This is not about a new database
- This is about:
  - Making data understandable
  - Making data connected
  - Making data reusable across the lifecycle
- Shift in thinking:
  - From: storing data in systems
  - To: creating meaningful, connected knowledge
- Key message
  - If we get the meaning right, everything else becomes easier