JOT
JOURNAL OF
OBJECT TECHNOLOGY

Product Digital Twin Ontology: A Pivot for Bridging Asset Administration Shell-based and Ontology-based Digital Product Passports

Quang-Duy Nguyen, Fatima Danash, Arnab Sinha, and Chokri Mraidha
Université Paris-Saclay, CEA, List, France

ABSTRACT

The Digital Product Passport (DPP) is an essential conceptual technology enabling circular manufacturing (CM). It provides a mechanism for stakeholders in CM to manage the entire lifecycle of individual products, from material sourcing to their end-of-life, while also promoting reduction, reuse, and recycling strategies. To implement the DPP, many industrial organizations rely on the Asset Administration Shell (AAS) and the ECLASS vocabulary. On the one hand, the combination of these two standards is robust, as they are well-defined, easy to use, and supported by a broad industrial community. On the other hand, many other partners in both public and private sectors are exploring alternative solutions that are open, more distributed, and knowledge-centric, such as ontology-based approaches. To harmonize these two perspectives, this paper proposes a new ontology, called Product Digital Twin Ontology (PDTO), and a method that uses PDTO as the pivot for converting between AAS-based and ontology-based DPP representations.

KEYWORDS Industry 5.0, Circular Manufacturing, Digital Product Passport, Ontology, Asset Administration Shell

1. Introduction

Circular Manufacturing (CM) is a key model for advancing the sustainability direction in Industry 5.0 (Maija Breque et al. 2021). It can be described as *"a system of ideally endless reutilization, remanufacturing, and recycling of resources and goods"* (Mogos & Fragapane 2022). In which, the practices of the 10R framework (refuse, rethink, reduce, reuse, repair, refurbish, remanufacture, repurpose, recycle, recover) are applied by stakeholders across the supply chain to reduce material scarcity and environmental impacts, and improve product lifespan and resource efficiency. In this context, data generated throughout the lifecycle of products (from material sourcing to their end-of-life) need to be made accessible to all stakeholders.

JOT reference format:
Quang-Duy Nguyen, Fatima Danash, Arnab Sinha, and Chokri Mraidha. *Product Digital Twin Ontology: A Pivot for Bridging Asset Administration Shell-based and Ontology-based Digital Product Passports*. Journal of Object Technology. Vol. 25, No. 3, 2026.
http://dx.doi.org/10.5381/jot.2026.25.3.a21

While addressing the above requirement, the European Commission (EC) advocates for the Digital Product Passport (DPP) as a core component of its Ecodesign for Sustainable Products Regulation (ESPR) (European Parliament and the Council of the European Union 2024). Conceptually, DPP is a *"collection of all product information relevant for circularity and authorities"* (Christoph Attila Ku 2024) and serves as an instrument to make such *"information available to actors along the entire value chain"* and over the product lifecycle. Each DPP instance is uniquely associated with an individual product instance and is digitally stored and managed within a computing infrastructure, referred to as a DPP system (Jansen et al. 2023). The DPP system can be a single server or a composition of multiple physical servers and logical components, allowing stakeholders' systems to connect to and access DPP instances. Figure 1 illustrates a DPP system serving as a common access point to shared DPP instances in CM. The information of DPP instances must be represented in a format accepted by all involved stakeholders' systems, together with using a data schema and vocabulary that can be clearly analyzed and processed by such systems.

The proposed DPPs must ensure both syntactic and semantic interoperability among all stakeholders involved in CM.

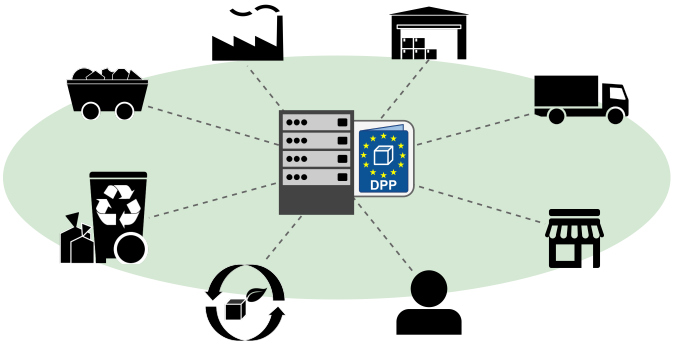

Figure 1 DPP as an instrument to share circularity and regulatory data among stakeholders across a supply chain

Three approaches to DPP semantic representation (schema + external dictionary, schema + context, and ontology) pave the way for standards and technologies to support CM (Hbeich et al. 2026). Among these, the Asset Administration Shell (AAS) standard is emerging as one of the most widely adopted in practice (Wicaksono et al. 2025). Its success relies not only on its clear, easy-to-implement, and modern-designed schemas aligned with experiences from the German industrial community, but also on the strong support receiving from multiple European projects and research studies. For example, Catena-X[1], a project funded by NextGenerationEU to support data spaces in the automotive sector, proposes a vision for the DPP in which DPPs are aligned with the AAS standard and deployed on AAS-based systems (Ann-Carina Tietze et al. 2023); DaCapo[2], a European-funded project, defines a set of tools compatible with AAS-based DPPs to support human actors in circular innovation of multiple domains, such as aeronautics, smartphones, and warehousing (Valtanen et al. 2024); RAASCEMAN[3] is another European project that identifies the use of the AAS standard as the core for modeling product digital twins and DPPs within its cross-sector Manufacturing-as-a-Service architecture (Nguyen et al. 2025). Moreover, the AAS can be used with an external dictionary, such as the ECLASS[4] standard, as a complement and an enrichment of its schema. ECLASS offers a big collection of well-defined and standardized terms related to products and their properties. In which, each term is assigned to a globally unique identifier, improving not only the semantic consistency of AAS but also the interoperability across systems. The combination of the AAS and ECLASS is robust; however, it remains constrained by centralized governance, licensing costs, and limited readiness for semantic reasoning.

The ontology-based approach has also received significant attention within the DPP development community. This interest stems not only from its potential to address the limitations of the AAS + ECLASS combination, but also from of the compelling potentials that ontology-based modeling brings to DPPs: from ensuring interoperability beyond structural data formats (Zhang et al. 2026), to enabling the common understanding of complex lifecyle and circular economy concepts (Pourjafarian et al. 2025), to supporting automated reasoning and decision making (Adu-Duodu et al. 2025), to promoting modularity and the reuse of existing standards, vocabularies, and ontologies (Kebede et al. 2024). Several projects explore ontologies in the context of DPPs. The European-funded CIRPASS[5] project investigates the semantic interoperability mechanisms for European DPP deployment, followed by the CIRPASS-2[6] project which advances toward demonstrating the practical implementation of ontology-based DPPs. The Catena-X initiative also incorporates ontology-based semantic models to enable interoperable DPP-related use cases. In particular, the Catena-X standard (Catena-X Automotive Network e.V. 2023), *CX-0067: Ontology Models to Realize Federated Query in Catena-X*, specifies the use of W3C Semantic Web technologies (e.g., RDF, OWL, SPARQL) to support federated queries and semantic interoperability across organizations within the automotive data ecosystem. Finally, the Global Battery Alliance[7] initiative has developed the Battery Passport as a standardized DPP for batteries aiming to harmonize sustainability and lifecycle data reporting across the global battery value chain (Gianvincenzi et al. 2024).

Another interoperability challenge in CM emerges: stakeholder systems may not support the same DPP representation. For example, one factory may operate a DPP system aligned with the AAS-based approach, while other stakeholders' systems work only with ontology-based DPPs. This divergence leads to a strong need to harmonize the two approaches to achieve an interoperable solution. Unfortunately, there is still a lack of contributions on this subject. Two found related works, referenced in (Rimaz et al. 2024) and (Kuiper et al. 2026), provide methods and tools for fully or partially converting DPP-represented in AAS into ontology-based formats. The first also proposes an additional approach for backward conversion. Both approaches are heavily dependent on the AAS metamodel, do not propose an independent ontology for DPPs, and focus primarily on format translation rather than on DPP semantics. As a result, they are less suitable for ontology-driven modeling and may pose challenges for ontologists. To address the gap, this paper proposes a bidirectional conversion method between AAS-based and ontology-based DPPs, based on a newly developed ontology called the Product Digital Twin Ontology (PDTO). An additional contribution of this research is a methodology that supports applying the PDTO and the conversion method to a domain-specific sector, such as furniture.

The rest of this paper is organized as follows. Section 2 presents the background of this research, including the modeling and data representation approaches of the AAS standard and ontologies. Section 3 focuses on the main contributions, and Section 4 demonstrates the feasibility of these contributions through an application in the furniture sector. Next, Section 5 discusses the implications and key findings. Finally, a brief conclusion sums up the paper and outlines future work.

[1] https://catena-x.net/
[2] https://dacapo-project.eu/
[3] https://raasceman.eu/
[4] https://eclass.eu/en/
[5] https://cirpassproject.eu/
[6] https://cirpass2.eu/
[7] https://www.globalbattery.org/

2. Background

The principles of AAS information model and ontological modeling are the cornerstone for understanding this research. The following two subsections respectively detail them.

2.1. AAS-Based Modeling

Asset Administration Shell (AAS) is a standard for deploying industrial-grade Digital Twins (DTs). The standard is developed and maintained by the Industrial Digital Twin Association[8] (IDTA) and is widely recognized by the industrial community. Among its five core specifications, Part 1, which focuses on the metamodel of AAS information models (Industrial Digital Twin Association 2023), is the subject of this research.

Figure 2 illustrates an excerpt of the AAS metamodel represented using Unified Modeling Language (UML). The **AssetAdministrationShell** class contains exactly an instance of the **AssetInformation**, indicating that an AAS DT is associated with exactly one asset. It is worth noting that an asset in manufacturing can be a resource, product, or process involved in production. The information used to describe an asset can be specified using several attributes: `assetKind` indicates whether the asset is a type or an instance, and `assetType` shows its referenced type; `specificAssetId` and `globalAssetId` provide identifiers for the asset, one at a local scale and another at the global cross-system scale; `defaultThumbnail` is useful for containing a small representative image of the asset. The **AssetAdministrationShell** class has a `derivedFrom` attribute that can reference itself, indicating that an AAS instance can derive from an AAS template. Also, this class comprises one or multiple instances of the **Submodel** class, which in turn comprises one or several instances of the **SubmodelElement** class. The **SubmodelElement** class is abstract and is the superclass of multiple classes, including **DataElement**, **EventElement**, **Operation**, **Capability**, **Entity**, **RelationshipElement**, **SubmodelElementCollection**, and **SubmodelElementList**. The latter two are used to organize **SubmodelElement** instances into groups; therefore, they have a composition relationship pointed back to the **SubmodelElement** class. While a **SubmodelElementCollection** instance comprises unordered elements, a **SubmodelElementList** instance contains elements of the same type in an ordered manner. **DataElement** is also an abstract class with six subclasses: **Property**, **File**, **Blob**, **ReferenceElement**, **MultiLanguageProperty**, and **Range**. These subclasses define attributes for expressing data values, whose representations depend on the specific class type. In detail, the value of a **Property** instance can be a primitive type, such as an integer, a double, or a string; the value of a **File** instance contains the path to access a file; the value of a **Blob** instance is a binary content, such as a serialization of a file; the value of a **ReferenceElement** is a reference to another AAS element or an external resource; the value of a **MultiLanguageProperty** instance can contain multiple texts specified in different languages; the value of a **Range** instance represent a range of value using the minimal and maximal thresholds. It is worth noting that the classes **AssetAdministrationShell**, **Submodel**, and **SubmodelElement** inherit from some abstract classes of **Identifiable**, **HasDataSpecification**, **HasKind**, **HasSemantics**, **Qualifiable**, and **Referable**. They respectively indicate that an instance must have an identifier, relate to a specification, be classified as either a type or an instance, can include a `semanticId` referencing the source that clarifies its meaning, support qualification, and include information that enables it to be understood and referenced. The `semanticId` is normally used to reference an external source, such as a concept defined in a vocabulary like ECLASS or an ontology. Moreover, each AAS element has an `idShort`, which serves as a short, local and human-understandable identifier.

With the AAS metamodel, AAS engineers can design and implement the AAS information model for an AAS DT. In the market, there are two remarkable tools for this implementation. The first is the AASX Package Explorer[9] developed by IDTA, which supports creating AAS information models in JSON or AASX formats. The second is Papyrus4Manufacturing[10], developed by CEA List, which provides a framework for designing AAS information models using UML and generating the design in XML format. The AAS community encourages reusing existing AAS submodel templates (if they exist) rather than designing AAS information models from scratch. About 100 AAS submodel templates created by domain experts and validated by IDTA are available on the IDTA Hub[11]. Also, other project consortia and working groups tend to create and host their own AAS submodel templates for their use cases (Nguyen et al. 2025). Furthermore, the semantics of the AAS information model can be enhanced by using the mechanism of referencing AAS elements to the globally semantic identifier of a reputable vocabulary, ontology or standard, such as ECLASS.

An AAS-based DPP has an AAS information model in which information is organized into submodels corresponding to different aspects of the DPP. For example, the information of a DPP covering two aspects of product characteristics and bill of materials can be represented by an AAS information model containing two submodels **Characteristics** and **BillOfMaterials**. The detailed information for each aspect is further modeled using **SubmodelElement**. In practice, the DPP primarily relies on **DataElement** (e.g. **Property**, **File**, etc.), **SubmodelElementCollection**, and **SubmodelElementList**, while others are rarely used.

Engineering an AAS-based DPP follows the above-mentioned principles. In detail, DPP engineers identify existing AAS submodel templates and combine them to form a complete AAS information model template for a given product type. The information generated during the creation of product instances is then used to populate the AAS information model template, producing AAS information model instances (AAS-based DPP instances). For example, an AAS information model template containing a **Characteristics** submodel with a `Color` prop-

[8] https://industrialdigitaltwin.org

[9] https://projects.eclipse.org/projects/dt.aaspe

[10] https://eclipse.dev/papyrus/components/manufacturing/

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

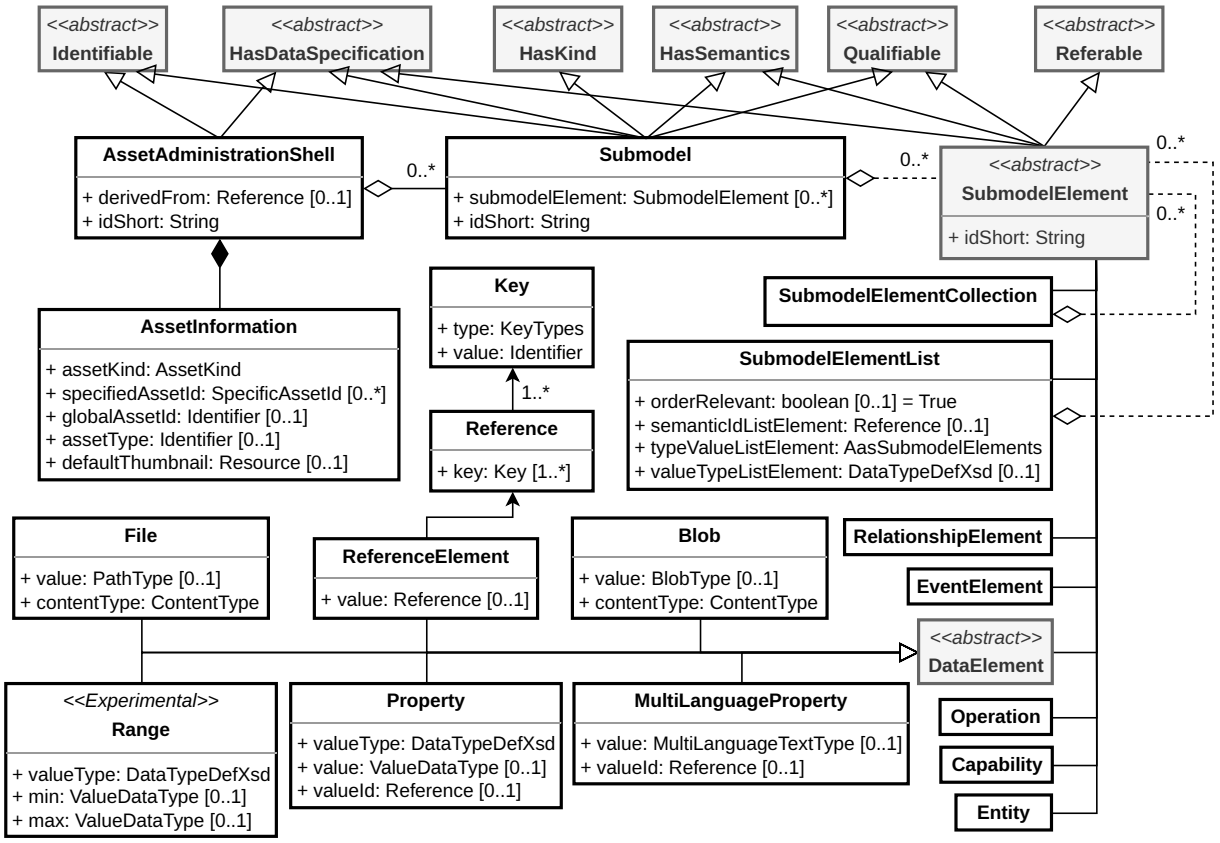

Figure 2 Excerpt from the AAS metamodel in UML notion

erty can be adopted for the sofa production in a factory. The value of the `Color` property is left empty by default. The production results in two product instances: the first sofa is beige, and the second sofa is red. Each sofa has a corresponding DPP instance. The DPP instance of the first sofa contains a **`Characteristics`** submodel with a `Color` property assigned with the string *beige*. The DPP instance of the second sofa also contains a **`Characteristics`** submodel with a `Color` property, but this value is assigned the string *red*.

2.2. Ontology-Based Modeling

An ontology is commonly defined as *"an explicit specification of a conceptualization"* (Gruber 1993). In knowledge engineering, this definition has been refined to emphasize that ontologies are *formal*, *shared*, and *machine-interpretable* representations of domain knowledge (Studer et al. 1998). An ontology typically defines a set of concepts (classes), relationships (properties), constraints, and axioms that collectively describe a domain in a structured and consistent manner.

By providing formal semantics grounded in logic, ontologies are particularly valuable in cross-sectoral and multidisciplinary domains, where stakeholders may employ different terminologies or conceptual frameworks (Guarino et al. 2009). Formally, Description Logic (DL) provides the formal semantics in which knowledge is represented using a **`TBox`** (terminological component) and data is represented using an **`ABox`** (assertional component). Practically, ontology-based systems are typically implemented using W3C Semantic Web standards, primarily OWL[12] (Web Ontology Language), RDF[13] (Resource Description Framework), and SPARQL[14] (SPARQL Protocol and RDF Query Language).

OWL is a formal ontology language grounded in DL. It enables the definition of knowledge under the **`TBox`** which defines:

- class hierarchies e.g., $\mathit{Furniture} \sqsubseteq \mathit{Product}$ indicating that the class **`Furniture`** is a subclass of **`Product`**;
- property characteristics e.g., transitive, functional;
- cardinality constraints e.g., $\mathit{Furniture} \sqsubseteq \geq 1\ \mathit{hasMaterial}.\mathit{Material}$ indicating that every instance of the class **`Furniture`** must be associated via the object property `hasMaterial` with at least one instance of the class **`Material`**, and $\mathit{Furniture} \sqsubseteq \geq 1\ \mathit{hasColor}.\mathit{xsd}{:}\mathit{string}$ indicating that the instance is also associated with one literal value of type xsd:string via the datatype property `hasColor`;
- and logical equivalence and disjointness axioms.

There are formal axioms that allow reasoning engines to infer implicit knowledge, validate consistency, and classify instances. They provide the schema (skeleton) of the ontology.

RDF enables representing the instance-level assertions in **`ABox`**. An RDF triplet (s, p, o) is a data entry where s (subject)

[12] https://www.w3.org/TR/owl2-overview/
[13] https://www.w3.org/TR/rdf11-concepts/
[14] https://www.w3.org/TR/sparql11-query/

denotes a resource, p (predicate) a property, and o (object) either another resource or a literal value. For example, the DPP from the above example can be described with ABox assertions

$$Furniture(f1)$$
$$hasMaterial(f1, fabric)$$
$$hasColor(f1, "Beige")$$

may be expressed in RDF as:

```
:f1 rdf:type :Furniture .
:f1 :hasMaterial :Fabric .
:f1 :hasColor "Beige"^^xsd:string .
```

These triples collectively form a directed labeled graph, where nodes represent entities and edges represent relationships as demonstrated in Figure 3, resembling the data graph which is compliant to schema of the ontology.

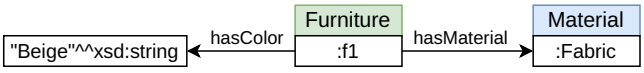

Figure 3 Example RDF graph fragment representing a furniture DPP

However, representing DPP data in RDF does not by itself ensure semantic interoperability. The serialization format in RDF only structures data syntactically while semantic meaning arises from the underlying ontology that defines the shared vocabulary, class hierarchies, properties, and constraints. In this sense, the RDF data graph (`ABox`) must conform to a formally defined ontology (`TBox`) that provides the conceptual schema of the domain.

In summary, ontological modeling languages such as RDF, OWL, and SPARQL provide engineers with a formally grounded and standards-based framework for constructing interoperable knowledge models. Ontology engineers can use modeling environments such as Protégé[15] to define classes, properties, and logical constraints in OWL, serialize the resulting ontology in RDF, validate data using reasoners or constraint languages such as Shapes Constraint Language (SHACL)[16], and query it through SPARQL endpoints.

Beyond their formal foundations and engineering capabilities, ontology-based modeling offers modularity and the ability to reuse and import existing vocabularies and domain ontologies. This capability is especially critical in the context of DPP modeling due to the inherently heterogeneous, multi-domain, and cross-sectoral nature of a DPP. A DPP aggregates knowledge originating from diverse domains, including product design and manufacturing, supply chain logistics, environmental and lifecycle assessment, substances of concern, regulatory compliance, recycling and circularity metrics, and potentially statistical or certification data.

Ontology engineers can therefore integrate multiple semantic resources within a modeled DPP rather than redefining concepts from scratch. In the CIRPASS project, several existing ontologies have been identified and analyzed for potential inclusion in the subsequent CIRPASS-2 project (Bernier & Danash 2024). These include the GS1 Global Data Model vocabularies[17] for product identification and the PRoduct ONTOlogy (PRONTO) (Vegetti et al. 2011) for product description, Schema.org[18] for general web semantics, and ISO-aligned terminologies such as ISO 14040 and ISO 14044 (International Organization for Standardization 2006a,b). This is in relation to other initiatives demonstrating the reuse of ontologies across projects. For example, the FEDeRATED[19] project contributes semantic interoperability recommendations to the European Commission Directorate-General for Mobility and Transport (DG MOVE), and the earlier ontology engineering efforts in the TOVE (Toronto Virtual Enterprise) project (Fox 1992).

3. Contributions

The conversion between AAS-based and ontology-based DPPs is not straightforward due to three major constraints. First, as the AAS-based approach, whose schema and vocabulary are already concretely defined and widely accepted by the community, the ontology-based approach also needs a well-defined ontology to serve as its schema. It is worth recalling that a DPP represented in RDF or OWL has no semantic value unless there is a concrete ontology serving as the schema skeleton. Second, even when a suitable ontology exists, conversion remains impossible if its schema is not aligned with the AAS-based DPP structure style. Third, data-type mapping and the heterogeneity of information structures must be addressed to ensure that information can be represented correctly in both approaches.

Subsection 3.1 introduces a new ontology for DPPs that addresses the first and second constraints. Subsection 3.2 details the conversion process with respect to the second and third constraints. Moreover, Subsection 3.3 presents a methodology that supports applying these solutions in a domain-specific context.

3.1. Product Digital Twin Ontology

The Product Digital Twin Ontology (PDTO) is an ontology for representing a product digital twin (PDT). A PDT can be defined as the *"collection of all information related to a product, including ones of the Digital Product Passport"* (Christoph Attila Ku 2024). In other words, the information of a PDT encompasses the information of a DPP, and a DPP can be considered a specialized kind of PDT. These important statements allow ontology engineers in this research to broaden the scope of the search and reuse elements from ontologies related not only to DPP and products, but also to PDT. The concepts from the following ontologies are imported into PDTO.

- The Web-of-Thing Digital Twin Ontology (WoTDT or DTW) is an ontology extending the W3C Thing Description ontology to represent digital twins (DT) and their different dimensions (González-Gerpe et al. 2024). It offers a well-structured framework for representing different types of DTs using different model types.
- The DPP ontology network proposed by (Jansen et al. 2024) comprises five ontological modules, each describing

[15] https://protege.stanford.edu/
[16] https://www.w3.org/TR/shacl/
[17] https://ref.gs1.org/voc/
[18] https://schema.org
[19] https://www.federatedplatforms.eu/

a distinct category of information relevant to DPPs. Its DPP Information (DPP-INFO) module is useful in detailing the element-level information contained in a DPP.
- The GoodRelations (GR) is a well-known ontology for products and e-commerce information (Hepp 2008). It provides a lightweight but expressive vocabulary for product descriptions in a widely accepted manner.
- The Semantic Sensor Network (SSN) ontology and its modular extension Sensors, Observations, Sampling, and Actuation (SOSA) are highly credible ontologies in observing real-world phenomena (Haller et al. 2018). In particular, their concepts of features of interest and properties can be used to improve the description of an object.

Figure 4 illustrates the main components of the PDTO. The concepts for the physical entity (**dtw:PhysicalEntity**) and the digital entity (**dtw:DigitalEntity**), which respectively represent the physical and digital perspectives related to a digital twin, are reused from the DTW ontology. The concept of a product in the GR ontology can be considered the subconcept of a physical asset (as a product is also a physical asset); thus, the class gr:Product can be declared as a subclass of **dtw:PhysicalEntity**. The three datatype properties `gr:name`, `gr:description`, and `schema:productID` associated with **gr:Product** are reused to describe the details of a product. In particular, `schema:productID` has several subproperties, such as `gr:hasGTIN-8`, `gr:hasGTIN-14`, and `gr:hasEAN_UCC-13`, which are useful for identifying a product in different identifier formats. Moreover, the object property `pdto:thumbnail` is added to the PDTO for **gr:Product** to link it to the **schema:ImageObject** class. This addition represents the fact that a product can have a small thumbnail image.

Concerning the digital perspective, the DTW ontology is compatible with (Frédéric Sanchez 2023), in representing the fact that a digital twin can be associated with multiple types of models. Also in DTW, several model types are already predefined as subclasses of **dtw:Model**, such as **dtw:GeometricModel** and **dtw:PhysicalModel**. The **pdto:InformationModel** class is newly introduced in the PDTO as a subclass of **dtw:Model** to describe the type of model dedicated to information. This class has the object property `pdto:hasInformation` and the datatype property `pdto:hasSimpleInformation`. While the former links to an instance of the **dpp-info:DPPInformation** class to present more detailed information, the latter provides a mechanism to simplify the information content in plain text. The class **pdto:AspectModel** is a subclass of **pdto:InformationModel** and, in turn, has several subclasses: **pdto:Nameplate**, **pdto:QualityModel**, **pdto:InstructionModel**, and **pdto:RegulationModel**. While **pdto:AspectModel** is used to represent the AAS submodel concept, its subclasses are used for classifying submodels more clearly and effectively. For example, AAS submodels describing repair instructions and disassembly instructions should be represented as two instances of **pdto:InstructionModel**, whereas a submodel concerning durability should be represented as an instance of **pdto:QualityModel**. The object property `dtw:modelAggregates` is used to represent the fact that an instance of **pdto:InformationModel** comprises multiple instances of **pdto:AspectModel**, similar to how an AAS information model consists of multiple AAS submodels.

Some concepts from the DPP-INFO are reused with appropriate modifications and additions to represent different types of information. In detail, the classes **pdto:AtomicInformation** and **pdto:CompositeInformation** are introduced in the PDTO as subclasses of **dpp-info:DPPInformation**. While the former can only describe discrete qualitative and quantitative information, the latter can represent a group of complex information that includes smaller components. The class **pdto:AtomicInformation** has two datatype properties, `dpp-info:value` and `pdto:hasSimpleValue`, to represent the value of information in a simple manner. Moreover, it has two object properties `pdto:hasFeatureOfInterest` and `pdto:clarifiedProperty`, which respectively link to the classes **sosa:FeatureOfInterest** and **ssn:Property**. This design enables specifying which product the information relates to and which property of that product it describes. The two classes **dpp-info:ProductCharacteristics** and **dpp-info:ProductQuality** are reorganized as subclasses of **pdto:AtomicInformation**. The former is used to present quantitative information, and the latter serves to present qualitative information. Their object property `dpp-info:unit` is also reused in PDTO to specify the value's unit (if it exists).

The class **pdto:CompositeInformation** can include other DPP information using the object property `pdto:hasInformationComponent`. Moreover, its subclasses are newly defined to better support information classification. In detail, the **pdto:MaterialInformation** class is introduced as a subclass of **dpp-info:CompositionInformation** (which itself is a subclass of the class **pdto:CompositeInformation**) to represent the components of a product, such as a material or intermediate product; the **pdto:InstructionInformation** class is introduced to represent different kinds of instructions, such as repair and disassembly instructions; the **pdto:IndicatorInformation** class is introduced to represent different kinds of indicators, such as climate change and particulate matter impacts in the Product Environmental Footprint (PEF) defined by the EC. Moreover, the **dpp-info:CertificationInformation** class is reused and reorganized as a subclass of **pdto:CompositeInformation** to describe certificates. The commonality of the above subclasses is that they are specified with a name and a description through the datatype properties `schema:name` and `schema:description`. However, they have own specific details. The class **pdto:MaterialInformation** has the datatype property `pdto:isMandatory` to specify whether the material is mandatory for the product, and the object property **pdto:hasProductReference** to link the material information to its original entity. The class **dpp-info:CertificateInformation** has the datatype property `pdto:hasIssuedDate` to indicate the date the certificate was issued and uses the property `dpp-info:externalDocument` to reference the digital document (**schema:DigitalDocument**) of the certificate. Likewise, the classes **pdto:InstructionInformation**

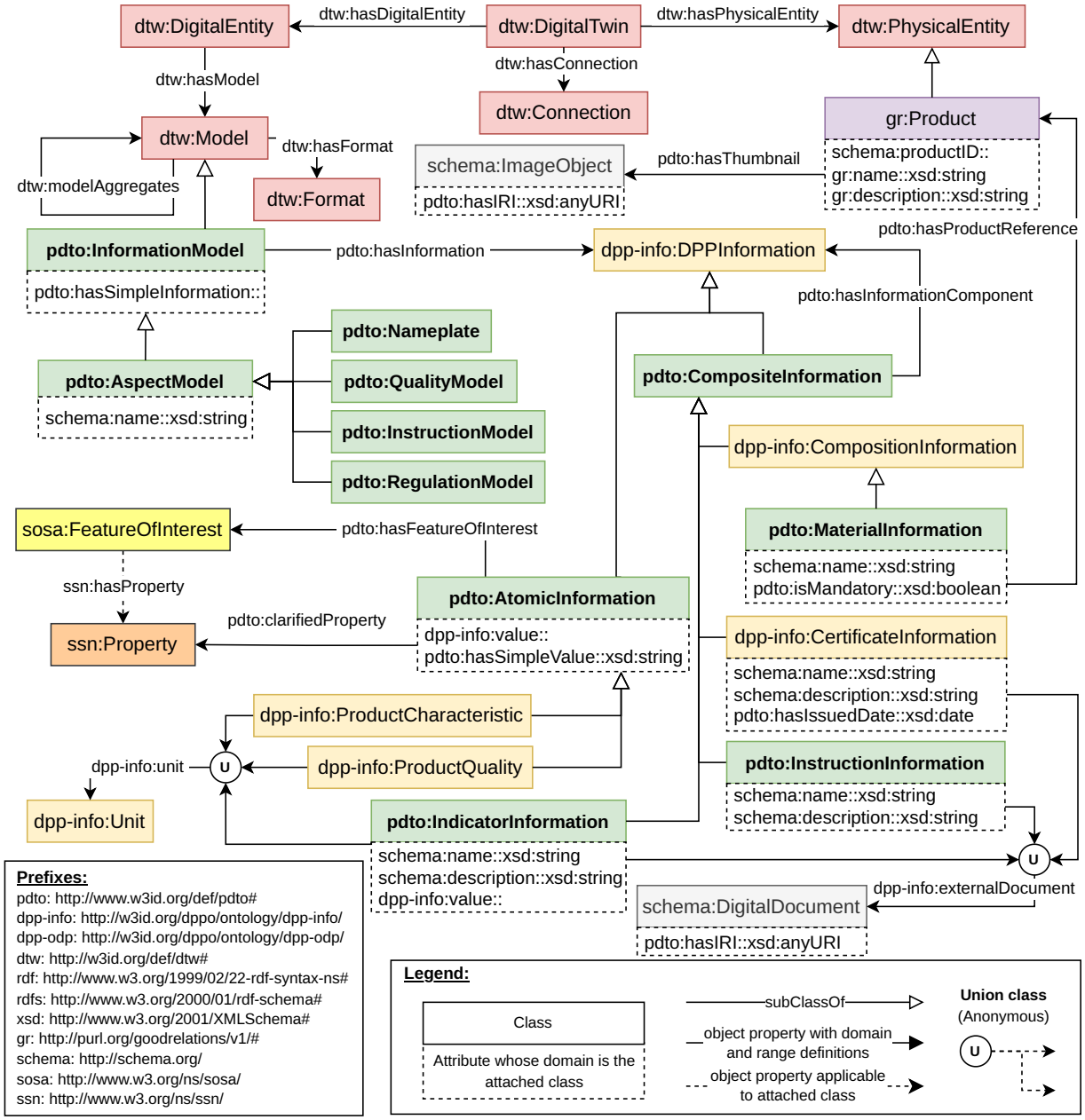

Figure 4 Overview of the Product Digital Twin Ontology (PDTO)

and `pdto:IndicatorInformation` also use the property `dpp-info:externalDocument` to reference their digital documents. Moreover, the class `pdto:IndicatorInformation` uses the properties `dpp-info:value` and `dpp-info:unit` to express the evaluation of an indicator.

The PDTO was designed and implemented using the Protégé tool. Its current version includes 61 classes, 61 object projects, and 25 data properties. The ontology was validated and checked for modeling pitfalls using OOPS! [20] (Poveda-Villalón et al. 2014), with no "critical" or "important" pitfalls detected. It was then documented using the WIDOCO[21] tool and made available on https://w3id.org/pdto.

[20] https://oops.linkeddata.es/
[21] https://dgarijo.github.io/Widoco/

3.2. Mapping Between AAS-Based and Ontology-Based DPPs

The aim is to convert a DPP represented using AAS into a DPP represented using PDTO, and vice versa. An AAS-based DPP uses the AAS schema and is typically encoded in JSON (using the AASX Package Explorer) or UML/XML format (using Papyrus4Manufacturing). An ontology-based DPP uses the PDTO schema and is normally encoded in one of the widely used ontology formats, such as RDF/XML, Turtle, or OWL/XML.

The following are major mapping rules adopted in the conversion. Note that the description prioritizes the direction from AAS to PDTO, but the logic can also be applied in reverse.

(a) `AAS:AssetInformation` is mapped to the `gr:Product`. Thus, the information contained in `AssetID`, `assetType`, and `defaultThumbnail` is converted into the correspond-

ing information related to schema:productID, gr:name, and pdto:hasThumbnail.

(b) **AAS:AssetAdministrationShell** is mapped to the **pdto:InformationModel**.

(c) **AAS:Submodel** is mapped to the **pdto:AspectModel** in general. Moreover, the idShort attribute can be used to further classify an instance as a **pdto:Nameplate**, **pdto:InstructionModel**, **pdto:QualityModel**, or **pdto:RegulationModel**. For example, if the value of idShort corresponds to a quality-related aspect of a product, such as safety or durability, the instance should be classified as **pdto:QualityModel**. If the value of idShort contains the term *"instruction"*, such as a *"repair instruction"*, the instance should be classified as **pdto:InstructionModel**. If the value of idShort relates to a regulation, such as *"health impact"*, the instance should be classified as **pdto:RegulationModel**.

(d) Both AAS and ontology use the XML datatype schema standard[22] to represent data types; thus, data type conversions can be performed naturally.

(e) In DPP design, some of the AAS submodel elements are rarely or never used: **RelationshipElement**, **EventElement**, **Operation**, **Capability**, **Entity**, **Range**, **Blob**. Thus, they are excluded from the mapping.

(f) **AAS:Property** mapped to **dpp-info:ProductQuality** or **dpp-info:ProductCharacteristic** depends partly on its attributes idShort and valueType. When the value of valueType is a string, the instance can be classified as **pdto:AtomicInformation** or **dpp-info:ProductQuality**. It is worth noting that using **pdt:AtomicInformation** is more secure, as it is always correct, even if the instance means qualitative and quantitative data. When the value of valueType is not a string but other types, such as integer, the instance should be considered as a **dpp-info:ProductCharacteristic**. Moreover, when the value of idShort contains the term *"unit"*, the instance must be classified as **dpp-info:Unit** and become the unit of another instance.

(g) An instance of **AAS:File** contains the URL of a file. It is mapped to **dpp-info:ProductCharacteristic**. The PathType value corresponds to the value of dpp-info:value of type xsd:anyURI.

(h) **AAS:MultiLanguageProperty** is mapped to **pdto:AtomicInformation**. The value with type MultiLanguageTextType is actually a LangStringSet that may contain several texts in different languages. It can also be represented by the dpp-info:value of type rdf:LangString.

(i) **AAS:SubmodelELementCollection** is mapped to **pdto:CompositeInformation**. The attribute idShort is used to further classify an instance of its subclasses. In detail, if the value of idShort contains the term *"certificate"*, the instance is classified as **dpp-info:CertificateInformation**. The same logic applies to the terms *"indicator"*, *"material"*, and *"instruction"* for the other three subclasses. Next, the submodel elements of the **AAS:SubmodelElementCollection** are corresponding mapped to the datatype and object properties of the target ontology class.

(j) The mapping of **AAS:ReferenceElement** distinguishes two cases. If the Reference value is used within a material-related **AAS:SubmodelElementCollection** instance, the key.value of the reference is then mapped to the property pdto:hasProductReference. Note that key.value indicates that a **ReferenceElement** contains a **Reference** as its value, which is composed of one or more **Key** objects, each having a value that serves as the string identifier of the referenced element, as illustrated in Figure 2. Otherwise, **AAS:ReferenceElement** is mapped to **dpp-info:ProductCharacteristic**, and the key.value of the contained Reference is mapped to dpp-info:value.

(k) **AAS:SubmodelElementList** can be mapped to **pdto:CompositeInformation**. To distinguish it from an **AAS:SubmodelElementCollection**, its elements typically share the same idShort label with a numeric suffix. For example, a submodel list **IndicatorSet** may contain two submodel collections **Indicator_01** and **Indicator_02** as elements. However, to reduce complexity, the list may also be omitted from the ontology data. When converting from PDTO back to AAS, the mentioned rule can still allow the submodel list to be reconstructed in the AAS information model.

(l) The mapping rules (c), (f), and (i) rely on idShort string heuristics, which are not solid enough to cover all cases. For example, an AAS submodel instance whose idShort contains the term *"repair guideline"* should be classified as **pdto:InstructionModel**, but is actually assigned to **pdto:AspectModel** because rule (c) only detects the keyword *"instruction"*. An approach to overcome this limitation is to use the semanticId to reference the exact concept in PDTO. Returning to the example of the AAS Submodel instance representing a repair guideline, it can include a semanticId referencing http://www.w3id.org/pdto#InstructionModel. This semantic information can then be used to explicitly assign the instance to **pdto:InstructionModel**.

To implement the conversion program for this model transformation, a canonical model should be designed based on the PDTO schema using a programming language, such as Python. Thus, each PDTO class and its properties should be formalized as a Python dataclass. This model acts as an intermediate layer between the source and target models. For example, an AAS-based DPP in JSON format is transformed into the canonical model, which subsequently converts the data into a PDTO-compliant representation in OWL format. In this approach, the PDTO serves as the conceptual pivot, while the canonical model functions as the operational pivot for the conversion.

3.3. Methodology for Applying PDTO in a Domain-Specific Context

While the PDTO provides a general schema, it should be customized for specific sectors as they imply several differences.

[22] https://www.w3.org/TR/xmlschema-2/

For example, automotive feature interests include properties such as speed, which have no equivalent in furniture. Likewise, the Maintenance model (corresponding to the maintenance aspect) is essential in automotive but less common in furniture.

Figure 5 illustrates the methodology that combines the mini-Waterfall and Linked Open Terms[23] (LOT) for customizing the PDTO and the conversion tool for a domain-specific sector. Literally, the system's validation results in the Testing phase can demonstrate the correctness of the ontology.

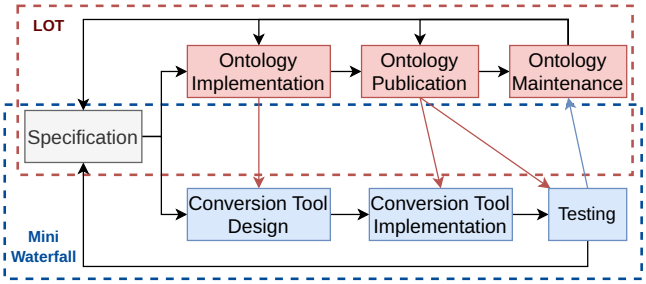

Figure 5 Mini-Waterfall combined with Linked Open Terms

In the Specification phase, engineers must analyse the requirements and identify existing ontologies relevant to the domain. In the LOT flow, based on this specification, the engineers create a domain-specific ontology that imports the PDTO and other appropriate ontologies. In particular, the sector's products and their properties should be newly declared or reused from the finding ontologies. Subclasses of `pdto:AspectModel` and `pdto:CompositeInformation` should also be extended according to the analysed requirements. As PDTO, the domain-specific ontology should also be validated and checked for pitfalls using the OOPS! tool. Next, it should be published on the project's website or through another ontology management service, such as https://www.semantic-treehouse.nl, to enable downloading and sharing. Moreover, a contact platform should be deployed to collect feedback and support future improvements during the Ontology Maintenance phase.

The Mini-Waterfall flow is for the conversion tool development. As the tool's canonical model relies on the PDTO, one of its inputs is the ontological schema from the Ontology Implementation phase. The second input is the expected conversion formats (e.g., JSON or OWL) from the specification document. Implementing a format generally requires two modules: one to convert the considered information model into the canonical model, and another to convert the canonical model back into the information model. For example, converting between an AAS-based DPP in JSON and an ontology-based DPP in OWL requires four modules: (1) JSON to canonical model, (2) canonical model to OWL, (3) OWL to canonical model, and (4) canonical model to JSON. The Conversion Tool Implementation and Testing phases need to import the ontology from the ontology's published site; thus, they depend on the outputs of the Ontology Publication phase. The results of the Testing phase is used to verify not only whether the conversion tool is correctly implemented, but also whether the ontology is well designed.

[23] https://lot.linkeddata.es/

4. Case Study: DPP for the Furniture Sector

The Cir4Fun[24] project is an initiative aimed at advancing the furniture industry by integrating circular economy principles throughout the entire product lifecycle. Its consortium comprises 22 partners from both the public and private sectors across 10 European countries. These partners have different expertise and knowledge in the CM; thus, can effectively contribute their experiences to the project. For example, Pozzi Arturo Spa is a fabric provider, Kave Home is a furniture designer and retailer, and ENDIA is a waste management and recycling party.

The Cir4Fun project has several specific requirements for DPPs tailored to the furniture sector, aiming to make them a central tool for circularity and sustainability. In which, two fundamental requirements are (1) to ensure that DPPs are aligned with ESPR and existing furniture ontologies and vocabularies (e.g., funStep[25]), and (2) to provide useful and trustworthy information to manufacturers, retailers, consumers, and repair/refurbishment/remanufacturing centers. One of the major tasks of the project is to develop a method to represent DPPs for furniture products. The solution must be flexible and interoperable enough to support diverse stakeholders not only within the current project, but also across the wider furniture sector. Supporting both AAS-based and ontology-based approaches is a beneficial choice, but it raises the need to convert DPPs produced in one approach into the other.

At an early stage of the project, we identified 25 AAS submodel templates from several sources: obtained from IDTA, extracted from ESPR, and derived from specific stakeholder requirements. They are used to construct the AAS information model for AAS-based DPPs. To recall, an AAS information model includes only those relevant to its specific needs.

The methodology presented in Section 3.3 is adopted to develop an ontology dedicated to DPPs in the furniture sector, and the conversion tool harmonizes AAS-based and ontology-based DPPs. The ontology is called DPP4Fun, which stands for DPP for Furniture. In the Specification phase, ontology engineers reviewed existing ontologies in the furniture sector, then decided to reuse concepts from the furniture sector ontology[26] (FSO). The reused vocabularies are mostly the `fso:Product` class and its subclasses, such as `fso:Funiture` and `fso:Components`. FSO also contains several predefined instances useful to richen the DPP4Fun. In the Ontology Implementation phase, more concepts are added to the PDTO as illustrated in Figure 6. In detail, `dpp4fun:DigitalPassportProduct` is introduced as a subclass of `dtw:DigitalTwin` to clarify that its instances represent DPPs. The class `fso:Product` and its subclasses are organized as subclasses of `sosa:FeatureOfInterest`. The classes `dpp4fun:QuantitativeProperty` and `dpp4fun:QualitativeProperty` are newly declared as subclasses of `ssn:Property`. Note that this small extension clarifies the classification of a `pdto:AtomicInformation` instance as `dpp-info:ProductCharacteristics` or

[24] https://cir4fun.eu
[25] http://www.funstep.org
[26] http://auxproyti.aidimme.es/FurnitureSectorOntology/

dpp-info:ProductQuality, thus improving the mapping rule for **AAS:Property** presented in Section 3.2. Several subclasses are added to extend the **pdto:AspectModel** class, based on the requirements declared by the project stakeholders. For example, **dpp4fun:RecyclingInstruction** and **dpp4fun:DisposalInstruction** are introduced as subclasses of **pdto:InstructionModel** to represent the recycling and waste management instructions. Moreover, a set of instances of **fso:Product** (and its subclasses) and **ssn:Property** (and its subclasses) is added to DPP4Fun. These instances are derived from the products (e.g., KaveHome Gala 3-Seater Beige) and properties (e.g., color) proposed by the stakeholders. The DPP4Fun ontology was implemented using Protégé. It imports PDTO and FSO, and extends them with 35 classes, 7 object properties, and 4 datatype properties. DPP4Fun was validated and checked for pitfalls using OOPS!, and no "critical" or "important" pitfalls were detected. Then, its website resources were generated using WIDOCO.

This project requires the conversion tool to support three formats: AAS-based JSON, AAS-based YAML, and DPP4Fun-based OWL. This requirement arises from the fact that within the development teams, only one member is familiar with three formats, while the others work with either AAS-based JSON or the ontology. Additionally, the default DPP viewer only supports YAML. Thus, the conversion tool requires five modules: two for bidirectional conversion between AAS-based JSON and the canonical model, two for bidirectional conversion between DPP4Fun-based OWL and the canonical model, and one for converting the canonical model to AAS-based YAML. Note that the latter is a one-way conversion, since the DPP viewer is only for DPP reading; thus, the reverse conversion from YAML back to the canonical model is unnecessary. Moreover, additional detailed code needed to be implemented. For example, the `idShort` of an **AAS:SubmodelElementCollection** instance must be compared against more terms to ensure instances are classified correctly, as the number of **pdto:AspectModel** subclasses has increased with new sector-specific extensions.

Figure 7 and Figure 8 illustrate the information models for both AAS-Based and DPP4Fun-based DPPs of a product instance in a simple prototype testing. The selected product is a beige, three-seater sofa from the Gala series by Kave Home. Some information of the product instance is used to make its DPP, which are height, height unit, color, and the set of information about its fabric material. These attributes are selected because they represent the typical information included in a DPP. The conversion process was tested in both directions.

The AAS-based DPP, as illustrated in Figure 7, follows an AAS information model with two submodels: **Characteristics** and **Bill of Materials**. The first submodel contains three properties: `Height`, `HeightUnit`, and `Color`. The second contains the submodel element list `Material Set`, which in turn includes a submodel element collection `Material_04`. This collection contains the basic properties `Name` and `Mandatory`, as well as a `ReferenceElement` that points to another product, `Fortezza 9041`, which is fabric material. Note that in the figure, the relationship with the stereotype «ReferenceElement» is simplified to improve readability. The AAS information model also includes asset information for the **Gala 3-Seater** product. It has a link to a resource containing the URL of the sofa's thumbnail image.

The DPP4Fun-based DPP, as shown in Figure 8, is associated to the instance of **gr:Product** corresponding to the sofa instance. The **pdto:InformationModel** of the DPP is aggregated from a **pdto:AspectModel** with `schema:name` *Characteristics* and from a **dpp4fun:BillOfMaterials** with `schema:name` *Bill Of Materials*. The first model links to two information instances using `pdto:hasInformation`. One is an instance of the class **dpp-info:ProductCharacteristics**, the other is of the class **dpp-info:ProductQuality**. Both instances link to the instance *dpp4fun:KaveHome_Gala_3_Seater_Beige* of the class **fso:Canapé**. The former represents the quantitative value of height and its unit; thus it links to the instance *dpp4fun:height* using `pdto:clarifiedProperty`. The latter represents the qualitative value of color, linking to the instance *dpp4fun:color* using `pdto:clarifiedProperty`. The instance of **dpp4fun:BillOfMaterials** has complex information which is an instance of the **pdto:MaterialInformation** class. This information is linked to the instance of **gr:Product** representing the fabric material Fortezza 9041 using the object property `pdto:hasProductReference`.

In the testing phase, 11 DPP instances were evaluated. Stakeholders provided information about the 11 products, including both final and intermediate ones. An AAS engineer populated such information into 11 AAS-based DPPs in JSON format. The Python conversion tool then converted all of them into two formats: DPP4Fun-based OWL and AAS-based YAML. The conversion was performed automatically using a single run command. Table 1 reports the conversion time of the 11 DPP files with a 100% success rate. Note that each DPP filename consists of a product type and a local identifier. The total time required to convert all 11 files from JSON to OWL is relatively small (0.51 seconds), and the conversion from JSON to YAML is about 3.4 times faster (0.15 seconds). This is expected, as the YAML format is much simpler than OWL. Similarly, the peak memory usage in the case of JSON to YAML conversion (0.41 MB) is about 10 times lower than that of JSON to OWL (4.59 MB). While the 11 files in YAML were loaded into the DPP viewer for stakeholders to perform manual visual evaluations, the 11 files in OWL were evaluated by ontology engineers of the development team. Feedback from both evaluations was sent back to the engineers to improve the tool. One technique applied by the ontology engineers is the use of SHACL. They developed a Python program using the pySHACL[27] library as the validation engine. They also implemented more than 16 SHACL shapes comprising 36 rules that each OWL instance file must satisfy. Each shape corresponds to a class and includes one or more rules corresponding to its properties. Most of the selected shapes are involved in the conversion process described in Section 3.2. When an OWL file failed validation, the debug message was analyzed. The issue could originate either from the conversion tool or from an error in the input AAS-based DPP. The team then fixed the issues and repeated the validation.

[27] https://github.com/rdflib/pyshacl

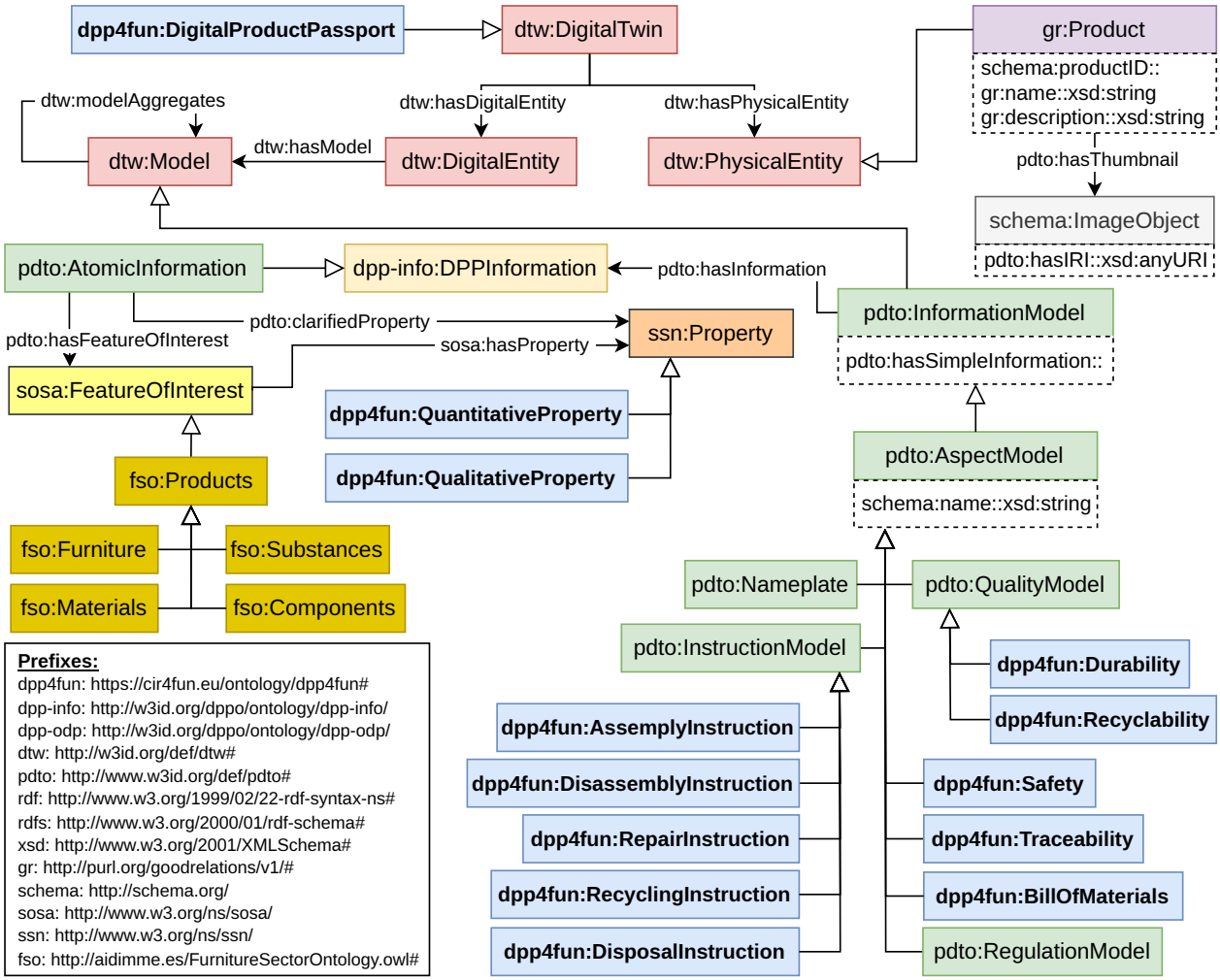

Figure 6 Overview of a part of the Digital Product Passport for Furniture (DPP4Fun) ontology

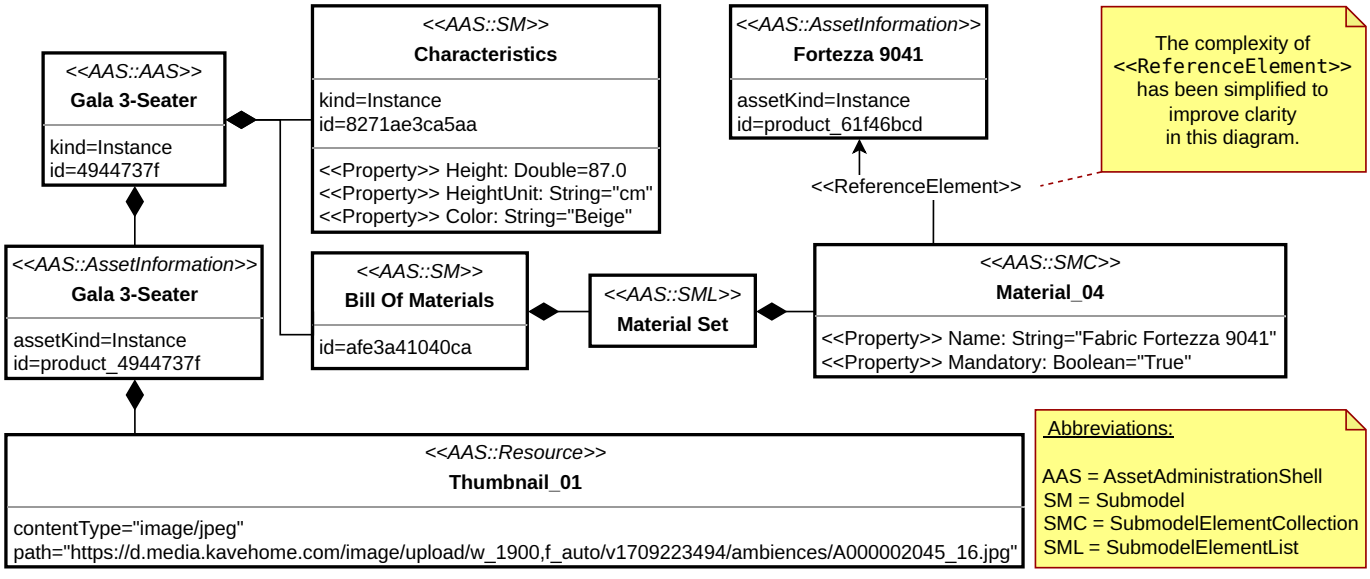

Figure 7 Excerpt of the Gala 3-Seater DPP modeled using the AAS-based approach

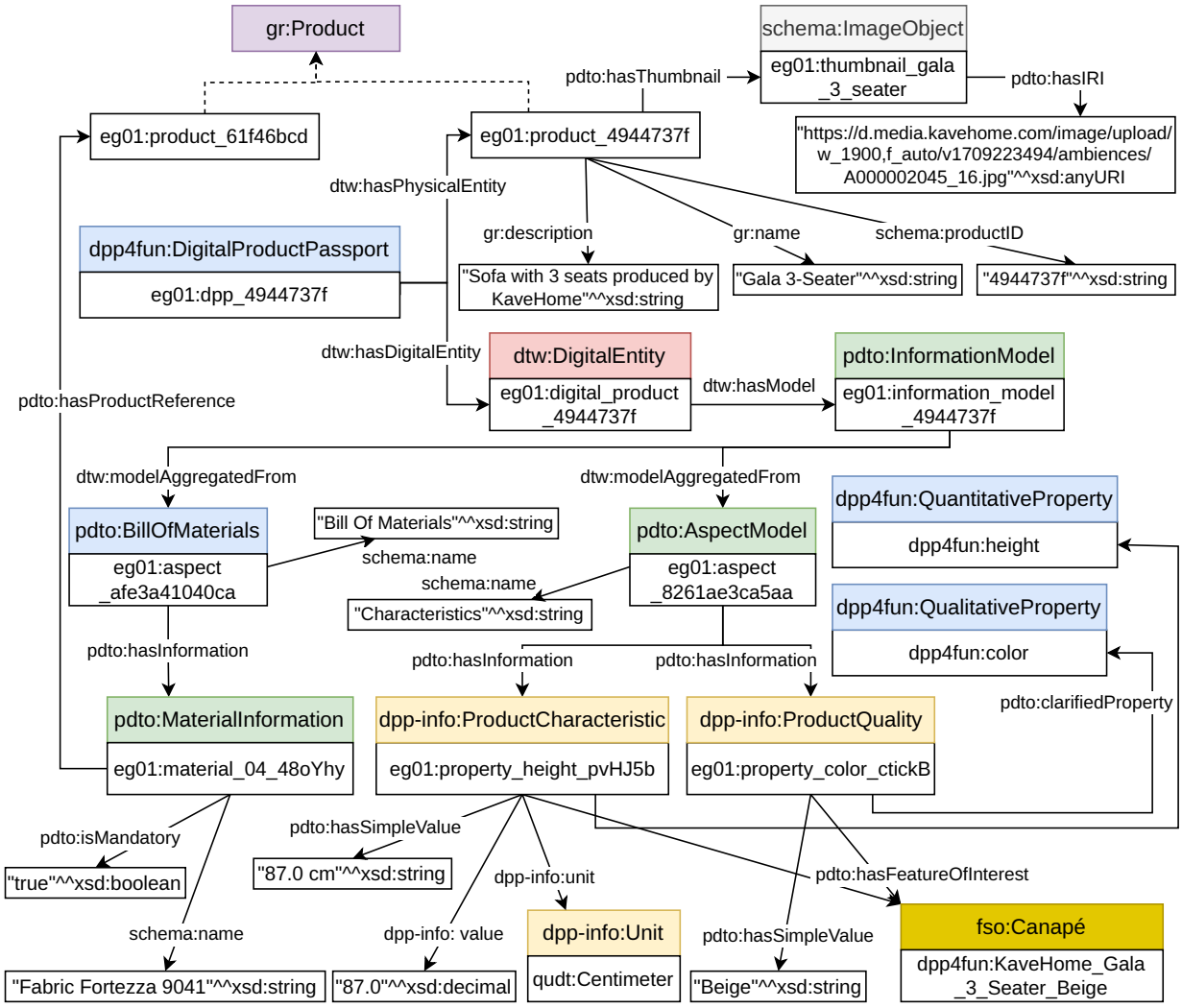

Figure 8 Excerpt of the Gala 3-Seater DPP modeled using the ontology-based approach

Table 1 DPP conversion time (seconds) from JSON to OWL and JSON to YAML

#	DPP File	JSON to OWL	JSON to YAML
1	Sofa-4944737f.json	0.0657	0.0207
2	Fabric-61f46bcd.json	0.0528	0.0189
3	Chair-e9230c88.json	0.0604	0.0167
4	ChairWithArms-e9230c87.json	0.0651	0.0205
5	ConvertableCrib-0731a231.json	0.0440	0.0136
6	MelamineBoard-5065467.json	0.0202	0.0039
7	BabyHighChair-5ca3a78f.json	0.0449	0.0130
8	FixedTray-a7432b20.json	0.0406	0.0112
9	ChairUpsholstery-62cfa286.json	0.0365	0.0112
10	LegsKit-0c7b5be5.json	0.0375	0.0114
11	Locker-66203ea6.json	0.0459	0.0139

5. Discussion

The results presented in Section 4 demonstrate the feasibility and practicality of the contributions introduced in Section 3. Moreover, the following points can be further discussed.

First, PDTO can be applied not only to the furniture sector but also to other sectors, such as automotive or textiles. By using `sosa:FeatureOfInterest` and `ssn:Property` as the root classes for domain-specific products and their properties, PDTO can be flexibly extended with vocabularies from other contexts. In other words, a domain-specific ontology only needs to import PDTO together with the relevant product vocabulary to form a complete DPP ontology aligned with that domain.

Second, a DPP ontology that imports PDTO can support not only CM applications related to DPP, but also a broader range of production applications. Indeed, since PDTO reuses part of DTW to represent PDTs in manufacturing, it can contribute to ontological reasoning in scenarios where DTW is also used to model other manufacturing assets (i.e., resources or processes). The data from all assets forming a knowledge graph can be reasoned over to derive additional insights for the factory.

Third, the current PDT design is composed of several smaller design choices, some of which could be modified or designed differently. For example, the decision to model `pdto:AspectModel` as a subclass of `pdto:InformationModel` instead of being a subclass of `pdto:CompositeInformation`, may be debated. Indeed, an aspect can also be viewed as a component of an information model. However, the current approach is more advantageous because it allows instances of `pdto:AspectModel` to flexibly aggregate different types of information.

6. Conclusion

In this paper, we have presented an approach to implement interoperability between AAS-based and ontology-based Digital Product Passports specifications. To this end, we have designed the PDTO ontology and defined a mapping between AAS-based models and PDTO-based models. We have demonstrated the feasibility of our approach by defining a specialization of PDTO for the furniture sector and by providing a proof-of-concept in converting an AAS-based DPP for a furniture product in an equivalent ontology-based DPP using our conversion tool.

This research has three limitations. First, the sample product data used for testing is relatively small, making it unclear whether the PDTO and DPP4Fun can fully cover all DPP information across all product types, and in particular across all furniture product types. Second, this research focuses on DPP information representation and does not address other features involved in a DPP system, such as authorization. Third, the implementation of the conversion method is a custom Python program built upon a canonical model, which lacks the formal guarantees as traditional model-transformation languages (e.g., QVT, TGG), thereby limiting the robustness, traceability, and maintainability of the transformation.

Our team is planning some future works. First, the PDTO will be further developed and maintained for at least five years by CEA List to ensure its correctness and ability to accommodate new requirements. Second, the DPP4Fun ontology will be applied in other work packages of the Cir4Fun project, and the feedback collected from these activities will support continuous improvement of both DPP4Fun and PDTO. Third, we also plan to release the official version of the conversion tool and the DPP4Fun ontology at the end of the project.

Acknowledgement

This research has been supported by the European Union's HORIZON Research and Innovation Action Program under the grant agreement No 101182081, the project Cir4Fun.

References

Adu-Duodu, K., Wilson, S., Li, Y., Oladimeji, A., Huraysi, T., Barati, M., … Shah, T. (2025). A circular construction product ontology for end-of-life decision-making. In *Proceedings of the 40th acm/sigapp symposium on applied computing* (p. 1943–1952). New York, NY, USA: Association for Computing Machinery. doi: https://doi.org/10.1145/3672608.3707870

Ann-Carina Tietze, Fabian Kentsch, Dorottya Simon, & Florian Mohr. (2023). *Digital Product Passports as the enabler for the Circular Economy* (White Paper). Germany: Catena-X.

Bernier, C., & Danash, F. (2024). *DPP Prototypes* (Tech. Rep. No. https://doi.org/10.5281/zenodo.11393742). European Union: CIRPASS.

Catena-X Automotive Network e.V. (2023). *CX-0067: Ontology Models to Realize Federated Query in Catena-X.* Retrieved from https://catenax-ev.github.io/docs/standards/CX-0067-OntologyModelsToRealizeFederatedQueryInCatenaX.

Christoph Attila Ku. (2024). *Implementing the EU Digital Product Passport with Digital Data Chain and Asset Administration Shell - Overview* (VCI – Webinar on DPP Standardization). Germany: Digital Data Chain Consortium.

European Parliament and the Council of the European Union. (2024). *Regulation (eu) 2024/1781 of 13 june 2024 establishing a framework for the setting of ecodesign requirements for sustainable products, amending directive (eu) 2020/1828 and regulation (eu) 2023/1542 and repealing directive 2009/125/ec* (Vol. L) (No. 2024/1781).

Fox, M. S. (1992). The tove project towards a common-sense model of the enterprise. In *Industrial and engineering applications of artificial intelligence and expert systems* (pp. 25–34). Berlin, Heidelberg: Springer Berlin Heidelberg.

Frédéric Sanchez. (2023). *Digital Twin: Leveraging the Digital Transformation of the Industry* (Brochure). France: Alliance Industrie du Future.

Gianvincenzi, M., Marconi, M., Mosconi, E. M., & Tola, F. (2024). A standardized data model for the battery passport: Paving the way for sustainable battery management. *Procedia CIRP*, *122*, 103-108. doi: https://doi.org/10.1016/j.procir.2024.01.014

González-Gerpe, S., Poveda-Villalón, M., & García-Castro, R. (2024). DTAG: A Methodology for Aggregating Digital Twins Using the WoTDT Ontology. *Applied Sciences*, *14*(13), 5960. doi: 10.3390/app14135960

Gruber, T. R. (1993). A translation approach to portable ontology specifications. *Knowledge Acquisition*, *5*(2), 199–220. doi: 10.1006/knac.1993.1008

Guarino, N., Oberle, D., & Staab, S. (2009). What is an ontology? In *Handbook on ontologies* (pp. 1–17). Berlin, Heidelberg: Springer. doi: 10.1007/978-3-540-92673-3_0

Haller, A., Janowicz, K., Cox, S. J., Lefrançois, M., Taylor, K., Le Phuoc, D., … Stadler, C. (2018). The modular SSN ontology: A joint W3C and OGC standard specifying the semantics of sensors, observations, sampling, and actuation. *Semantic Web*, *10*(1), 9–32. doi: 10.3233/SW-180320

Hbeich, E., Bernier, C., Hooland, S. V., Robal, T., & Maigre, R. (2026). Achieving Semantic Interoperability for Digital Product Passports. *Procedia Computer Science*, *277*, 2981–2992. doi: 10.1016/j.procs.2026.02.334

Hepp, M. (2008). GoodRelations: An Ontology for Describing Products and Services Offers on the Web. In *Knowledge Engineering: Practice and Patterns* (Vol. 5268, pp. 329–346). Berlin, Heidelberg: Springer Berlin Heidelberg. doi: 10.1007/978-3-540-87696-0_29

Industrial Digital Twin Association. (2023). *Specification of Asset Administration Shell - Part 1: Metamodel Schema* (Specification No. IDTA Number: 01001-3-0). Germany: IDTA.

International Organization for Standardization. (2006a). *Iso 14040:2006 environmental management – life cycle assessment – principles and framework.* https://www.iso.org/standard/37456.html.

International Organization for Standardization. (2006b). *Iso 14044:2006 environmental management – life cycle assessment – requirements and guidelines.* https://www.iso.org/standard/38498.html.

Jansen, M., Blomqvist, E., Keskisärkkä, R., Li, H., Lindecrantz, M., Wannerberg, K., ... Berg, H. (2024). Modelling Digital Product Passports for the Circular Economy. In *Proceedings of the 2nd International Workshop on Knowledge Graphs for Sustainability (KG4S 2024) co-located with ESWC 2024*. Greece.

Jansen, M., Meisen, T., Plociennik, C., Berg, H., Pomp, A., & Windholz, W. (2023). Stop guessing in the dark: Identified requirements for digital product passport systems. *Systems*, *11*(3). doi: 10.3390/systems11030123

Kebede, R., Moscati, A., Tan, H., & Johansson, P. (2024). A modular ontology modeling approach to developing digital product passports to promote circular economy in the built environment. *Sustainable Production and Consumption*, *48*, 248-268. doi: https://doi.org/10.1016/j.spc.2024.05.007

Kuiper, J., Chirvasuta, T., & Breteler, J. (2026). A General-Purpose Tool to Convert Asset Administration Shell Templates to RDFS/OWL. *Procedia Computer Science*, *277*, 2943–2952. doi: 10.1016/j.procs.2026.02.330

Maija Breque, Lars De Nul, & Athanasios Petridis. (2021). *Industry 5.0: Towards a Sustainable, Human-Centric and Resilient European industry* (Policy Brief No. 10.2777/308407). Luxembourg: Directorate-General for Research and Innovation, European Commission.

Mogos, M. F., & Fragapane, G. (2022). Ways to Circular and Transparent Value Chains. In *Advances in Production Management Systems. Smart Manufacturing and Logistics Systems: Turning Ideas into Action* (Vol. 664, pp. 390–398). Cham: Springer Nature Switzerland. doi: 10.1007/978-3-031-16411-8_45

Nguyen, Q.-D., Suri, K., & Sidibe, G. D. S. (2025). Towards engineering product digital twins for industry 5.0: Definition and modeling approach. In *2025 IEEE 30th International Conference on Emerging Technologies and Factory Automation (ETFA)* (p. 1-4). doi: 10.1109/ETFA65518.2025.11205550

Pourjafarian, M., Plociennik, C., Bergweiler, S., Moarefvand, N., Brozeit, J., Rezapour, M., & Ruskowski, M. (2025). An odp-based ontology for the digital product passport. *Procedia CIRP*, *135*, 930-935. doi: https://doi.org/10.1016/j.procir.2024.12.125

Poveda-Villalón, M., Gómez-Pérez, A., & Suárez-Figueroa, M. C. (2014). OOPS! (OntOlogy Pitfall Scanner!): An Online Tool for Ontology Evaluation. *International Journal on Semantic Web and Information Systems (IJSWIS)*, *10*(2), 7–34.

Rimaz, M. H., Plociennik, C., & Ruskowski, M. (2024). Semantic Asset Administration Shell for Circular Economy. In *Proceedings of the 2nd International Workshop on Knowledge Graphs for Sustainability (KG4S 2024) co-located with ESWC 2024* (Vol. 3753, pp. 1–11). CEUR-WS.org.

Studer, R., Benjamins, V. R., & Fensel, D. (1998). Knowledge engineering: Principles and methods. *Data & Knowledge Engineering*, *25*(1–2), 161–197. doi: 10.1016/S0169-023X(97)00056-6

Valtanen, K., Saari, L. M., Domínguez, J. A. A., & Landolfi, G. (2024). Asset Administration Shell-Enabled Digital Product Passport Boosting Circular Innovation. In *Proceedings of the XXXV ISPIM Innovation Conference* (p. 7). Estonia.

Vegetti, M., Leone, H., & Henning, G. (2011). Pronto: An ontology for comprehensive and consistent representation of product information. *Engineering Applications of Artificial Intelligence*, *24*(8), 1305-1327. doi: https://doi.org/10.1016/j.engappai.2011.02.014

Wicaksono, H., Mengistu, A., Bashyal, A., & Fekete, T. (2025). Digital Product Passport (DPP) technological advancement and adoption framework: A systematic literature review. *Procedia Computer Science*, *253*, 2980–2989. doi: 10.1016/j.procs.2025.02.022

Zhang, Z., Gu, J., Li, R., Lian, W., Ma, C., & Tian, J. (2026). Bridging digital and green transitions: A literature review of digital product passport systems. *Renewable and Sustainable Energy Reviews*, *228*, 116600. doi: 10.1016/j.rser.2025.116600

About the authors

Dr. Quang-Duy Nguyen is a researcher at CEA List. Owning an engineer's degree in computer engineering, a master's degree in computer networking, and a Ph.D. in computer science, he has solid skills both in engineering and research. He is interested in building complex systems applied with novel technologies from the domains of IoT, OPC UA, Ontology, and AAS DT. You can contact the author at quang-duy.nguyen@cea.fr.

Dr. Fatima Danash is a researcher at CEA List specializing in ontologies. She holds a PhD in Applied Ontology, focusing on the abstraction and formal modeling of relations and concepts. Her current research explores Life Cycle Assessment (LCA), integrating Model-Based Systems Engineering (MBSE) and ontologies. You can contact the author at fatme.danash@cea.fr.

Dr. Arnab Sinha is a researcher at CEA List where he contributes to European R&D initiatives on eco-innovation and circular economy. His work focuses on the design and implementation of Digital Product Passport (DPP) aligned with emerging European and international sustainability and product compliance regulations, leveraging distributed systems, semantic technologies, and cloud infrastructures. He holds a Ph.D. in Computer Science and has extensive experience in data engineering, federated cloud/HPC platforms, and industrial digital transformation. His research interests include decentralized system architectures, knowledge representation, and data-driven sustainability. You can contact the author at arnab.sinha@cea.fr.

Dr. Chokri Mraidha is a director of research at CEA List where he is leading the Embedded and Autonomous Systems Design Laboratory. He got a master degree in distributed computing in 2001 and a PhD in Computer Science in 2005, and a "habilitation à diriger des recherches" diploma in the domain of computer science from Paris-Saclay University in 2015. His research and development interests include trustworthy software engineering for safety critical autonomous systems, and model-driven development methods and tools. You can contact the author at chorki.mraidha@cea.fr.