

7th International Conference on Industry of the Future and Smart Manufacturing
(former International Conference on Industry 4.0 and Smart Manufacturing)

A General-Purpose Tool to Convert Asset Administration Shell Templates to RDFS/OWL

Jesper Kuiper^{a,*}, Theodor Chirvasuta^a, Jeroen Breteler^a

^a*Netherlands Organisation for Applied Scientific Research (TNO), The Hague, The Netherlands*

Abstract

The Asset Administration Shell (AAS) has been an important industry framework for modeling data to populate Digital Twins. Recently, there has been interest in using AAS to populate Digital Product Passports. Since Digital Product Passport data can involve domains other than industry, many relevant data standards are expressed using Semantic Web standards rather than AAS, creating an interoperability issue. To address this issue, we present software for converting from AAS templates to a description in the Resource Description Framework (RDF).

Most previous literature on such conversions has presented domain-specific solutions. Our solution is general-purpose, which we demonstrate by applying it to various examples, including the Battery Pass and CarbonFootprint models. We also improve on previous work by targeting the latest major version (3.0) of the AAS standard. Furthermore, our solution casts the conversion output in terms of the OWL and RDFS standards, enabling Semantic Web tooling to consume detailed information from the AAS model about type hierarchies and value ranges. We highlight these interoperability gains by applying the Semantic Treehouse toolsuite to the converted AAS models, demonstrating new opportunities for model reuse and for consolidation of the AAS models with a broader range of semantic standards.

© 2025 The Authors. Published by Elsevier B.V.

This is an open access article under the CC BY-NC-ND license (<https://creativecommons.org/licenses/by-nc-nd/4.0>)

Peer-review under responsibility of the scientific committee of the 7th International Conference on Industry of the Future and Smart Manufacturing (former International Conference on Industry 4.0 and Smart Manufacturing)

Keywords: Asset Administration Shell; Ontology; Semantics; Resource Description Framework; Data Interoperability; Vocabulary Hub; Digital Product Passports

1. Introduction

In this paper, we present an implementation of software to convert from Asset Administration Shell (AAS) templates to data characterized by the Resource Description Framework (RDF) and leveraging the RDF Schema and Web Ontology Language concepts (RDFS, OWL, respectively). The AAS was introduced in the early 2010s as a solu-

* Corresponding author. Tel.: +31 6 27824239

E-mail address: jesper.kuiper@tno.nl

tion to address the fragmentation and lack of interoperability between machines and software systems from different manufacturers.

This work's primary motivation is the Digital Product Passports (DPP) usecase supporting the transition to a circular economy. The need for an interoperable digital framework for expressing DPP information is evident, and there have been many recent advancements towards this goal both using the AAS framework [12, 16] and RDF [9, 10]. In addition, the DPP reference architecture published in the CIRPASS-2 project recommends using RDF as the primary DPP communication framework [15]. Through the present work we seek to close the perceived AAS–RDF bifurcation in DPP developments; the conversion tool that we present is however domain-agnostic and can find usecases outside of the context of DPPs.

1.1. Overview Semantic Technologies – AAS and RDF/OWL

An AAS submodel comprises the actual content, or asset description of the asset administration shell. For submodels a distinction can be made between a submodel *instance* and a submodel *template* [8, Chapter 4.5]; whereas the former describes a concrete instantiation of an asset, the latter can be seen as a specification of how a submodel instance can be shaped. In short, a submodel template constitutes the data specification, whereas the submodel instance constitutes the data itself. The current research focuses on connecting the submodel template, specifically the data specification contained within it, with the RDF.

The current latest major version of AAS (V3.0) [8] has been released in 2022 bringing an alignment with IEC and ISO standards, while also improving the modularity and extensibility of the meta-models. RDF and OWL are semantic web frameworks standardized by W3C¹¹, used for representing machine-readable knowledge using classes, properties, and relationships. RDF describes data as collections of subject–property–object triple statements, while OWL adds expressiveness for reasoning about that data. In the scope of this research, only version 2 of OWL is considered, alongside the recommendation of the "QL" profile of OWL. This recommendation comes from the expectation that each AAS template will result in a limited-size semantic data model.

Research has noted the parallels between AAS and semantic standards developed in the context of Semantic Web [6], as well as the potential benefits of coupling the two. A notable overview of this is presented in Rongen et al. [18], who highlight among others the powerful querying and inference capabilities for RDF-based tooling that might also be unlocked for AAS data, if proper solutions for data transformation are in place. More recently, Louge et al. [14] demonstrated the integration of AAS and semantic web tooling for service orchestration using their COMPAAS ontology. They note (p. 9) that “converting asset descriptions [...] could encourage industrial applications [of COMPAAS] on production lines.” Finally, we wish to highlight an additional benefit: by strengthening the link between AAS and RDF contexts, AAS data can benefit from Semantic Web tooling for managing vocabularies. That is, users designing new AAS templates can be supported by an interface for organizing and reusing existing standardized vocabularies from both AAS and Semantic Web contexts. This can help to solve the challenge noted for AAS in van Leeuwen et al. [13] of “providing models that refer to formal ontologies or international standards” (p. 2).

1.2. Mapping AAS to RDF/OWL – The Status Quo

A variety of works have investigated aspects of mapping between AAS and RDF [6, 7, 3, 17]. While different papers have tried to tackle this challenge, we distinguish Bader and Maleshkova [1] as the foundation to our research, as it presents an RDF ontology for the AAS data model, along with RML mapping specifications for mapping from a serialized AAS instance to its semantic AAS counterpart in RDF. However, the ontology employs RDF concepts only for descriptive reasons and to allow inference, while the classes and properties themselves are proprietary to the Semantic Aspect Meta Model (SAMM) vocabulary⁹. Consequently, the SAMM ontology is not interoperable with other public ontologies built based on RDF / OWL concepts for classes and properties. Another drawback to this work is that it targets an outdated version of the AAS model.

Another relevant work is Da Silva et al. [3], where the authors provide a semantic Industry 4.0 component architecture using RDF & OWL, covering the top level concepts of “AdministrativeShell”, “Object”, “TechnicalFunctionality”, “Manifest”, and “Data”. However, the proposed solution is limited to non-standardized AAS submodels, it does not cover the AAS 3.0 specification. Furthermore, the paper also does not include a methodology or algorithm for automated translation of AAS formatted models to RDF / OWL.

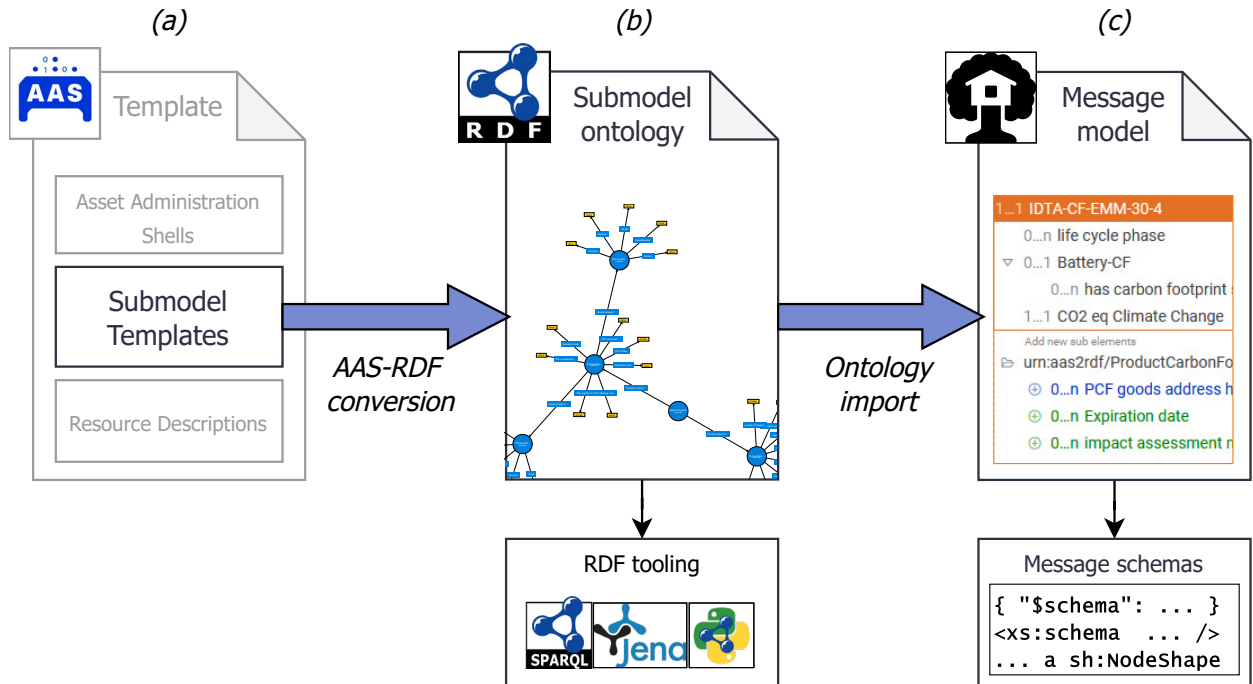


Fig. 1. Data model manipulation pipeline enabled by AAS to RDF conversion. From an AAS template (a) we use the conversion tool to convert to an ontology (b) that represents the AAS template datamodel. This ontology can be imported in tools like Semantic Treehouse, e.g. to create concrete digital message schemas (c).

When assessing the entirety of the related work, we find a lack of an implementation of transforming template-level data from an AAS submodel to an RDFS/OWL graph. Such a conversion capturing the entirety of the data model described in the AAS submodels could serve to improve the semantic interoperability of AAS models with one another and also with semantic data models developed outside of the AAS context. Only once converted to an RDFS/OWL format, an AAS submodel can be leveraged using existent ontology manipulation and vocabulary hub tools to create new value opportunities for parties using said models. In the following sections, we present our conversion solution and demonstrate how one can leverage the resulting, interoperable RDFS/OWL graph for model reuse and consolidation.

In Section 2 we briefly report on existing tooling within the AAS ecosystem and introduce the implemented AAS–RDF conversion method. Section 3 then reports on conversion examples of select AAS submodel templates and discusses their relevance and application. Finally, in Section 4 we conclude by discussing the general scientific contribution.

2. Material & Methods

In this section we provide a succinct overview of available tooling for the AAS ecosystem, and give a conceptual description of the proposed AAS to RDF conversion method. We provide a working example by converting several AAS template submodels released by the Industrial Digital Twin Association⁶ (IDTA) to RDF resources, and show how this fosters semantic interoperability by loading these resources into the aforementioned Semantic Treehouse. This enables the data specification design pipeline as shown in Figure 1.

2.1. Available Tooling for AAS

Since its publication in 2015, Asset Administration Shell has matured over the years, with a broader adoption through IDTA since the period 2020–2022. In turn, this motivated the creation of multiple software tools to leverage

this standardized digital representation. We provide a list of the most popular open source software tools we have identified and how they are related to our AAS–RDF interoperability goal.

aas-core-works is an open-source initiative providing software tools designed to facilitate the creation, manipulation, validation, and serialization of AAS models [2]. While the software toolbox provides an option for RDF serialization of AAS models, it does not convert submodels present in an AAS model to an RDF / OWL-based data model; rather, it introduces the submodels in the format of the AAS metamodel, serialized in RDF.

FA³ST (Fraunhofer AAS Tooling Suite) is a software development toolbox used for defining, validating and deploying AAS models.

Eclipse Digital Twin Multiple AAS-related software tools: *AASX Package Explorer*⁴ – a GUI-based tool used for reading, designing, and packaging asset administration shells, submodels; *BaSyx*⁵ – an open source framework for implementing and deploying AAS models.

Given this brief overview of existing AAS tools, we perceive a lack in coverage of the AAS–RDF interoperability goal that we have defined: Semantic model interoperability at the level of combining (subsets of) AAS submodel templates with other elements from non-AAS semantic models, as well as the creation of custom submodels based on reusing a subset of properties of existent submodels and thus integrating existent definitions in a new AAS submodel.

2.2. Goals for Our Solution

The capabilities of the solution we sought to implement can be summarized as follows.

- Firstly, we aim to convert AAS submodel *templates* to an RDF/OWL format that conceptually reflects the W3C fashion of nesting of properties and classes (e.g. `submodel a owl:Class`; `submodelProperty a owl:DatatypeProperty`; `submodelElementCollection a owl:ObjectProperty`; and consequently introduce a new `owl:Class` that reflects the collection of properties). In turn, this enables the two goals of our research:

AAS submodel template combinability through reuse Current semantic web tooling enable the creation of new named graphs / ontologies that reuse elements part of other semantic models. Converting the desired to be combined AAS submodels would then allow for the creation of a new AAS submodel, through the semantic web tooling

AAS submodel semantic interoperability with non-AAS semantic models By converting the data model contained in AAS template submodels to a general OWL-based RDF format, enable the connection with semantic models that are specified outside the AAS ecosystem.

- Secondly, the solution should be able to convert any AAS submodel template that adheres to the IDTA AAS V3.0 meta-model [8], i.e. we aim to create a generic and domain-agnostic implementation.

2.3. Conversion Method from AAS-Formatted XML to RDF

We use the software toolsuite **aas-core-works** [2] for interpreting XML- or JSON-serialized AAS resources into a runtime intermediate representation of the AAS structure. We then iterate over the deserialized AAS submodels and from them build an RDF graph, using **RDFLib** [11]. This graph can now be serialized again into one of many RDF formats, such as Turtle or JSON-LD. The intermediate representation is modelled after the **aas-core-meta**¹ formalised meta-models for AAS components.

Every submodel is reflected in the RDF graph as a separate OWL class where metadata fields are converted to their equivalent RDF relations that maintain the fields' semantics. A minimal example of a conversion of a single AAS submodel template can be found in Figure 2. Here, the CarbonFootprint submodel template and its description are converted to RDF/OWL, shown in Turtle syntax. The `displayName` XML element of the submodel, shown in Figure 2 (a), is translated to RDF/OWL as `rdfs:label` using the RDF language-tagging feature to identify English and German label values for the same property. The `description` XML element, expressed in Figure 2 (b), is translated

```

<submodel>
  <id>https://example.org#CarbonFootprint</id>
  <kind>Template</kind>
  <displayName>
    <langStringNameType>
      <language>de</language>
      <text>CO2-Fußabdruck</text>
    </langStringNameType>
    <langStringNameType>
      <language>en</language>
      <text>Carbon Footprint</text>
    </langStringNameType>
  </displayName>
  <description>
    <langStringTextType>
      <language>en</language>
      <text>The Submodel provides the means to access the Carbon Footprint (...)</text>
    </langStringTextType>
  </description>
  <submodelElements> (c) </submodelElements>
</submodel>

```

(a)

(b)

```

@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix aas: <https://admin-shell.io/aas/3/1/> .
@prefix ex: <https://www.example.org#> .

ex:CarbonFootprint a aas:Submodel ;
  rdfs:label "CO2-Fußabdruck"@de, "Carbon Footprint"@en ;
  rdfs:comment "The Submodel provides the means to access the Carbon Footprint (...) "@en ;
.

```

(a)

(b)

Fig. 2. An example of how an XML-serialized AAS template submodel is converted to RDF. (top) XML-serialized AAS submodel; (bottom) the converted RDF resource, in Turtle-format. Marked sections show how specific components convert.

to RDF/OWL as `rdfs:comment`, only in English. Finally, the translation of the `submodelElements` element shown in Figure 2 (c) will be touched upon in the coming paragraphs as it comes with additional complexity.

Note that we also annotate the created resources with the relevant class from the AAS ontology² published by the IDTA. As we will discuss later, this is helpful for converting back from an RDF representation to an AAS format.

Much of the logic of our implementation is in service of converting templates of submodel elements, i.e. the individual data points present in the submodel which, through properties, element lists and element collections, combine into a hierarchical data structure describing the asset. Submodel *templates* describe the submodel elements that are expected in a derived submodel instance, which can include their types, semantics and e.g. cardinality constraints. As the elements of submodel templates merely define the *shape* of the submodel's data structures, there are no direct instance values to convert.

2.3.1. Translating Submodel Elements

For our solution, submodel elements are converted to OWL object properties and datatype properties. A “property” submodel element represents a single data point and is as such modeled as a datatype property with as domain the submodel class itself (i.e. the class defined in Figure 2, bottom) and as range the XSD (XML Schema Definition) data type of the data point. Submodel element lists and collections are containers for multiple submodel elements and are therefore converted to OWL classes themselves, with an object property linked to it from the submodel class. An example of how submodel elements in its many forms are converted to RDF, is shown in Figure 3. Note here how a submodel element collection is converted to two RDF resources: an OWL class corresponding with the collection itself, and an OWL datatype property that relates the submodel class with the submodel element collection class.

This example also shows conversion for several submodel elements for different datatypes, and an expression of a cardinality qualifier as an OWL restriction on the relevant class.

2.3.2. On Global and Local Identification

Within the AAS meta-model, some components have a globally unique identifier (i.e., *identifiable*) whereas some other components have only a locally unique identifier “short ID” (unique within the set of components directly contained within the component’s parent element), making them only *refereable*. Where submodels are identifiable, submodel elements are only referable. This is problematic when converting them to nodes in the RDF graph, because in RDF all nodes *must* have a global identifier (in the shape of an Internationalised Resource Identifier, or IRI). We therefore construct RDF identifiers for non-identifiable components by prefixing the RDF identifier of its direct parent to the component’s short ID. This makes non-identifiable components in an AAS submodel globally identifiable in the RDF graph.

2.3.3. Available Serialization and Deserialization Methods

Currently, the deserialization of JSON- and XML-formatted AAS resources is supported. Reading the AAS package format AASX is not directly supported, but the constituting AAS resources can be extracted from such a package through the AASX Package Explorer⁴ tool. The converted RDF graph can be deserialized into many serialization formats such as Turtle, JSON-LD, N-triples, Notation-3, etc.

3. Results & Discussion

The source code for our conversion solution is publicly available under an open-source license in a git repository⁸. Example conversions of submodel templates, can be found in a separate repository⁷ to maintain a separation between the tool’s source code and its example artifacts. To illustrate the DPP usecase we converted some circularity and provenance-related IDTA submodels⁶, as well as the Battery Pass data model³. The Turtle files that can be found there have been automatically generated by the presented conversion tool. Currently the following submodel element types are supported: Property, MultiLanguageProperty, File, SubmodelElementList, and SubmodelElementCollection. These were found to be the most commonly occurring types, however during testing for the aforementioned conversion instances we found the types ReferenceElement, Entity, and Range to also occur. These are marked as future points of research and implementation.

For submodel element qualifiers, the conversion tool supports conversion of cardinality and one-of (“choices”) constraints. Other qualifiers that were found commonly relate to data entry illustration, such as example values and form titles. These were not considered for conversion as they have limited value in RDF graphs.

3.1. Leveraging AAS-RDF in Semantic Treehouse

The benefit of having an RDF representation of an AAS model is given by the available open source tooling to use the RDF representation for interoperable data specifications. For semantic web data model manipulation we consider the open-source vocabulary hub solution Semantic Treehouse¹⁰ which facilitates semantic interoperability within data spaces. The most notable feature that provides value to our solution is its interface to guide users through the process of configuring customized data specifications starting from one or multiple vocabularies (JSON Schemas or RDF ontologies). The resulting customized data specifications are materialized by generating JSON (instances and schemas), XML (instances and schemas), and OpenAPI specifications, based on the selection made by the users via the GUI.

To illustrate this, we consider modeling a Carbon Footprint data specification. Carbon Footprint as an AAS submodel is described by both the IDTA template repository and the Battery Pass data model. Having both submodels converted to RDF, we can now generate data specifications that reuse and repurpose elements from either submodel or the entire submodels all together. Figure 4 highlights how inside a Battery Pass specification, we can use the Semantic Treehouse interface to replace the specification of the Carbon Footprint property, switching between or even combine the IDTA template and the Battery Pass submodels. Crucially we retain a semantic backing of both submodels (albeit converted to RDF), enabling support for rigorous machine validation and semantic provenance for instance data.

```

<!-- as part of Submodel https://example.org#CarbonFootprint -->
<submodelElements>
  <property>
    <idShort>CarbonEquivalent</idShort>
    <qualifiers>
      <qualifier>
        <kind>ConceptQualifier</kind>
        <type>Cardinality</type>
        <valueType>xs:string</valueType>
        <value>One</value>
      </qualifier>
    </qualifiers>
    <valueType>xs:double</valueType>
  </property>
  <submodelElementCollection>
    <idShort>HandoverAddress</idShort>
    <value>
      <property>
        <idShort>Street</idShort>
        <valueType>xs:string</valueType>
      </property>
      <property>
        <idShort>HouseNumber</idShort>
        <valueType>xs:integer</valueType>
      </property>
    </value>
  </submodelElementCollection>
</submodelElement>

```

```

ex:CarbonFootprint rdfs:subClassOf [
  a owl:Restriction ;
  owl:cardinality 1;
  owl:onProperty ex:CarbonFootprint/carbonEquivalent ;
] .
ex:CarbonFootprint/HandoverAddress/street a aas:Property, owl:DatatypeProperty ;
  rdfs:domain ex:CarbonFootprint/HandoverAddress ;
  rdfs:range xsd:string ;
.
ex:CarbonFootprint/HandoverAddress/houseNumber a aas:Property, owl:DatatypeProperty ;
  rdfs:domain ex:CarbonFootprint/HandoverAddress ;
  rdfs:range xsd:integer ;
.
ex:CarbonFootprint/carbonEquivalent a aas:Property, owl:DatatypeProperty ;
  rdfs:domain ex:CarbonFootprint ;
  rdfs:range xsd:double ;
.
ex:CarbonFootprint/HandoverAddress a aas:SubmodelElementCollection;
.
ex:CarbonFootprint/handoverAddress a owl:ObjectProperty ;
  rdfs:domain ex:CarbonFootprint ;
  rdfs:range ex:CarbonFootprint/HandoverAddress ;
.

```

Fig. 3. An example of how submodel elements are converted to RDF, which would be embedded in Figure 2. (top) XML-serialized AAS SubmodelElements component, substitutes Figure 2 (c); (bottom) Converted RDF resources in Turtle format. Some details, like Turtle @prefix-statements, are omitted for brevity.

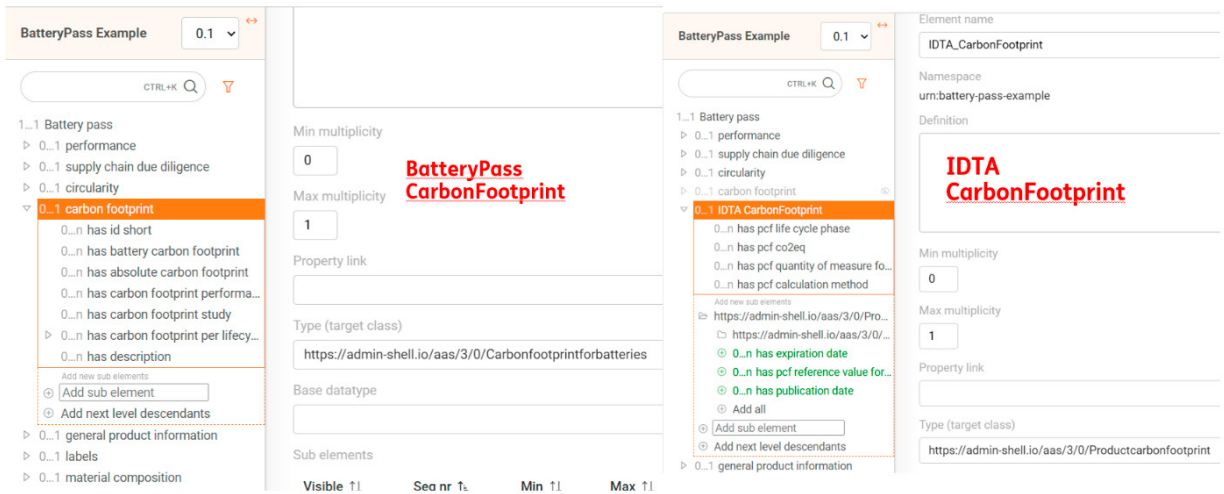


Fig. 4. Consolidation of a battery product passport data model by sourcing specifications from both the BatteryPass CarbonFootprint model and the IDTA CarbonFootprint template submodel

In this fashion the resulting data specifications thus enhance the interoperability of the two Carbon Footprint AAS models, as they can now be used interchangeably or mixed together, in combination with properties from other AAS and RDF data models.

3.2. Limitations & Future Work

The current implementation of the AAS–RDF conversion tool is limited in a number of ways: firstly, it does not yet support all AAS submodel element types, and secondly, the tool considers all AAS identifiers globally unique, making deduplication strategies necessary and reducing the interpretability of the output RDF ontologies. Furthermore, we have found that some AAS constructs (notably submodel element lists and collections) warrant the generation of two RDF constructs, namely an object property, and a class. An example is `ex:CarbonFootprint/[H|h]andoverAddress` from Figure 3 (e). This hinders a consistent identification in the RDF graph – ideally, both the property and the class should have the identifier of the original AAS construct to ensure proper back-referencing, yet they must also have different identifiers to ensure uniqueness. All these points need to be addressed before the tool can be reliably used in any domain.

A point of future work to fully integrate the proposed tooling with existing AAS infrastructure is to also implement the backwards conversion, namely from RDFS/OWL graphs to an AAS model. This point has also been mentioned by Rongen et al. [18] who outline the viability of such an RDF-to-AAS conversion and provide a rigorous ruleset for converting RDF classes and relations to submodels, elements, and collections. In our proposed AAS-to-RDF conversion tool, we support the conversion by annotating each RDF resource with its original AAS class type, which would significantly simplify this ruleset. Note that this annotation isn't always necessary, for example an `owl:Restriction` subclass with an `owl:cardinality` constraint *always* originates from an AAS Qualifier class of type “Cardinality”.

3.2.1. Conversion as a Mapping Specification

Another technical improvement for our solution could be to describe it as a mapping that can be consumed by generic mapping software. Bader and Maleshkova [1] took this approach; their conversion solution is cast as a specification for RDF Mapping Language [5]. The benefit of such approaches over custom scripting is that it splits the logic of how to calculate or construct the output from the description of what the desired transformation should be. The latter can be its own knowledge object, which could also be easier to inspect by users that might otherwise lack the familiarity with the programming language or codebase of the scripts. Suitable frameworks for describing the conversion include RML, which has seen continued development, and Facade-X [4]. Having a consistent AAS–RDF mapping

description would also more tightly support the AAS–RDF complementarity that is highlighted in i.a. Rongen et al. [18].

4. Conclusion

In this paper, we reported on an implementation of general-purpose conversion software from Asset Administration Shell templates to RDF-formatted data that leverages general RDFS and OWL standards. We demonstrated the conversion functionality for several examples, notably by converting select IDTA submodel templates and BatteryPass data models to Turtle, which is an RDF serialization. In addition to the benefits of Semantic Web standards for the AAS context identified by previous literature, we highlighted the fact that converting AAS templates to W3C-standard RDF enables their use in the Semantic Treehouse vocabulary hub tooling.

Through this work we contribute to closing the perceived semantic and tooling gap that exists between AAS and RDF, with a direct usecase in the domain of Digital Product Passports. Ideally, our solution would be addressing interoperability and user-friendliness challenges of adopting AAS for DPPs, as they are described by Valtanen et al. [19]. Finally, we connect our research with existing work around AAS–RDF *complementarity* – combining the strengths of both while cross-compensating for their weaknesses. Thus, we hope our work will help users reap the maximal rewards from all data modelling and standardization efforts.

Acknowledgements

This work was funded by the European Union’s Horizon Europe research and innovation program DaCapo (Grant agreement No. 101091780). We thank our colleagues in the DaCapo project for their collaboration. From VTT, we thank Markus Ylikärälä, Kristiina Valtanen, and Marko Jurmu for sharing their experiences with integrating different AAS models in creating new API specifications, and for providing us with relevant AAS models such as “Digital Nameplate V3.0” and “Technical Data”. From AIMEN, we thank Johnny Alvarado Domínguez and Emilio Costa Tobío for discussing an earlier version of this work.

Declaration of Generative AI and AI-Assisted Technologies in the Writing Process

During the preparation of this document the authors used available generative Large Language Models (LLMs) in order to draft short passages or generate alternative choices of words. After using these services, the authors reviewed and edited the content as needed. The authors take full responsibility for the content of the published article.

References

- [1] Bader, S., Maleshkova, M., 2019. The Semantic Asset Administration Shell, pp. 159–174. doi:[10.1007/978-3-030-33220-4_12](https://doi.org/10.1007/978-3-030-33220-4_12).
- [2] Braunisch, N., Ristin-Kaufmann, M., Lehmann, R., Wollschlaeger, M., Van De Venn, H.W., 2023. Empowering Industry 4.0 with Generative and Model-Driven SDK Development, in: IECON 2023- 49th Annual Conference of the IEEE Industrial Electronics Society, IEEE, Singapore, Singapore. pp. 1–6. URL: <https://ieeexplore.ieee.org/document/10312302/>, doi:[10.1109/IECON51785.2023.10312302](https://doi.org/10.1109/IECON51785.2023.10312302).
- [3] Da Silva, L.M.V., Köcher, A., Gill, M.S., Weiss, M., Fay, A., 2023. Toward a mapping of capability and skill models using asset administration shells and ontologies, in: 2023 IEEE 28th International Conference on Emerging Technologies and Factory Automation (ETFA), IEEE. p. 1–4. URL: <http://dx.doi.org/10.1109/ETFA54631.2023.10275459>, doi:[10.1109/etfa54631.2023.10275459](https://doi.org/10.1109/etfa54631.2023.10275459).
- [4] Daga, E., Asprino, L., Mulholland, P., Gangemi, A., 2021. Facade-X: an opinionated approach to SPARQL anything, in: Further with Knowledge Graphs. IOS Press. volume 53 of *Studies on the Semantic Web*, pp. 58–73. doi:[10.3233/SSW210035](https://doi.org/10.3233/SSW210035).
- [5] Dimou, A., Vander Sande, M., Colpaert, P., Verborgh, R., Mannens, E., Van de Walle, R., 2014. RML: A generic language for integrated RDF mappings of heterogeneous data., in: Bizer, C., Heath, T., Auer, S., Berners-Lee, T. (Eds.), *Proceedings of the 7th workshop of Linked Data on the Web*.
- [6] Grangel-González, I., Halilaj, L., Coskun, G., Auer, S., Collarana, D., Hofmeister, M., 2016. Towards a semantic administrative shell for industry 4.0 components. doi:[10.1109/ICSC.2016.58](https://doi.org/10.1109/ICSC.2016.58).
- [7] Huang, Y., Dhoub, S., Medinacelli, L., Malenfant, J., 2023. Semantic interoperability of digital twins: Ontology-based capability checking in aas modeling framework, pp. 1–8. doi:[10.1109/ICPS58381.2023.10128003](https://doi.org/10.1109/ICPS58381.2023.10128003).
- [8] Industrial Digital Twin Association, et al., 2023. Specification of the asset administration shell part 1: metamodel. URL: <https://industrialdigitaltwin.org/en/content-hub/aasspecifications/specification-of-the-asset-administration-shell-part-1-metamodel-idta-number-01001-3-0-2>. accessed: 2025-05-28.
- [9] Jansen, M., Blomqvist, E., Keskiärrkkä, R., Li, H., Lindecrantz, M., Wannerberg, K., Pomp, A., Meisen, T., Berg, H., 2024. Modelling digital product passports for the circular economy, in: *Proceedings of the The 2nd International Workshop on Knowledge Graphs for Sustainability*

- (KG4S 2024) colocated with the 21st Extended Semantic Web Conference (ESWC 2024), Hersonissos, Greece, pp. 14–28. URL: <https://www.ida.liu.se/~huali50/papers/2024-KG4S-JBKLLWPMB.pdf>.
- [10] Jansen, M., Blomqvist, E., Keskisärkkä, R., Li, H., Lindercrantz, M., Wannerberg, K., Pomp, A., Meisen, T., Berg, H., 2025. Designing an Ontology Network for Digital Product Passports, in: Meroño Peñuela, A., Corcho, O., Groth, P., Simperl, E., Tamma, V., Nuzzolese, A.G., Poveda-Villalón, M., Sabou, M., Presutti, V., Celino, I., Revenko, A., Raad, J., Sartini, B., Lisena, P. (Eds.), *The Semantic Web: ESWC 2024 Satellite Events*. Springer Nature Switzerland, Cham. volume 15345, pp. 220–237. URL: https://link.springer.com/10.1007/978-3-031-78955-7_19, doi:10.1007/978-3-031-78955-7_19. series Title: Lecture Notes in Computer Science.
- [11] Krech, D., Grimnes, G.A., Higgins, G., Car, N., Hees, J., Aucamp, I., Lindström, N., Arndt, N., Sommer, A., Chuc, E., Herman, I., Nelson, A., McCusker, J., Gillespie, T., Kluyver, T., Ludwig, F., Champin, P.A., Watts, M., Holzer, U., Summers, E., Morris, W., Winston, D., Perttula, D., Kovacevic, F., Chateaneu, R., Solbrig, H., Cogrel, B., Stuart, V., 2025. RDFLib. URL: <https://github.com/RDFLib/rdfLib>, doi:10.5281/zenodo.6845245.
- [12] Kühn, M., Baumann, M., Volz, F., Stojanovic, L., 2025. Digital product passport design supporting the circular economy based on the asset administration shell. *Sustainability* 17, 969. URL: <https://publica-rest.fraunhofer.de/server/api/core/bitstreams/0d3f8348-2b3b-4bc7-b018-86468c352b03/content>. publisher: MDPI.
- [13] van Leeuwen, C., Bouter, C., Hindriks, R., Wilterdink, R.J., 2022. Representing the virtual: Using AAS to expose digital assets, in: *Proceedings of the Third International Workshop on Semantic Digital Twins co-located with the 19th Extended Semantic Web Conference (ESWC 2022)*.
- [14] Louge, T., Araghi, S.N., Karray, M.H., Sarkar, A., 2025. Events-based semantic services composition in industry 4.0 using asset administration shell meta-model for digital twins. *Expert Systems with Applications* 271, 126641. URL: <https://www.sciencedirect.com/science/article/pii/S0957417425002635>, doi:<https://doi.org/10.1016/j.eswa.2025.126641>.
- [15] van Nieuwenhuijze, H., Chawla, K., Smeets, N., Roefs, D., Rongen, S., Kuiper, J., Rurup, R., Lazrak, M.N., Bernier, C., Gérard, S., Henry, T., Tamm, T., Eisler, R., Matruglio, A., Volpini, M., 2025. D4.1 reference architecture. URL: <https://doi.org/10.5281/zenodo.15467720>, doi:10.5281/zenodo.15467720.
- [16] Pourjafarian, M., Plociennik, C., Rimaz, M.H., Stein, P., Vogelgesang, M., Li, C., Knetsch, S., Bergweiler, S., Ruskowski, M., 2023. A multi-stakeholder digital product passport based on the asset administration shell, in: *2023 IEEE 28th International Conference on Emerging Technologies and Factory Automation (ETFA)*, IEEE. pp. 1–8. URL: <https://ieeexplore.ieee.org/abstract/document/10275715/>.
- [17] Rimaz, M.H., 2024. Semantic Asset Administration Shell for Circular Economy. Master's thesis. doi:10.13140/RG.2.2.33957.10729.
- [18] Rongen, S., Nikolova, N., van der Pas, M., 2023. Modelling with AAS and RDF in Industry 4.0. *Computers in Industry* 148, 1–12. URL: <https://www.sciencedirect.com/science/article/pii/S016636152300060X>, doi:<https://doi.org/10.1016/j.compind.2023.103910>.
- [19] Valtanen, K., Saari, L.M., Alvarado Domínguez, J.A., Landolfi, G., 2024. Asset Administration Shell-enabled Digital Product Passport boosting circular innovation, in: *Proceedings of the XXXV ISPIIM Innovation Conference*, Tallinn, Estonia. pp. 1–6. URL: https://cris.vtt.fi/ws/portalfiles/portal/102996405/Dacapo_AAS_DPP_ISPIIM_2024.pdf. ISBN Pending.

Web references

1. AAS Core v3.0 meta-model: <https://aas-core-works.github.io/aas-core-meta/v3/index.html>, accessed 2025-05-27.
2. AAS Ontology: <https://github.com/admin-shell-io/aas-specs-metamodel/blob/6fd399cd64ea7793a9c7c71ca4cfbd065ae81a00/schemas/rdf/rdf-ontology.ttl>, accessed 2025-06-25.
3. Battery Pass Data Model: <https://github.com/batterypass/BatteryPassDataModel>, accessed 2025-05-28.
4. Eclipse AASX Package Explorer: <https://github.com/eclipse-aaspe/package-explorer>, accessed 2025-09-17.
5. Eclipse BaSyx: <https://basysx.org/>, accessed 2025-09-17.
6. IDTA AAS Submodel Templates: <https://industrialdigitaltwin.org/en/content-hub/submodels>; <https://github.com/admin-shell-io/submodel-templates>, accessed: 2025-05-08.
7. Repository of converted AAS template submodels: <https://gitlab.com/jesper-sk/aasrdf-artifacts/-/tree/f8bab9caf10bd86294bcfde8a5fffb65b9bcff37c/>, accessed 2025-06-25.
8. Repository containing the source code of this contribution: <https://gitlab.com/jesper-sk/aas-rdf-converter/-/tree/1eecb8e6cf8d245c81418a30ab472b0f078dd880/>, accessed 2025-06-25.
9. Semantic Aspect Meta Model: <https://github.com/eclipse-esmf/esmf-semantic-aspect-meta-model>, accessed 2025-05-29.
10. Semantic Treehouse: <https://www.semantic-treehouse.nl/>, accessed 2025-05-28.
11. W3C standardized semantic web frameworks: RDF <https://www.w3.org/RDF/>, accessed 2025-05-29; OWL <https://www.w3.org/OWL/>, accessed 2025-05-29.