

# Semantic Asset Administration Shell for Circular Economy

Mohammad Hossein Rimaz<sup>1</sup>, Christiane Plociennik<sup>1,\*</sup> and Martin Ruskowski<sup>1</sup>

<sup>1</sup>Deutsches Forschungszentrum für Künstliche Intelligenz GmbH (DFKI), Trippstadter Str. 122, 67663 Kaiserslautern

## Abstract

The need to shift from a linear economy to a circular economy (CE) is not only because of the decline in raw material resources, but also because of regulations and mandates for climate and environmental protection. Digital Twins serve as enablers for this shift. However, proprietary systems and data formats create interoperability barriers between stakeholders. In this context, the implementation of a standard Digital Twin is crucial, and the Asset Administration Shell (AAS) aims to solve these interoperability gaps. This work utilizes AAS and semantic technologies to address interoperability issues. SPARQL allows querying and retrieval of information for knowledge discovery. SHACL provides a flexible method for validating RDF data and ensuring its quality. Rule-based and Ontology-based reasoning can also aid in decision-making processes. By using the RDF representation of AAS, seamless integration with existing RDF models can be achieved. Furthermore, our work introduces the first RDF-JSON de-/serializer compatible with AAS metamodel version 3, which is crucial for the AAS community. Our approach enables complex knowledge discovery and inference for both IT experts and normal users through simplified user interfaces, which can be applied in the context of CE to improve visibility, transparency, and decision support, as demonstrated through a use case.

## Keywords

Asset Administration Shell, Circular Economy, Digital Twin, Digital Product Passport, SPARQL

## 1. Introduction

The circular economy (CE) is a concept that was developed in response to the environmental issues caused by the traditional linear economy, which excessively depletes resources and produces a substantial amount of waste. Electronic waste is currently the fastest-growing waste stream in Europe, as well as the fastest-growing category of hazardous solid waste [1]. This waste stream contains valuable raw materials, such as copper, cobalt, silver, gold, and lithium, which are costly and environmentally damaging, emphasizing the importance of recycling them. Despite the European Union's 2019 mandate to recycle 65% of all electrical appliances, actual recycling rates remain low, with Germany achieving only 44.1% and the EU average falling below 40%. This leads to a significant loss of recyclable materials [2].

Digitalization is a crucial factor in enabling the CE. By assigning a digital representation, or Digital Twin (DT), to each product, machine, or other significant asset, numerous opportunities


---

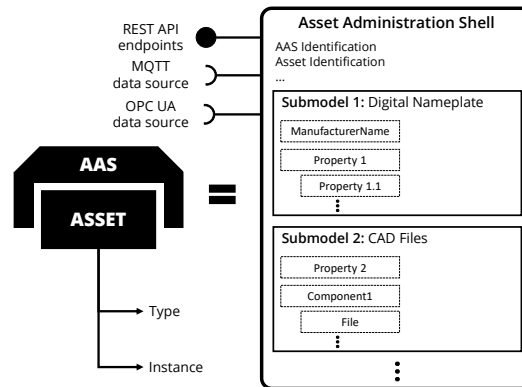
*The 2nd International Workshop on Knowledge Graphs for Sustainability (KG4S2024) – Colocated with the 21st Extended Semantic Web Conference (ESWC2024), May 27th, 2024, Hersonissos, Greece.*

\*Corresponding author.

✉ [hossein.rimaz@dfki.de](mailto:hossein.rimaz@dfki.de) (M. H. Rimaz); [christiane.plociennik@dfki.de](mailto:christiane.plociennik@dfki.de) (C. Plociennik); [martin.ruskowski@dfki.de](mailto:martin.ruskowski@dfki.de) (M. Ruskowski)

ORCID [0000-0002-7777-9556](https://orcid.org/0000-0002-7777-9556) (M. H. Rimaz); [0000-0002-6534-9057](https://orcid.org/0000-0002-6534-9057) (M. Ruskowski)

 © 2024 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).



**Figure 1:** Basic structure of AAS.

become accessible. For a DT to be effective, it must be easily comprehensible and usable by all parties involved, both humans and machines. When a DT is interoperable, it facilitates seamless communication and interaction among all stakeholders. Digital Product Passports (DPP) are digital profiles that provide detailed information about a product's origin, materials, usage, and end-of-life handling. It might also be called by other names, such as Product Passport, Digital Lifecycle Passport, or specialized in certain domains, such as Battery Passport [3]. Beginning in February 2027, batteries over 2 kWh are required by law to have a unique battery passport, retrievable using a unique product identifier in the form of a QR code [4].

Platform Industry 4.0 was launched on March 16, 2015 by the German Federal Ministry for Economic Affairs and Climate Action and the Federal Ministry of Education and Research. In the same year, the Reference Architectural Model Industry 4.0 (RAMI 4.0) was introduced, providing a structured framework for understanding and communicating the concept of Industry 4.0. The term *Asset Administration Shell* (AAS) started to appear in 2018 with the publication of the first detailed specification. The Industrial Digital Twin Association (IDTA) was established in 2021, aiming to promote the practical use of the AAS and serve as the user organization for it [5]. Version 3 of the AAS specification is the most recent version and was published in April 2023 [6]. Realizing the concept of DPP with standardized approaches like AAS reduces interoperability barriers and makes it understandable and accessible.

AAS, unlike any other proprietary format, aims to solve interoperability issues. This is done by introducing a common structure and skeleton to describe and represent the Digital Twin, as depicted in Figure 1. With standardized modeling elements similar to UML, which form a metamodel, assets can be described, and a Digital Twin can be created. Additionally, it is crucial to agree on a standardized format for data exchange to ensure serialized information is properly structured. JSON, XML, and RDF are standardized representation formats for AAS. Information about an asset is formed in specific structures, called `Submodel1s`, each focusing on a specific aspect of an asset and holding various attributes and properties that are called `SubmodelElements`.

The project ReCirCE<sup>1</sup> started with the goal of improving the efficiency of material recycling

<sup>1</sup><https://www.recirce.de/>

by combining Digital Twins and artificial intelligence methods. With five project consortium members, which were supported by more than seven associated partners, the project concluded successfully in September 2023. The main motivation of this work originates from the needs for intelligent and configurable waste sorting decision-making. Information about each product and its life cycle can be made available in an interoperable and machine-readable structure using AAS. These data can be accessed and manipulated via AAS REST API specification. However, when millions of products have their own digital representation, a proper and interoperable approach is needed for query, knowledge discovery, and individual decision-making. Explicitly modeling information is not preferred and causes many inconsistencies in derived attributes. Hence, reasoning and knowledge inference are important aspects for many use cases. In this paper, we show how this can be achieved.

This work is the first within the landscape of AAS that leverages the official RDF representation of AAS. Leveraging this RDF representation allows us to use W3C standards such as SPARQL to query information, SHACL to validate information, and semantic reasoners for decision-making in an interoperable manner. We developed a python package<sup>2</sup> that solves the lack of a proper tool that handles the RDF representation of AAS. The lack of such tooling hampered adoption in the community and, hence, kept many errors in the RDF representation hidden. This work also emphasizes these issues, such as lack of support for ordered elements, in the current RDF representation of AAS. Furthermore, in order to show that our approach is usable, we present user interfaces developed to support non-expert users to interact with the system.

In Section 2 we present a concrete use case, so our goal becomes more clear. Section 3 presents related work with respect to circular economy and AAS. In Section 4 we explain the overall architecture of system and components and how different technologies can be used together. In Section 5 we will have a discussion about our approach and also the current problems in the RDF model of AAS. Finally, in Section 6 we will have our conclusion and future works.

## 2. Motivation and Use Case

In the project ReCircE, an intelligent waste sorting process has been employed in the Fraunhofer Research Institution for Materials Recycling and Resource Strategies (IWKS) as a pilot waste sorting facility, which is depicted in Figure 2. Figure 2a shows the input of the sorting facility, which consists of different electronic waste, such as smartphones, cameras, and printers. Based on the available information about each device, items are separated based on user-defined rules by an air nozzle, as depicted in Figure 2b. Identification of assets can happen via object recognition methods [7]. Finally, these items are stored in different containers, as shown in Figure 2c. For reporting reasons, the user wants to know the total amount of gold and other materials, or the average age of all the assets in a container.

In the context of the project ReCircE, various Submodels were reused or developed. In order to fulfill our use case scenario, a proportion of the AAS models was taken from the project ReCircE. However, *py-aas-rdf* is capable to handle all valid Submodels. As depicted in Figure 3, *DeviceSpecification* is a simple Submodel initially designed for sorting electronic waste which contains basic information about electronic devices and their raw materials' composition. Such

---

<sup>2</sup><https://github.com/mhrimaz/py-aas-rdf>



(a) Input conveyor belt.

(b) Air nozzles for separation.



(c) Sorted output.

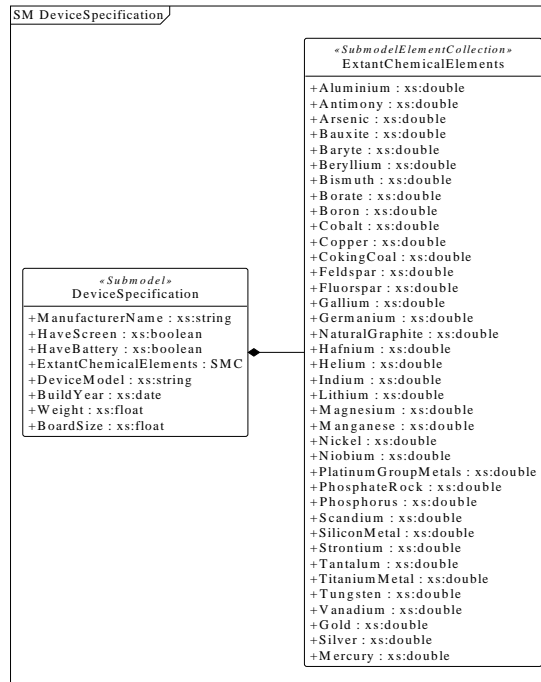
**Figure 2:** E-Waste sorting process in a pilot sorting facility in project ReCircE.

information may exist in the form of standard Submodel templates in the future. For the list of materials, the list of European critical raw materials [8] has been used. Within the scope of the project ReCircE, these data were only available for a small number of devices. For this reason, we synthetically generate random data to create a large number of catalogs. This will allow for performance benchmarks later on.

Based on the provided structure and syntactically generated data, we develop queries, related user interfaces, and tools. The provided solution can also be applied to other use cases.

### 3. Related Work

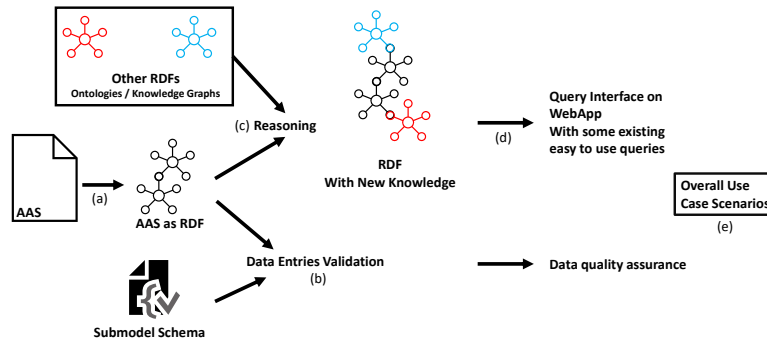
The role of digital technologies in the circular economy is emphasized by Pagoropoulos et al. [9] and the concept of circular economy and the interactions between stakeholders are also studied by Salminen et al. [10]. As mentioned by Walden et al. [11], the reasons for being so far away from a circular economy are the lack of transparency, standardization, and data sharing. The Digital Lifecycle Passport, introduced by Plociennik et al. [12], is meant to maintain all related data about a product throughout its lifecycle by various stakeholders through an interoperable approach via utilizing Asset Administration Shell. Li et al. [13] promoted the idea of using ontologies to enable cross-domain understanding in the circular economy, which includes a survey of existing ontologies related to sustainability, materials, products, manufacturing, and logistics. Mügge et al. [14] presented an R-Strategy Assistant for automotive industry, developed within the Catena-X project to determine the best CE strategy at the end of a vehicle's life.



**Figure 3:** Submodel for *DeviceSpecification* represented as a class diagram.

Bader et al. [15] did the foundation work for the semantic representation of AAS and coined the term "Semantic Asset Administration Shell". It provides a mapping of the elements of AAS to RDF, as well as SHACL rules to validate the RDF representation. An AAS contains various *Submodels* that hold information about an asset. Bouter et al. [16] proposed a Submodel development methodology and introduced an approach to finding matching Submodels required for a specific application. As shown by Rongen et al. [17], the migration of existing RDF-based models to an AAS model enables semantic discovery of assets and interoperability. In their work, existing RDF models are translated into AAS models based on their SHACL shapes. However, rather than using the official RDF representation of AAS, they used a modified version to allow writing queries that involved References. Using AAS allows using its REST API to expose the content of a DT. However, syncing it with the RDF model with their mapping approach is not straightforward. Huang et al. [18] presented a capability-checking solution based on AAS. In their approach, information in AAS are extracted and converted into instances of the MaRCO ontology [19] and then they used SPARQL Inferencing Notation (SPIN) to represent the matchmaking rules. Rimaz et al. [20] showed data entry validation of AAS via SHACL shapes to be sure about the consistency and quality of existing data, which enables a higher level of interoperability. Luxenburger et al. [21] proposed an open-source AAS-based service infrastructure that leverages an LPG-based knowledge graph in Neo4j and Cypher as a query language. Moreno et al. [22] showcased an architecture that converts AAS to RDF via RML mappings and showed how other semantic models and ontologies are integrated as a coherent solution. In Section 4, we will elaborate how our approach differs from previous works.

## 4. Methods



**Figure 4:** Overall architecture of our solution.

Our solution takes a semantic approach and relies on a semantic technology stack. The core components of our semantic Digital Twin or semantic Asset Administration Shell solution are shown in Figure 4. First, Figure 4–(a) represents the components used to construct the Knowledge Graph and to serialize AAS from other representation formats such as JSON and XML to RDF and vice versa.

It is important to note that the derived RDF serialization complies with the specifications of version 3.0 of the Asset Administration Shell. Validation is not limited to the structure of RDF, but it can also be expanded to validate the content and business logic constraints. As it is shown in 4–(b), at the end, performing data quality assurance tasks will be possible.

As depicted in 4–(c), the reasoning component infers new facts based on given information and rules that are also expressed in the RDF form. Such inferred knowledge can eventually be used in the context of knowledge discovery and decision-making scenarios, as depicted in 4–(d). Finally, all the components fit together in concrete use cases derived from the circular economy context and shown in 4–(e).

Table 2 provides a comprehensive overview of various aspects of this work. It highlights how state-of-the-art methods tackle these aspects, identifies current gaps, and summarizes our proposed approach.

In the research community, RML is a typical way to convert JSON, XML, and CSV data to RDF, and R2RML to convert relational databases to RDF. In a previous work [20], RML mapping was used to convert AAS to RDF developed as the first approach for Knowledge Graph (KG) construction. However, developing and maintaining a complete mapping is very challenging because there are more than 50 classes in the metamodel. However, the issue with RML is that it is a one-way trip, so going back from RDF representation to JSON or XML representation is no longer possible with RML. This is essential, as many web technologies and front-end frameworks rely heavily on JSON serialization. Furthermore, other software, such as AASX Package Explorer, might rely on XML/JSON representation. For this purpose, a custom parser and de-/serializer is a more suitable approach.

The implemented solution uses python pydantic<sup>3</sup> library to express the core elements of AAS. The advantage of pydantic is that it can support JSON serialization and automatically generate a JSON schema, which is helpful for API documentation. Furthermore, if any modification happens at metamodel level, then the corresponding pydantic model will be updated, and the JSON serialization and schema will get updated automatically. pydantic does not support RDF serialization. As a result, another third-party library is required. RDFLib<sup>4</sup> is a popular python library to deal with RDF. For each element in the metamodel, we have a pydantic model, which also facilitates the functionality to convert an instance of it to an RDF graph or convert an input RDF graph to an instance object of that class. Table 1 summarizes all the third-party components used in this work.

**Table 1**

Main third-party solutions used in this work.

Software	Version	Role	License
Apache Jena	4.10.0	RDF Triplestore and Reasoning	Apache License, Version 2.0
FastAPI	0.108.0	REST API	MIT License
Ontotext GraphDB	10.4.2	RDF Triplestore	Freemium and Enterprise license
pydantic	2.5.3	Metamodel Elements in python	MIT License
RDFLib	7.0.0	RDF manipulation	BSD-3-Clause license

**Table 2**

Brief overview of our contributions compared to the state of the art.

	State-of-the-Art	Gaps	Our Approach
<b>KG Construction</b>	- Only RML [15, 18, 22]	- Converting RDF to JSON/XML is not possible. - RML is not suitable for recursive structures.	- Partial RML for Version 3 [20] - Custom de-/serializer ( <i>py-aas-rdf</i> )
<b>Data Quality Assurance</b>	- Custom validators - SHACL/JSON/XML schema for metamodel [15]	- Co-Constraint and complex scenarios	- SHACL - SHACL-SPARQL for complex cases
<b>Search in AAS</b>	- Task-Specific solutions - Neo4j & Cypher query language [21]	- Hard-coded solutions - Lack of standard query language - Lack of common database schema	- Official AAS represented as RDF - SPARQL standard query language
<b>Reasoning with AAS</b>	- RDF & SPIN Rules [18]	- SPIN is superseded by SHACL - Complex Rule-Based reasoning	- Apache Jena Rule-Based reasoning
<b>AAS Metamodel Version</b>	- Version 2, Version 3 RC02	- Version 3 introduced in April 2023	- Version 3

The dashboard uses server-side template rendering and is written with Python, Django, Bootstrap 5, HTML5, CSS3, and vanilla JavaScript and jQuery. Since users are not SPARQL experts, a simple query-building component was created. Figure 5 depicts the content of this page. The query builder uses jQuery-QueryBuilder<sup>5</sup> MIT licensed. Predefined criteria can be selected and combined, and then a JSON structure will represent the combination of criteria and selected values. These values will then be translated into a SPARQL query, and then the query engine will evaluate it.

<sup>3</sup><https://pydantic.dev/>

<sup>4</sup><https://github.com/RDFLib/rdfliib>

<sup>5</sup><https://github.com/mistic100/jQuery-QueryBuilder>

## Knowledge Discovery

Your Active Rules:

Rule Name	Rule Description	Manage
Dangerous	Dangerous items according to the BOM Submodel	<a href="#">Select/Edit</a> <a href="#">Remove</a>
Valuable	Valuable items according to the BOM Submodel	<a href="#">Select/Edit</a> <a href="#">Remove</a>

Showing 1 to 2 of 2 rows

Criteria rules:

**AND** **OR** [Add rule](#) [Add group](#)

Gold [gramm]	greater	12	<a href="#">Delete</a>
Build Year [YYYY]	greater	2010	<a href="#">Delete</a>

[Rules](#)

**Figure 5:** Query builder page.

Figure 6 depicts the content of the the *Search Engine* page for expert users. The *Search Engine* page provides a simple interface to directly run an information retrieval query. A set of existing queries can be configured for each user. This configuration can be intelligent, with a query recommender system working in the background, or, in this case, some hard-coded examples.

The result of the query will only contain basic information, including the identifier of the Shell. Then, with the Shell viewer page, the user can see the content of a specific Shell. Figure 7 depicts the user interface to explore all available assets. The user can click on the "View Shell" button to see the content of the Digital Twin. A user interface to edit the content of the DT is implemented, which can validate the user input based on predefined SHACL shapes [20].

Furthermore, Ontotext GraphDB offers a visualization dashboard, depicted in Figure 9, through which users can interactively see the graph and its content, expand elements, and see relations. As an alternative to the GraphdDB visualization dashboard, it is also possible to use AWS's Graph Explorer<sup>6</sup> which is an open-source React application published under Apache 2.0 license for graph visualization and exploration.

Rule-based reasoning is not part of the created dashboard, however, Figure 8 depicts an example rule expressed in Apache Jena's rule syntax. As a condition, the rule checks if the temperature level is more than a specific value. In that case, various attributes can be assigned to the appropriate component, or a new Property can be added to the Submodel. This allows you to dynamically create AAS attributes and model dependent properties. In our use case, instead of modeling explicitly if a product can cause a hazard in the recycling phase, we can derive this attribute based on predefined rules.

Figure 10 shows an example query to identify dangerous assets. The advantage of this query is that since both SPARQL and RDF representation of AAS are standardized, any other stakeholder

<sup>6</sup><https://github.com/aws/graph-explorer>



understands the query, and, more importantly, federated queries are easily possible due to the usage of standardized approaches. With SPARQL, users can aggregate desired properties or create reports, for example, the average age of recycled electronic devices from a specific manufacturer.

## 5. Discussion

The implemented dashboard does not leverage rule-based reasoning because most available reasoners only work in-memory and the ability to scale up horizontally is practically limited to commercial solutions. We utilized Apache Jena and its rule syntax for reasoning, which also provides rule-derivation logs that act as explanations of the decision-making process. However, we do not cover other rule languages like SWRL or RIF.

One of the most important aspect of AAS is its standardized REST API endpoints that allows interacting with and modifying information of a DT. AAS hosting solutions like Eclipse BaSyx allows this with JSON as the representation format. However, none such hosting solution offers an RDF-based backend. This is something that is possible as demonstrated in a pull-request<sup>7</sup>. This means interaction with the REST API is possible without the need to maintain or sync different systems.

The most critical issue with the RDF representation, at the time of writing this paper, is that it is not a lossless representation, because the ordering of order-relevant elements is not kept<sup>8</sup>. In the metamodel, we have the `Reference` element. The `Reference` element has a `type` and `keys`. `type` can be either `ModelReference` that refers to an element of AAS or `ExternalReference`. `keys` is a list of keys and the ordering of them is important. Also, `SubmodelElementList` is an ordered list of `SubmodelElements`. There are various ways to represent ordered elements in RDF, such as using the `aas:index` property to hold the ordering information. With this addition, *py-aas-rdf* is capable to convert all AAS models bidirectionally from JSON to RDF and vice versa. For XML, other third-party libraries can be used to convert JSON to XML or AASX package files.

The usage of datatypes is not convenient, and all literals have `xsd:string` datatype and the actual datatype is decoupled from the literal. There are some decision factors<sup>9</sup> related to this. However, this heavily impacts the performance of the system in real-world use cases. Furthermore, instead of using an RDF language tag like `"multi language text"@en`, two separate attributes called `text` and `language` are used.

In RDF, prefixes are typically used to define compact representations for URIs. When you define a prefix, it is typically followed by a local name to create a compact URI. However, in RDF, the local name after a prefix should adhere to the syntax rules for `NCName` (non-colonized names). Specifically, the forward slash (`/`) is not allowed in `NCNames`. Slashes are used to separate different components in a URI, indicating a hierarchical structure. The AAS namespace is set to be `https://admin-shell.io/aas/3/0/`, this allows to refer to a `Submodel` by simply using `aas:Submodel`, however further elements such as `Property` should be explicitly mentioned

---

<sup>7</sup><https://github.com/eclipse-basyx/basyx-java-server-sdk/pull/167>

<sup>8</sup><https://github.com/admin-shell-io/aas-specs/issues/45>

<sup>9</sup><https://github.com/admin-shell-io/aas-specs/issues/284>

like `<https://admin-shell.io/aas/3/0/AasSubmodelElements/Property>`. Such naming is not according to the best practices and is not idiomatic RDF.

The structure of AAS allows for an indefinite level of nesting. This means you can have a `SubmodelElementCollection` inside another `SubmodelElementCollection`. This causes some impracticality and performance issues, as well as interoperability issues. For example, indefinite nesting can cause stack-overflow problems, and as it is already suggested<sup>10</sup> a maximal recursion depth should be proposed.

## 6. Conclusion

The shift from a linear economy to a circular economy is driven by the scarcity of raw materials and environmental regulations, with Digital Twins playing a key role in facilitating this transition. However, interoperability challenges arise due to proprietary systems and data formats, necessitating a standardized approach such as the Asset Administration Shell (AAS). While the AAS introduces a metamodel to describe assets, issues persist, including the need for methods that ensure data correctness and consistency within the AAS and facilitate knowledge discovery for diverse stakeholders.

This work is based on the RDF representation of the AAS, which was started by Bader et al. [15] and is now part of the AAS specification. However, due to a lack of tooling, converting XML or JSON representation to RDF and vice versa was not previously possible. To address this, RML was employed as a preliminary solution to map JSON to RDF [15, 20, 22]. This work introduces the first functional custom parser that can serialize JSON to RDF and RDF to JSON. In addition, this work sheds light on issues with the current RDF representation of the AAS and proposes possible solutions. The most important aspect is the lack of support for ordered elements. This can be easily expressed in JSON format but is not trivial in RDF. The inherent benefits of semantic technologies that can help in knowledge discovery via SPARQL, data validation via SHACL, and semantic reasoning by relying on semantic axioms are presented. In addition, the usability of these concepts in the context of the circular economy was demonstrated via a use case scenario from the project ReCirE. We further showcased some user interface prototypes and ideas to not only keep the work purely technical, but also bring it into action and help non-technical end-users.

A crucial aspect of any value chain is data sovereignty, and it is essential for partners and stakeholders to ensure that their data are used as agreed upon and accessed for the agreed-upon duration. The inclusion of International Data Spaces and Gaia-X in an end solution is vital. The use of Large Language Models (LLMs) and AAS presents a promising future direction. The integration of LLMs and AAS has the potential to unlock numerous possibilities in various domains. Conversational question-answering over Knowledge Graphs (KGs) is gaining momentum, and LLMs can facilitate reporting and verbalizing entities. Furthermore, LLMs can aid in constructing new `Submodels` or their corresponding SHACL shapes, and recommend appropriate `Submodels` based on user requirements. In the end, it will be interesting to integrate all these aspects into a cloud solution as an extension of our previous work [23].

---

<sup>10</sup><https://github.com/admin-shell-io/aas-specs/issues/333>

## References

- [1] R. Widmer, H. Oswald-Krapf, D. Sinha-Khetriwal, M. Schnellmann, H. Böni, Global perspectives on e-waste, *Environmental impact assessment review* 25 (2005) 436–458.
- [2] M. Shahabuddin, M. N. Uddin, J. Chowdhury, S. Ahmed, M. Uddin, M. Mofijur, M. Uddin, A review of the recent development, challenges, and opportunities of electronic waste (e-waste), *International Journal of Environmental Science and Technology* 20 (2023).
- [3] M. Jansen, T. Meisen, C. Plociennik, H. Berg, A. Pomp, W. Windholz, Stop Guessing in the Dark: Identified Requirements for Digital Product Passport Systems, *Systems* 11 (2023) 123.
- [4] European Union, Regulation (EU) 2023/1542 of the European Parliament and of the Council of 12 July 2023 concerning batteries and waste batteries, <http://data.europa.eu/eli/reg/2023/1542/oj>, 2023. [Accessed 24-12-2023].
- [5] Plattform Industry 4.0, Progress report 2023 - industry 4.0: On the way to an intelligently networked industry, <https://www.plattform-i40.de/IP/Redaktion/DE/Downloads/Publikation/2023-fortschrittsbericht.html>, 2023. [Accessed 22-10-2023].
- [6] Industrial Digital Twin Association, Specification of the asset administration shell part 1: Metamodel, [https://industrialdigitaltwin.org/wp-content/uploads/2023/04/IDTA-01001-3-0\\_SpecificationAssetAdministrationShell\\_Part1\\_Metamodel.pdf](https://industrialdigitaltwin.org/wp-content/uploads/2023/04/IDTA-01001-3-0_SpecificationAssetAdministrationShell_Part1_Metamodel.pdf), 2013. [Accessed 22-10-2023].
- [7] A. Nazeri, C. Plociennik, M. Vogelgesang, C. Li, M. Ruskowski, A Novel Approach for Sensor Fusion Object Detection in Waste Sorting: The Case of WEEE (2023).
- [8] European Union, European Critical Raw Materials Act - ANNEX II: Critical raw materials, <https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:52023PC0160>, 2023. [Accessed 26-12-2023].
- [9] A. Pagoropoulos, D. C. Pigosso, T. C. McAlloone, The emergent role of digital technologies in the Circular Economy: A review, *Procedia cirp* 64 (2017) 19–24.
- [10] H. Salminen, M. Marjamaa, R. Tapaninaho, A. Heikkinen, L. Gonzalez Porras, J. Kujala, How do stakeholders understand sustainable circular economy-consensus or contradictions?, in: *International Society for Circular Economy (IS4CE) Conference*, 2020.
- [11] J. Walden, A. Steinbrecher, M. Marinkovic, Digital Product Passports as Enabler of the Circular Economy, *Chemie Ingenieur Technik* 93 (2021).
- [12] C. Plociennik, M. Pourjafarian, A. Nazeri, W. Windholz, S. Knetsch, J. Rickert, A. Ciroth, A. d. C. P. Lopes, T. Hagedorn, M. Vogelgesang, et al., Towards a digital lifecycle passport for the circular economy, *Procedia CIRP* 105 (2022) 122–127.
- [13] H. Li, M. Abd Nikooie Pour, Y. Li, M. Lindecrantz, E. Blomqvist, P. Lambrix, A survey of general ontologies for the cross-industry domain of circular economy, in: *2023 World Wide Web Conference, WWW 2023, Austin, Texas, USA, 30 April-4 May 2023*, ACM, 2023.
- [14] J. Mügge, J. Grosse Erdmann, T. Riedelsheimer, M. M. Manoury, S.-O. Smolka, S. Wichmann, K. Lindow, Empowering End-of-Life Vehicle Decision Making with Cross-Company Data Exchange and Data Sovereignty via Catena-X, *Sustainability* 15 (2023) 7187.
- [15] S. R. Bader, M. Maleshkova, The semantic asset administration shell, in: *Semantic Systems. The Power of AI and Knowledge Graphs: 15th International Conference, SEMANTiCS 2019, Karlsruhe, Germany, September 9–12, 2019, Proceedings 15*, Springer, 2019, pp. 159–174.

- [16] C. Bouter, M. Pourjafarian, L. Simar, R. Wilterdink, Towards a comprehensive methodology for modelling submodels in the industry 4.0 Asset Administration Shell, in: 2021 IEEE 23rd Conference on Business Informatics (CBI), volume 2, IEEE, 2021, pp. 10–19.
- [17] S. Rongen, N. Nikolova, M. van der Pas, Modelling with AAS and RDF in Industry 4.0, *Comput. Ind.* 148 (2023).
- [18] Y. Huang, S. Dhouib, L. P. Medinacelli, J. Malenfant, Semantic Interoperability of Digital Twins: Ontology-based Capability Checking in AAS Modeling Framework, in: 2023 IEEE 6th International Conference on Industrial Cyber-Physical Systems (ICPS), IEEE, 2023.
- [19] E. Järvenpää, N. Siltala, O. Hylli, M. Lanz, The development of an ontology for describing the capabilities of manufacturing resources, *Journal of Intelligent Manufacturing* 30 (2019) 959–978.
- [20] M. H. Rimaz, C. Plociennik, L. Kunz, M. Ruskowski, Am I in Good Shape? Flexible Way to Validate Asset Administration Shell Data Entry via Shapes Constraint Language, in: 2023 IEEE 28th International Conference on Emerging Technologies and Factory Automation (ETFA), IEEE, 2023, pp. 1–6.
- [21] A. Luxenburger, D. Porta, S. Knoch, J. Mohr, T. Schwartz, A Service Infrastructure for Industrie 4.0 Testbeds based on Asset Administration Shells, in: 2023 IEEE 28th International Conference on Emerging Technologies and Factory Automation (ETFA), IEEE, 2023.
- [22] T. Moreno, T. Sobral, A. Almeida, A. L. Soares, A. Azevedo, Semantic Asset Administration Shell Towards a Cognitive Digital Twin, in: *International Conference on Flexible Automation and Intelligent Manufacturing*, Springer, 2023, pp. 679–686.
- [23] M. Pourjafarian, C. Plociennik, M. H. Rimaz, P. Stein, M. Vogelgesang, C. Li, S. Knetsch, S. Bergweiler, M. Ruskowski, 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, 2023, pp. 1–8.

## A. Appendix

**Search Engine**

Recommended Queries based on your profile:

E-Waste per Year      Valuable E-Waste      Dangerous E-Waste

```
Filter(?property_idshort = "BuildYear")
Filter(?property_value_casted > "2020"^^xsd:int)
}
```

Search







Thumbnail	Name/ID	Actions	Options
	Manufacturer 3's Device - (2022)	<a href="#">View Shell</a> <a href="#">Add Submodel</a> <a href="#">Delete Shell</a>	 
	Manufacturer 4's Device - (2021)	<a href="#">View Shell</a> <a href="#">Add Submodel</a>	 

Figure 6: SPARQL search engine view with recommended queries.

Asset Administration Shell (AAS)

Content:

Device Specification

Export as Excel

Export as JSON

SM[co:manufacturer]... V 0.5 Instance

**Identification:** co:manufacturer\_0submodel\_device\_specification:31660fe2-370e-4437-9b6f-e7846ce0479b  
**Semantic ID:** dexdfki:submodeltemplates:DeviceSpecification:0.5

Version: 0, Revision: 5

ManufacturerName	Manufacturer 0	①
HaveScreen	false	①
HaveBattery	false	①
+ ExtantChemicalElements 89		
DeviceModel	Device Model 10	①
BuildYear	2007	①
Weight	2 kg	①

Figure 7: Asset Administration Shell viewer user interface.

```

1 [ruleIsChemicalHazardous:
2 (?submodel rdf:type aas:Submodel)
3 (?submodel <https://admin-shell.io/aas/3/0/Identifiable/id> ?submodel_id)
4 (?submodel <https://admin-shell.io/aas/3/0/Submodel/submodelElements> ?submodel_element)
5 (?submodel_element <https://admin-shell.io/aas/3/0/Referable/idShort> ?submodel_element_id_short)
6 equal(?submodel_element_id_short "Temperature")
7 (?submodel_element <https://admin-shell.io/aas/3/0/Property/value> ?submodel_element_value)
8 # Conditions
9 greaterThan(?submodel_element_value,80)
10 uriConcat(?submodel_id, "/elements/IsHazardousInferred",?newid)
11 →
12 (?submodel chameo:hasHazard chameo:ChemicalHazard)
13 (?submodel <https://admin-shell.io/aas/3/0/Submodel/submodelElements> ?newid),
14 (?newid rdf:type aas:Property),
15 (?newid <https://admin-shell.io/aas/3/0/Referable/idShort> "IsDangerous"^^xsd:string),
16 (?newid <https://admin-shell.io/aas/3/0/Property/valueType> <https://admin-shell.io/aas/3/0/DataTypeDefXsd/Boolean>),
17 (?newid <https://admin-shell.io/aas/3/0/Property/value> "true"^^xsd:boolean)
18 ]
19
20 [submodelChemicalRelation:
21 (?submodel chameo:hasHazard chameo:ChemicalHazard)
22 (?submodel chameo:hasHazard chameo:BiologicalHazard)
23 →
24 (?submodel chameo:hasHazard chameo:RecyclingHazard)
25 ]

```

Figure 8: Example rule expressed in Apache Jena's rule syntax to identify and categorize hazards.

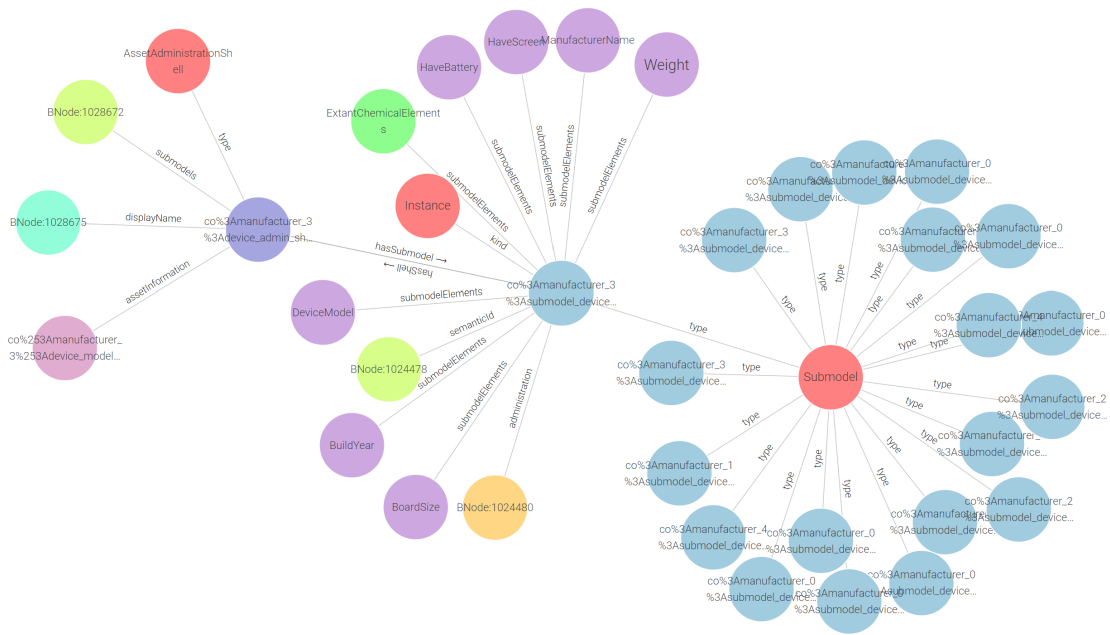


Figure 9: Ontotext GraphDB's visual graph interface.

```

1 PREFIX aas: <https://admin-shell.io/aas/3/0/>
2 PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
3 select Distinct ?aas_id ?global_asset_id ?aas
4 where {
5   ?aas a aas:AssetAdministrationShell .
6   ?aas <https://admin-shell.io/aas/3/0/Identifiable/id> ?aas_id .
7   ?aas aas:AssetAdministrationShell/assetInformation ?asset_info .
8   ?asset_info aas:AssetInformation/globalAssetId ?global_asset_id .
9
10  ?aas <https://admin-shell.io/aas/3/0/AssetAdministrationShell/submodels> ?a_submodel_ref .
11  ?a_submodel_ref aas:Reference/keys ?keys .
12  ?keys aas:Key/value ?ref_value .
13  ?submodel a aas:Submodel .
14  ?submodel aas:Identifiable/id ?submodel_id .
15
16  #check that submodel and aas belongs together
17  Filter(?ref_value = ?submodel_id)
18
19  ?submodel <https://admin-shell.io/aas/3/0/HasSemantics/semanticId> ?submodel_semantic .
20  ?submodel_semantic aas:Reference/keys ?submodel_semantic_keys .
21  ?submodel_semantic_keys aas:Key/value ?submodel_semantic_keys_ref_value .
22
23  # Filter to specific submodel with specific semantic id : Semantic Matching
24  # Check that submodel and aas belongs together
25  Filter(?submodel_semantic_keys_ref_value = "de:dfki:submodeltemplates:DeviceSpecification:0:5")
26  # (!<:>)+ any path!
27  ?submodel (!<:>+ ?property .
28  ?property a aas:Property .
29  ?property aas:Referable/idShort ?property_idshort .
30  ?property aas:Property/value ?property_value .
31  BIND(xsd:double(?property_value) as ?porperty_value_casted)
32  # Property idshort matching. Can be semantic matching
33  # "Precious Material" is variable Aluminium
34  Filter(?property_idshort IN ('Arsenic', 'Lead', 'Mercury'))
35  Filter(?porperty_value_casted > "14"^^xsd:double)
36 }

```

Figure 10: Example SPARQL query to identify hazardous assets based on the amount of specific materials.