Role-based Confidentiality Protection for Digital Product Passports: The champI4.0ns Supply Chain Use Case

Albin Ahmeti[1,3,*], Robert David[1,2] and Artem Revenko[1]

[1] *Semantic Web Company GmbH (Graphwise), Austria*
[2] *Vienna University of Economics and Business, Austria*
[3] *Vienna University of Technology, Austria*

Abstract

Digital Product Passports (DPPs) are being introduced as a core instrument of the EU's circular economy strategy. Regulations such as the Ecodesign for Sustainable Products Regulation (ESPR) specify transparency requirements for product information on the European market. Beyond compliance, DPPs can improve data management and enable new services and business models built on shared product data. Because DPPs are intended to be used by multiple stakeholders—including consumers, recyclers, regulators, and market surveillance authorities—practical deployments must protect confidential business information and support differentiated access based on stakeholder roles. To implement DPPs at scale, semantic interoperability and standardized data models are needed to ensure unambiguous interpretation across organizations. Knowledge graphs (KGs) and standards such as OWL and SPARQL provide a foundation for machine-processable, interoperable DPP data. In the champI4.0ns project, we investigate wood production value chains where DPP data is exchanged among suppliers in a data space. We present a KG-based solution for DPP data governance, integrating a trust model for role-based confidentiality protection while remaining independent of specific DPP schemas through configurable mappings. This work supports data sovereignty and contributes to interoperable DPP solutions in EU data spaces and beyond.

Keywords

Circular Economy, Sustainability, Supply Chain, ESPR, Digital Product Passport, Data Spaces, Knowledge graphs

1. Introduction

The Digital Product Passport (DPP) is a key element for achieving sustainability of resources and enabling an effective circular economy. Many resources—i.e., products including their parts and materials—on the European market are not reused or recycled, or they are even destroyed without use at all. This inefficient resource management leads to both, economic and environmental drawbacks, which could potentially be avoided by appropriate resource management. However, a prerequisite for achieving this is the *transparent sharing* of product information, which is relevant to such processes, namely a list of materials contained in a specific product.

The ESPR [1] defines ecodesign and information requirements for DPPs as a foundation, which can be further specified and expanded through delegated acts targeting individual product groups. Basically, DPPs describe a product's properties relevant to this purpose of circular economy: (i) Product data: *Materials, components, specs, certifications*; (ii) Lifecycle tracking: *Usage, maintenance, repairs, modifications, ownership changes*; (iii) Sustainability and environmental impact: *Carbon footprint, energy usage, sustainability metrics*; (iv) End-of-life options: *Guidelines for disposal/recycling, information on recycling, refurbishing, repurposing*. DPP data are shared among several stakeholders, such as members of a supply chain or vendors that focus on the reuse of end-of-life products. To achieve a clear understanding of the data, i.e., an unambiguous meaning, semantic data models and controlled vocabularies can be

The Workshop on Knowledge Graphs for Sustainability (KG4S). Co-located with the 23rd European Semantic Web Conference (ESWC), May 10 – 14, 2026, Dubrovnik, Croatia
*Corresponding author.
albin.ahmeti@graphwise.ai (A. Ahmeti); robert.david@graphwise.ai (R. David); artem.revenko@graphwise.ai (A. Revenko)
0000-0001-8766-4069 (A. Ahmeti); 0000-0002-3244-5341 (R. David); 0000-0001-6681-3328 (A. Revenko)

[1] https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A32024R1781

applied in accordance with the ESPR's requirement of a standards-based representation. There are several initiatives in the scope of modeling product data in a way to achieve standardized descriptions for shared use. The United Nations Transparency Protocol (UNTP) and the Asset Administration Shell (AAS) are two main initiatives, whereas various projects develop further ontologies around DPPs. For example, Circular Economy Ontology Network (CEON) is a network of ontologies, which are used in the champI4.0ns project [2] and presented in this paper. Details for related DPP data representation are described in Sec. 2. The approach we present in this paper can be easily configured to use different schemas as well, by specifying accordingly the alignment via mappings.

DPPs are intended to be shared among various stakeholders, from being publicly available for all customers to various degrees of confidentiality restrictions based on stakeholder roles, e.g., in a supply chain scenario. This is particularly important in practice because the ESPR requires publication, meaning that trade secrets must be protected and economic operators must retain full control over confidentiality while sharing data. To support DPP confidentiality protection along supply chains, we introduce an approach which implements standardized role-based security to restrict access to confidential parts of a DPP. We formally specify access rights via ODRL and introduce an approach to automatically enforce confidentiality restrictions by methods of *consistent query answering (CQA)* via *query rewriting* [1] leveraging ODRL vocabulary. Furthermore, we show how a statistical "binning" approach to approximate values can be implemented to replace confidential details without exposing confidential details.

By establishing access restrictions, we implement towards automatic DPP data management in supply chain data spaces used in the champI4.0ns pilots [2]. Our approach, however, is not specific to the project use cases and can easily be ported to different data models, data spaces and use cases. This work supports data sovereignty—data owners retain control over access to their data in shared scenarios—and contributes to interoperable DPP solutions in EU data spaces and beyond.

To summarize, our contributions are as following:

- The creation of champI4.0ns DPP Knowledge Graph (KG) and mapping to CEON in order to enable interoperability;
- The modeling of different levels of access control rights using the ODRL vocabulary;
- An algorithm and implementation of consistent query answering via query rewriting approach.

The remainder of the paper is structured as follows. Sec. 2 presents background and related work. Sec. 3 introduces the champI4.0ns KG and its mapping to CEON. Sec. 4 provides examples of ODRL expressions for different access-granularity models using the champI4.0ns KG. Sec. 5 describes an algorithm for query rewriting based on the ODRL vocabulary. Finally, Sec. 6 concludes the paper.

2. Background

ODRL formalisation As per ODRL Information Model [3], *Rules* can be specified as *Permissions*, *Prohibitions* or *Duties*. We don't consider *Duties* as they are not applicable in our DPP context. We denote *Permissions* (*Prohibitions*) with *PER* (*PRH*) respectively. We further specify that *PER* and *PRH* are mutually disjoint, clashing operations when applied on the same target asset leading to inconsistency. In the context of DPPs for action we consider and use the "read" operation only as it is commonly applied in the most of DPP use cases. In the following, we formalize several basic ODRL constructs that are used in the context of DPP modeling and access control mechanisms. In Sec. 4, we subsequently define these constructs in terms of ODRL modeling.

Definition 1. *The Rule* Permission *PER (*Prohibition *PRH) applies to user u performing action A_1 on target asset T, we formalise as following:*

$$PER^{u}_{A_1}(T)(PRH^{u}_{A_1}(T))$$

[2]https://www.champi40ns.eu/

[3]cf. Fig. 3 in Appendix.

In the context of *PartyCollection* modeling in ODRL, a role r is formalized as a set of users u_j:

$$PER^{r}_{A_1}(T) \equiv PER^{\sum_{u_j \in r} u_j}_{A_1}(T)$$

In the context of *AssetCollection* modeling in ODRL, an action A_1 can be applied on a set of assets T_i for the user u.

$$PER^{u}_{A_1}(\sum T_i)$$

Definition 2. *We define a* conflict *as the case where two pairwise disjoint rules (PER, PRH) are applied to the same user u, for the same action A_1 on the same target T:*

$$PER^{u}_{A_1}(T) \wedge PRH^{u}_{A_1}(T) \Rightarrow \perp$$

Consequently, permission is canceled by prohibition if applied sequentially on the same target asset, action and user.

Definition 3. *The constraint that applies to the user u that performs the action A_1 on target asset T only by satisfying the logical condition (l_o, op, r_o), where l_o = left operand, op = operator, r_o right operand as per ODRL definition.*

$$PER^{u}_{A_1}(T) \wedge (l_o, op, r_o)$$

The logical condition in the above formula should be satisfied, in order the permission to be applicable:

$$\models (l_o, op, r_o)$$

The above definition holds also for prohibitions. Further, the above definition can be (recursively) extended to a logical condition consisting of expression using a set of boolean conjunctives, namely $\wedge, \vee, \veebar$ (in ODRL `odrl:and, odrl:or, odrl:xone` resp.); as well as operators like $=, \neq, <, >, \leq, \geq$ (in ODRL `odrl:eq, odrl:neq, odrl:lt, odrl:gt, odrl:lteq, odrl:gteq` resp.).

Related Work In the following, we provide an approach for access control which is easily adoptable and portable between various schemas, utilizing any formal alignment, interlinking and RDF representation of DPP data might provide. These and other schemas and ontologies can be used to represent DPP information.

DPP Data Models Based on the ESPR definitions, DPP information must be provided according to existing established standards for data modeling. A way to represent this information using both syntactic and semantic standards is the W3C Semantic Web stack. The *UN Transparency protocol* (UNTP) [3] uses JSON-LD as the serialization format. Other well-established models, like the *Asset Administration Shell* (AAS) [4] do not use semantically defined schemas, but cover semantic alignment by using controlled vocabularies. Furthermore, there are initiatives and ongoing work aimed at the semantic alignment and interoperability of these different DPP modeling approaches [5].

Consistent Query Answering *Consistent query answering* (CQA) is an approach in which queries to a database return only results that are consistent with a given set of constraints. CQA is used in the context of (possibly) inconsistent databases to be able to return consistent results on query time without the need to change (repair) the whole database. To achieve this, integrity constraints are introduced into the query itself and are used to implicitly filter out inconsistent results from the result set. Since the seminal paper [6], a large body of work developed, including applications of CQA for OBDA [7] and Description Logics [8]. In the context of this paper, we apply CQA not to filter out inconsistent data, but rather to filter out data where users do not have permission to read.

3. ChampI4.0ns Knowledge Graph

In this section we describe the creation process of the champI4.0ns KG along with its building blocks [4]. The workflow detailing the inputs required for the creation of the KG is presented in the bottom section of Fig. 1. An initial preliminary version of the KG was reported in [9]; however not published online. We describe the application of the KG in the later sections in regards to filtering information (information hiding) via ODRL modeling and query rewriting, see top section of Fig. 1 for an overview.

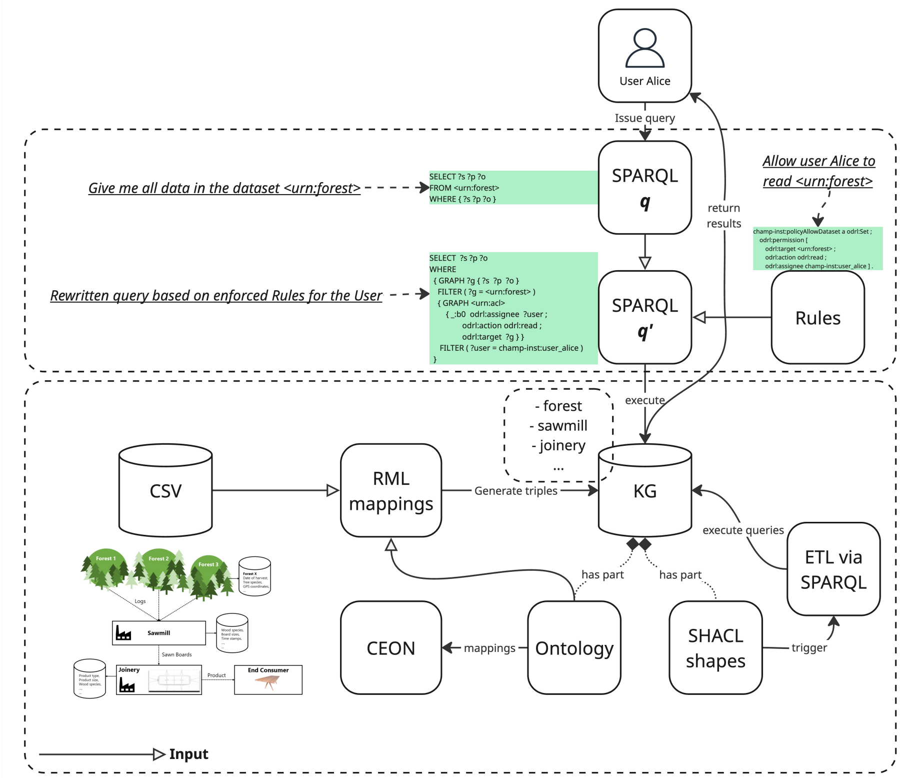

Figure 1: The workflow describing the steps required for the creation of the KG from the data sources (bottom). The query rewriting with respect to the Rules and the current User (top).

Data sources The furniture production process chain comprises multiple independent stakeholders, each maintaining a DPP whose information is passed to downstream actors. The process starts from the lumber harvesting and generation of logs, to the production of boards from the logs in the sawmill, and finally to the production of furniture from boards in the joineries. The data created in XLS (spreadsheet) format per each stakeholder is synthetic mimicking this process chain—forest, transport, sawmill arrival, sawmill output, joinery arrival, joinery product—and validated through consultations with industrial experts. The spreadsheets have been pre-processed and exported as a set of CSV files (cf. Fig. 1).

Ontology & Alignment The creation of the champI4.0ns ontology is informed from the source data, in the bottom-up fashion, containing the following ontological terms, namely Classes: *Product, Material, Transport, DPP, Board, Logs*, and more; Relations: *isUsedToProduce, refersToTransport, refersToLogTransport*, and more; Datatype properties: *length, width, thickness, woodSpecies, qualityClass, certification, storagePlace, wasteFromTrunkInPercentage, maintenanceInstructions*, and more.

In order to enable interoperability we have chosen CEON [10] as an upper-level ontology for the alignment, linking the DPP Material and Product classes to CEON core classes via `rdfs:subClassOf`

[4]The link to KG and its implementation: https://github.com/aahmeti/champI4.0ns

relations. CEON provides domain-independent ontology design patterns that support cross-domain interoperability, as illustrated in Fig. 2.

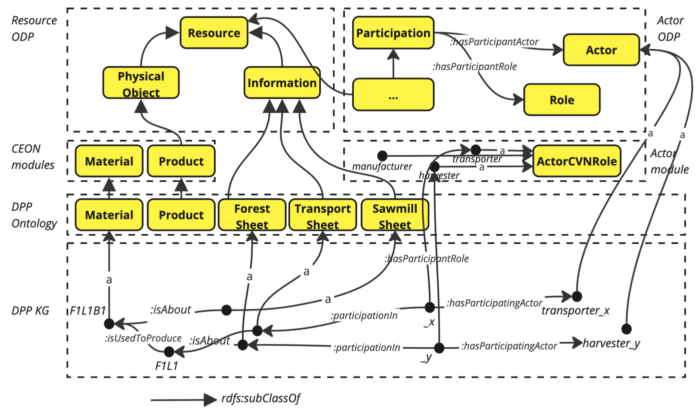

Figure 2: Ontology alignment with CEON [9].

SHACL shapes for data quality and completeness The spreadsheets alongside the data contained information on the value domains, enumerating all possible values for that particular column. This input was useful in creating SHACL shapes in order to check for data consistency of respective column values. For instance, the "forest" spreadsheet contained the value domains for *size class* column the resp. values: D2a, D2b, D3a, D3b, D4, D5, D6. Those values were used to create the SHACL shapes using `sh:in` operator. Most of the SHACL shapes are expressed using SHACL Core, with only a few using SHACL-SPARQL (e.g., a shape that checks if the label is concatenation of forest and log labels resp.). The advanced features of SHACL were used in order to associate the named graphs to SHACL shapes so that the validation is "contained" and not applied in all the graphs, as an example `[ a rsx:DataAndShapesGraphLink; rsx:shapesGraph <urn:shapes-joinery>; rsx:dataGraph <urn:joinery-product>].`

Knowledge Graph The instance data of champI4.0ns KG is created by applying RML mappings to the data sources (CSV files), resulting in triples stored in separate named graphs per each data source. An ETL (Extract–Transform–Load) process, implemented via a set of SPARQL update queries, is used to evolve the KG, for example after validation with SHACL shapes has been performed. In addition, the update queries are used to link the entities contained in the named graphs (via "foreign keys"), namely tracing of logs from boards, and boards from furniture products (cf. Fig. 4 in Appendix). The KG is capable of answering a set of Competency Questions related to the dimensions of traceability, quality, certification, and other production details in the context of DPPs. For example: *From which coordinates was the log (Raw Material) harvested for the table (Product)? How many steps in the value chain are involved in the production of the sideboard (Product)? Which boards (Material) have a higher quality than that of their respective logs (Raw Material)?*

The mapping to CEON enables to ask CQs in terms of more generic constructs, to name a few: *How many actors (stakeholders) exist and what is their participation role? How many resources exist in total?*

4. ODRL-based Access Control Rights for DPP

There have been proposals of using ODRL to express access policies for Linked Data [11]. Herein, we go further by reusing ODRL to model permissions/prohibitions in the context of DPPs and applying it in the case of champI4.0ns KG. Based on the proposed modeling we provide query rewriting techniques (cf. Sec. 5), which ensure the correct access of assets returned to the user.

4.1. Access Control Modeling with ODRL

We start with the assumption that nothing is allowed unless permission is granted explicitly, reflecting the security principles of *least privilege*. Starting from this premise, the permissions for a certain resource can be granted, and afterwards revoked again with prohibitions if necessary – these two operations canceling each other if applied in the specified order (cf. Def. 2). In the following we model four main scenarios starting with the highest granularity of access to the lowest, namely access on the level of *(i) named graphs (ii) property(-ies) (iii) property values (iv) property values with boolean conditions*. The precedence of access granularity ensures that permissions (prohibitions resp.) defined at the named graph level override those defined at the property level, which in turn override permissions (prohibitions resp.) defined at the property-value level, and so forth.

In the following, we show examples of different levels of granularity using `odrl:permission` and `odrl:prohibition` respectively.

Allow/Prohibit Dataset We illustrate how, using ODRL, a `odrl:read` permission (prohibition resp.) can be granted to a user (`user_alice`) for a specific dataset (named graph).

```
champ-inst:policyAllowDataset a odrl:Set ;
    odrl:permission [  %odrl:prohibition
                odrl:target <https://data.champi40ns.eu/forest> ;
                odrl:action odrl:read ;
                odrl:assignee champ-inst:user_alice ] .
```

Allow/Prohibit Property The following ODRL model demonstrates how to grant the `odrl:read` permission for the `moistureContentInPercentage` property to the users `user_alice` and `user_bob`. In order to constrain the policy to a specific set of users—effectively defining their role membership—we group them using an `odrl:PartyCollection` and connect the individuals via `vcard:hasMember`. Furthermore, to constrain the permission to a specific property, we create an `odrl:Asset` specifying the parent named graph with `odrl:partOf`, and specifying the target predicate with `rdf:predicate` [5].

```
champ-inst:HarvesterRole a odrl:PartyCollection ;
    vcard:hasMember champ-inst:user_alice ,
                champ-inst:user_bob .

champ-inst:Forest_prop1_Asset a odrl:Asset ;
    odrl:partOf <https://data.champi40ns.eu/forest> ;
    rdf:predicate champ-onto:moistureContentInPercentage .

champ-inst:policyAllowDatasetProperty a odrl:Set ;
    odrl:permission [
        odrl:target champ-inst:Forest_prop1_Asset ;
        odrl:action odrl:read ;
        odrl:assignee champ-inst:HarvesterRole ] .
```

This modeling approach facilitates the specification of many-to-many relationships; by bridging a `PartyCollection` with an aggregated `Asset`, we can grant *n* users access to *m* properties using a single policy rule.

Allow/Prohibit Property with an object-level constraint We extend the previous modeling where we qualify the constraint to be a certain property filtered on the level of the object values (e.g., `wasteInPercentage` *object values should be smaller or equal than 50*). We use `odrl:refinement` to represent this logical constraint along with the condition needed to be expressed for the property object values. In the following example we use `odrl:lteq`, but in general we can also use other available constructs `odrl:eq, odrl:neq, odrl:gteq, odrl:gt, odrl:lt`.

```
champ-inst:ProductView_1 a odrl:Asset ;
    odrl:partOf <https://data.champi40ns.eu/joinery-product> ;
```

[5] We adopt `rdf:predicate` from the RDF reification vocabulary defined in the RDF specification, where it is used to describe the predicate component of a reified statement.

```
    odrl:refinement [
        odrl:leftOperand champ-onto:wasteInPercentage ;
        odrl:operator odrl:lteq ;
        odrl:rightOperand 50 ] .

champ-inst:policyAllowDatasetProperty a odrl:Set ;
        odrl:permission [
        odrl:target champ-inst:ProductView_1 ;
        odrl:assignee champ-inst:user_alice ;
        odrl:action odrl:read ] .
```

Allow/Prohibit Property with an object-level constraint and boolean conditions The previous condition for the object values can be extended to an arbitrary set of logical conditions, which can be expressed using operators such as `odrl:and`, `odrl:or`, `odrl:xone` – as per ODRL specification. In the following modeling we are explicitly expressing that the condition is values between 30 and 40, for the designated property[6]. Note that we use the RDF List in order to express such condition.

```
champ-inst:ProductView_2 a odrl:Asset ;
    odrl:partOf <https://data.champi40ns.eu/joinery-product> ;
    odrl:refinement [
        odrl:and (
            [
              odrl:leftOperand champ-onto:wasteInPercentage ;
              odrl:operator odrl:lteq ;
              odrl:rightOperand 40
            ]
            [
              odrl:leftOperand champ-onto:wasteInPercentage ;
              odrl:operator odrl:gteq ;
              odrl:rightOperand 30
            ]) ] .

champ-inst:policyDisallowDatasetProperty a odrl:Set ;
    odrl:prohibition [
        odrl:target champ-inst:ProductView_2 ;
        odrl:assignee champ-inst:user_alice ;
        odrl:action odrl:read ] .
```

The previous boolean expression with `odrl:and` and represented using `rdf:List` can be recursively extended to an expression with multiple operators by using an arbitrary number of nested lists.

4.2. Information approximation with ODRL

In the context of DPPs, certain use cases restrict stakeholders to information at an approximate level of detail rather than the exact values. This is analogous to the statistical binning approach, in which numerical data are grouped into categorical ranges instead of being reported as precise numbers. To represent information approximation within ODRL, we model each approximated attribute as an `odrl:Asset` and specify that access to the corresponding property requires the application of a transformation. This transformation is implemented as a SHACL rule expressed via a CONSTRUCT query and associated with the asset via `sh:rule`. For clarity and traceability, the transformed property is materialized using a new predicate obtained by appending the suffix `_rew` to the original predicate and is permitted to the user (cf. Sec. 5). This approach complements access control with dynamic data generation, where the semantics of the ODRL permission allow access to values returned by a SHACL rule. While ODRL provides permission semantics, the SHACL rule provides the operation to generate the permitted data.

```
champ-inst:policyTransformProperty a odrl:Set ;
  odrl:permission [
    odrl:assignee champ-inst:user_alice ;
    odrl:target champ-inst:Sawmill_waste_transformed ;
    odrl:action odrl:read
  ] .
```

[6]Note that creating two separate policies with rules to capture the respective conditions is equivalent to using an `odrl:or` operator, as will be illustrated in the translation via query rewriting with UNION in Sec. 5.

```
champ-inst:Sawmill_waste_transformed a odrl:Asset ;
    odrl:partOf <https://data.champi40ns.eu/sawmill-output>;
    rdf:predicate champ-onto:wasteFromTrunkInPercentage_rew ;
    sh:rule champ-inst:RewriteRule_Binning .

champ-inst:RewriteRule_Binning
    a sh:SPARQLRule ;
    sh:construct """
        CONSTRUCT { ?this champ-onto:wasteFromTrunkInPercentage_rew ?category .}
        WHERE { ?this champ-onto:wasteFromTrunkInPercentage ?value .
            BIND(IF(?value > 0.8 && ?value <= 1.0, "A",
                    IF(?value > 0.6 && ?value <= 0.8, "B",
                        IF(?value > 0.4 && ?value <= 0.6, "C",
                            "D"))) AS ?category) }""" .
```

5. Algorithm for Query Rewriting

Considering the modeling of permissions/prohibitions presented previously, the challenge is to translate them so that they are enforced in the implementation of a DPP system [2], in which SPARQL is used for retrieval. We employ consistent query answering [1] in order to translate and enforce permissions and prohibitions expressed in ODRL. Analogue to query rewriting based on integrity constraints, our approach rewrites the original SPARQL query according to the set of ODRL permissions and prohibitions, ensuring that the user ultimately accesses only the resources they are authorized to. The rewriting of the SPARQL query that is issued to access a dataset, e.g., a 'sawmill' stakeholder accessing the 'forest' dataset, is rewritten based on the original ODRL vocabulary.

Alg. 1 describes the procedure of how the query is rewritten in respect to the enforced rules and the current user respectively [7]. The algorithm rewrites an input SPARQL query according to a set of access control rules in order to enforce permission and prohibition constraints for a given user. Given a user *u*, a set of access control rules (hereafter *ACL*), and an input SPARQL query *q*, the algorithm first extracts the target dataset d from the query and normalizes the query in order to enable joining with the rewritten query, for instance: `SELECT ?s ?p ?o WHERE { GRAPH <urn:forest> { ?s ?p ?o } }`[8] is normalised to the equivalent query `SELECT ?s ?p ?o WHERE { GRAPH ?g { ?s ?p ?o } FILTER (?g = <urn:forest>}`. If no permission rule exists in the rule set, the algorithm immediately returns an empty query, ensuring that no data can be accessed in the absence of explicit permissions. Next, the algorithm iterates over all rules in *ACL*. For each rule *acl*, it checks whether the rule is applicable to the given user *u* and the target dataset *d* - one can use ASK queries to do the checking[9]. Applicable rules are then classified according to their type. If the rule represents a permission, it is translated into a SPARQL pattern using the function *odrl2SPARQL* and added to the set of permission patterns via `UNION`. If the rule represents a prohibition, its translated SPARQL pattern is added as a `FILTER NOT EXISTS` clause to the set of prohibition patterns. After all rules have been processed, the algorithm combines the permission and prohibition patterns using a join operation to form the rewritten access control query fragment by using a separate named graph. This fragment is then joined with the original normalized query *q* to produce the final rewritten query q', which enforces the access control constraints at query execution time. In the end, the resulting query q' is returned as output.

Theorem 1. *Consider user u, and Rules acls represented using ODRL vocabulary. Then, for the SPARQL query q, Alg. 1 computes in polynomial time the rewritten query* q' *based on the size of acls.*

[7]The implementation of the query rewriting algorithm in Java using Jena API can be found in the same Github repository: https://github.com/aahmeti/champI4.0ns

[8]Same applies if we have the equivalent query: `SELECT ?s ?p ?o FROM <urn:forest> WHERE { ?s ?p ?o }`

[9]For instance, the query to check for the graph-level granularity: `ASK WHERE { ?x odrl:assignee*/vcard:hasMember ?user ; odrl:target ?graph. FILTER (?user=champ-inst:user_alice) FILTER (?graph = <urn:forest>) }`

Algorithm 1 SPARQL Query Rewriting Based on Access Control Rules

Input: User u, Rules *acls*, SPARQL query q = `SELECT * WHERE {GRAPH <_> {?s ?p ?o}}`
Output: Rewritten SPARQL query q'
Dataset $d \leftarrow$ *getDataset(q)*
$q \leftarrow normalise(q)$
if $acls_{perms} = \emptyset$ **then**
    return `SELECT * WHERE {}` // Return a query with an empty solution set
end if
$perms \leftarrow \{\}; prohs \leftarrow \{\}$
for Rule *acl* in *acls* **do**
    if *acl is applicable in u and target d* **then**
        if *acl is PERM* **then**
            $perms \leftarrow perms \cup \{$`UNION` *odrl2SPARQL(acl)*$\}$ // See Algorithm 2.
        end if
        if *acl is PRH* **then**
            $prohs \leftarrow prohs \cup \{$`FILTER NOT EXISTS` *odrl2SPARQL(acl)*$\}$ // See Algorithm 2.
        end if
    end if
end for
$q'' \leftarrow$ `GRAPH <urn:acl>` { *perms* . *prohs* } // Concatenate using "." to JOIN perms and prohs
$q' \leftarrow InsertIntoWhere(q, q'')$ // Join with original query
return q'

ODRL-SPARQL Mapping Considering Alg. 1, the remaining task is to convert the ODRL modeling to SPARQL triple patterns, as indicated with the function $odrl2SPARQL(acl)$. The function should check if the filtered acls are applicable in four different level of granularity of access.

We use θ as a helper function (the notation adopted from update rewriting [12]) that replaces entities with variables, for instance $acl\ \theta[\_ \rightarrow ?user, \_ \rightarrow ?g]$ replaces the respective graph name (*odrl:target*) _ with $?g$ and respective user (*odrl:assignee*) _ with $?user$. Similarly by building on this application rule, the replacement is done for other level of access granularity $acl\ \theta[\_ \rightarrow ?user, \_ \rightarrow ?g, \_ \rightarrow ?p]$.

Algorithm 2 implements the function $odrl2SPARQL(acl)$, namely it converts an access control rule into a SPARQL triple pattern by analyzing the rule's target granularity. Depending on whether the rule targets a graph, a property, or a property-value constraint, it constructs the corresponding variable bindings and filter expressions. Comparison operators are translated into SPARQL operators, and the resulting triple pattern is returned [10]. In addition, if the rule targets a rewritten property rule, the algorithm restricts the predicate variable to the rewritten property, as per information approximation (cf. Sec. 4.2).

Proposition 1. *Consider Rules that have conflicting permissions and prohibitions, then Alg. 1 will filter out such conflicting inconsistencies as per the* `SPARQL` *semantics [13] of* `JOIN` *and* `FILTER NOT EXISTS` *by removing such bindings.*

Proposition 1 indicates that the adopted resolution semantics via query rewriting resolve conflicts by rejecting access to the affected resource. Therefore, when a conflict arises, access is forbidden and prohibitions override permissions. If the same permission is recorded subsequently, then we have to retract the prohibition, i.e., INSERT is substituted with a DELETE/INSERT operation. Another way to circumvent this is to record provenance timestamps to every permission/prohibition and then use it as an additional FILTER in the query rewriting to check for the latest pair of permissions and prohibitions. This is left open for the future work.

Definition 4. *Prohibition is overridden by permission (or vice versa) if applied sequentially on the same target asset, action and user. Formally, let $t(\cdot)$ denote the assertion time of a rule. Then,*

$$PER^{u}_{A_1}(T) \succ PRH^{u}_{A_1}(T) \quad iff \quad t(PER) > t(PRH)$$

This notion of precedence requires an explicit temporal assertion, e.g., through provenance metadata.

[10] Ex. 1 in Appendix illustrates the rewriting.

Algorithm 2 *odrl2SPARQL*(*acl*): Translation of ODRL modeling to SPARQL triple patterns

Input: Rule *acl*
Output: SPARQL triple pattern q with variables
$q \leftarrow \{\}$
if *acl has graph target* **then**
  $q \leftarrow acl\ \theta[\_ \rightarrow ?user, \_ \rightarrow ?g]$ // The whole graph is permitted/prohibited
else if *acl has property targets* **then**
  $q \leftarrow acl\ \theta[\_ \rightarrow ?user, \_ \rightarrow ?g, \_ \rightarrow ?p]$
  $preds \leftarrow \{\}$
  for *pred in* `rdf:predicate` object values **do**
    $preds \leftarrow preds \cup \{pred\}$
  end for
  $q \leftarrow q\ .\ \texttt{FILTER}(?p\ \texttt{IN}\ preds)$
else if *acl has property value target* **then**
  $q \leftarrow acl\ \theta[\_ \rightarrow ?user, \_ \rightarrow ?g, \_ \rightarrow ?p]$
  for Constraint *constr* in *acl* **do**
    $prop \leftarrow$ `odrl:leftOperand`
    $op \leftarrow translate($`odrl:operator`$)$ // "<" for `odrl:lt`; ">" for `odrl:gt`, ...
    $value \leftarrow$ `odrl:rightOperand`
    $q \leftarrow q.\ \texttt{FILTER}((?p = prop) \&\& (?o\ op\ value))$
  end for
  if *acl* has `odrl:refinement` *cond* **then**
    $q \leftarrow ApplyBooleanCondition(q, cond)$ // && for `odrl:and`, || for `odrl:or`, ...
  end if
else if *acl has property rule target* **then**
  $prop_{rew} \leftarrow$ `rdf:predicate`
  $q \leftarrow q\ .\ \texttt{FILTER}(?p = prop_{rew})$ // Return the computed (rewritten) property.
end if
return q

6. Conclusions & Future Work

In this paper, we addressed a practical challenge at the core of DPPs: enabling standards-based, semantically unambiguous sharing of product information across many stakeholders while protecting confidential business information in distributed supply-chain data sharing scenarios. Building on Semantic Web and knowledge-graph standards, we presented an approach that remains independent of any specific DPP schema and can be configured through mappings to existing initiatives such as CEON. Our main contribution is an integrated trust model for role-based confidentiality protection, where access rights are formally expressed using ODRL and automatically enforced through consistent query answering via query rewriting. In addition, we outlined how confidentiality can be preserved not only by restricting access, but also by providing approximations through statistical binning when appropriate. While this approach represents a step towards a functional foundation for regulatory-aligned DPP deployments (such as under the ESPR), several areas require further investigation. Our approach is based on "read" operations only and the Access Control Modeling (Sec. 4) grounded in ODRL formalisation (Sec. 2), which is using a subset of ODRL; *Conflict Resolution* - the system would benefit from static checking and automated reasoning to detect policy conflicts during the specification phase. For future work, we will focus on *Scalability and Performance* evaluation of complex query rewritings within large-scale, operational data spaces. Ultimately, this KG-based governance model strengthens data sovereignty by allowing operators to retain control over what information is shared and with whom, contributing to the development of interoperable and secure EU data spaces.

Declaration on Generative AI

During the preparation of this work, the authors used ChatGPT and Gemini in order to: grammar and spelling check, paraphrase and reword. After using this tool/service, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

Acknowledgments

We thank Sarah Ritter from the champI4.0ns research project for kindly providing us the data sources.

References

[1] L. Bertossi, Database repairs and consistent query answering: Origins and further developments, in: Proceedings of the 38th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS '19, Association for Computing Machinery, New York, NY, USA, 2019, p. 48–58.

[2] A. Belova, V. Gallina, A. Revenko, A. Ahmeti, Digital product passport: Initial system architecture with knowledge graphs and data spaces, in: E. Blomqvist, R. García-Castro, D. Hernández, P. Hitzler, M. Lindecrantz, M. Poveda-Villalón (Eds.), Proceedings of The 3rd International Workshop on Knowledge Graphs for Sustainability (KG4S 2025) co-located with the 22nd Extended Semantic Web Conference (ESWC 2025), Portoroz, Slovenia, June 1st, 2025, volume 4002 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2025, pp. 60–66.

[3] UNECE, UN Transparency Protocol (UNTP) - Specification, Technical Specification, United Nations Economic Commission for Europe, 2025. URL: https://untp.unece.org/docs/about/, accessed: 2026-03-05.

[4] S. Bader, E. Barnstedt, H. Bedenbender, B. Berres, M. Billmann, B. Boss, A. Braunmandl, E. Clauer, C. Diedrich, B. Flubacher, W. Fritsche, K. Garrels, A. Gatterburg, M. Hankel, O. Hillermeier, M. Hoffmeister, M. Jochem, A. Köpke, Y. Kogan, C. Ziesche, Details of the asset administration shell. part 1 - the exchange of information between partners in the value chain of industrie 4.0 (version 3.0rc01), 2020.

[5] G. Havur, T. Pellegrini, G. Schenner, F. Fusco, A design study on embedding untp semantics into aas submodels for interoperable digital product passports, in: 2025 21st International Conference on Network and Service Management (CNSM), IEEE, 2025, pp. 1–4.

[6] M. Arenas, L. E. Bertossi, J. Chomicki, Consistent Query Answers in Inconsistent Databases, in: Proc. of PODS, ACM Press, 1999, pp. 68–79. URL: https://doi.org/10.1145/303976.303983. doi:10.1145/303976.303983.

[7] M. Bienvenu, R. Rosati, Tractable Approximations of Consistent Query Answering for Robust Ontology-based Data Access, in: F. Rossi (Ed.), IJCAI, IJCAI/AAAI, 2013. URL: http://www.aaai.org/ocs/index.php/IJCAI/IJCAI13/paper/view/6904.

[8] M. Bienvenu, C. Bourgaux, F. Goasdoué, Querying Inconsistent Description Logic Knowledge Bases under Preferred Repair Semantics, in: AAAI, AAAI Press, 2014. URL: http://www.aaai.org/ocs/index.php/AAAI/AAAI14/paper/view/8231. doi:10.1609/aaai.v28i1.8855.

[9] V. Gallina, E. Lanbach, A. Ahmeti, S. Ritter, A. Revenko, A. Belova, A. Steinwender, D. Bachlechner, Creating a digital product passport using data spaces and ontologies: A case study at a furniture dealer, IFAC-PapersOnLine 59 (2025) 643–648.

[10] E. Blomqvist, H. Li, R. Keskisärkkä, M. Lindecrantz, M. A. N. Pour, Y. Li, P. Lambrix, Cross-domain modelling-a network of core ontologies for the circular economy., in: WOP@ ISWC, 2023, pp. 1–12.

[11] S. Steyskal, A. Polleres, Defining expressive access policies for linked data using the ODRL ontology 2.0, in: Proceedings of the 10th International Conference on Semantic Systems, ACM, New York, NY, USA, 2014.

[12] A. Ahmeti, D. Calvanese, A. Polleres, V. Savenkov, Handling inconsistencies due to class disjointness in SPARQL updates, in: LNCS: Proceedings of the 13th European Semantic Web Conference (ESWC2016), Springer, Germany, 2016, pp. 387–404.

[13] J. Pérez, M. Arenas, C. Gutierrez, Semantics and complexity of SPARQL, in: I. Cruz, S. Decker, D. Allemang, C. Preist, D. Schwabe, P. Mika, M. Uschold, L. M. Aroyo (Eds.), The Semantic Web - ISWC 2006, Springer Berlin Heidelberg, Berlin, Heidelberg, 2006, pp. 30–43.

7. Appendix

Table 1
Prefixes and their namespace IRIs used.

Prefix	Namespace IRI
xsd	`http://www.w3.org/2001/XMLSchema#`
rdf	`http://www.w3.org/1999/02/22-rdf-syntax-ns#`
champ-inst	`https://resource.champi40ns.eu/`
champ-onto	`https://schema.champi40ns.eu#`
odrl	`http://www.w3.org/ns/odrl/2/`
sh	`http://www.w3.org/ns/shacl#`
rsx	`http://rdf4j.org/shacl-extensions#`
vcard	`http://www.w3.org/2006/vcard/ns#`

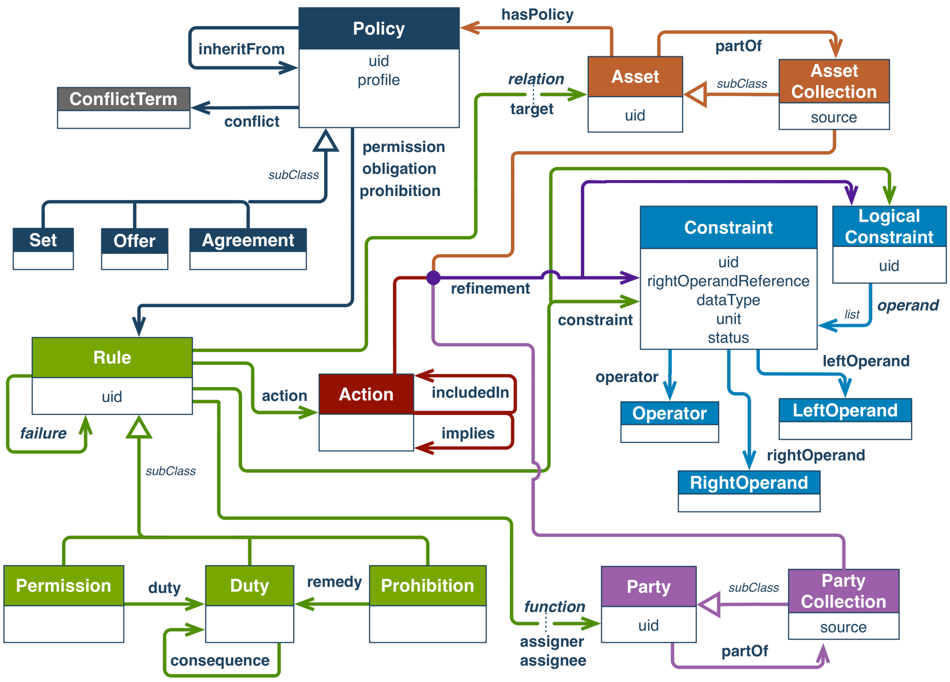

Figure 3: ODRL Information Model as per specification https://www.w3.org/TR/odrl-model/.

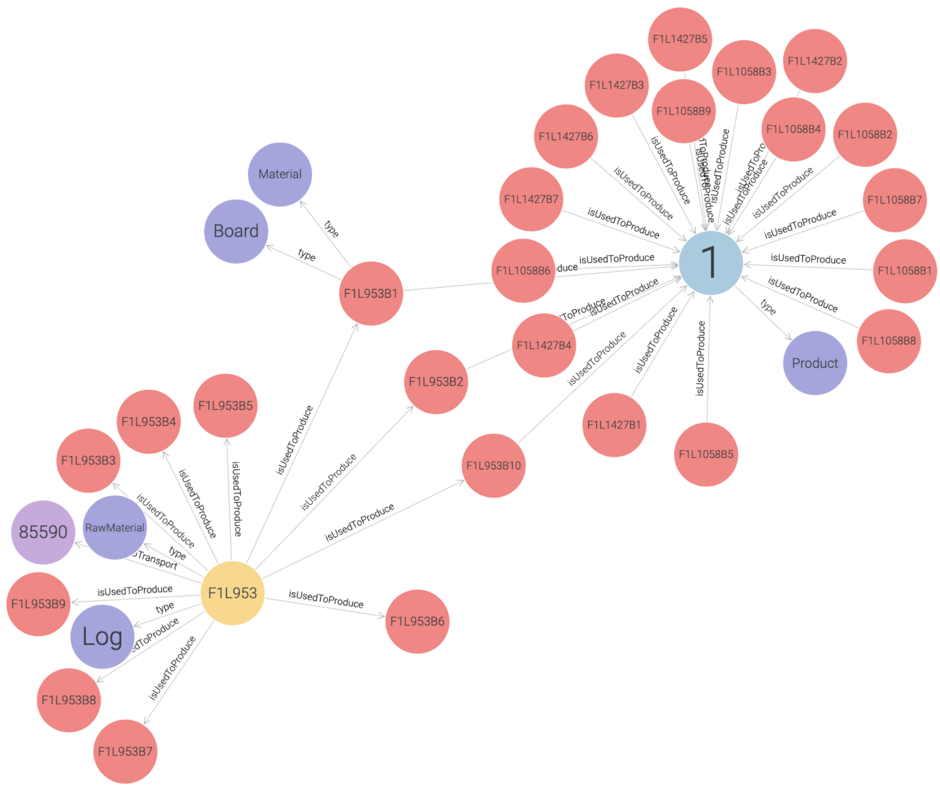

Figure 4: champI4.0ns Knowledge Graph enabling wood traceability of products down to boards and logs resp. via the properties `champ-onto:isUsedToProduce` (in the visualisation not all boards are expanded).

Example 1. *The application of Alg. 1 on the examples shown in Sec. 4 for user* alice *and target dataset* `<https://data.champi40ns.eu/joinery-product>`. *A conflict exists between permissions and prohibitions. As a result, values between 30 and 40 are filtered out and hence not returned to the user for the property* `wasteInPercentage`*, despite the presence of a permission that allows values less than or equal to 50.*

```
SELECT ?s ?p ?o
WHERE
 { GRAPH ?g { ?s  ?p  ?o }
   FILTER ( ?g = <https://data.champi40ns.eu/joinery-product> )

   { GRAPH <urn:acl>
        { champ-inst:ProductView_1
                    odrl:partOf  ?g ;
                    a  odrl:Asset .
          _:b1      odrl:action  odrl:read ;
                    odrl:assignee  ?user ;
                    odrl:target  champ-inst:ProductView_1
          FILTER ( ( ?p = champ-onto:wasteInPercentage ) && ( ?o <= 50 ) )
        }
   }
   FILTER NOT EXISTS {
       GRAPH <urn:acl>
          { _:b3 odrl:action  odrl:read ;
                 odrl:assignee  ?user ;
                 odrl:target  champ-inst:ProductView_2 .
            champ-inst:ProductView_2 odrl:partOf ?g ;
                     a odrl:Asset .
            FILTER ( ( ?p = champ-onto:wasteInPercentage ) && ( ( ?o <= 40 ) && ( ?o >= 30 ) ) )
          }
       }
   FILTER ( ?user = champ-inst:user_alice )
 }
```