Context Associations: an Application-Independent Annotation Method for RDF Knowledge Graphs

Ruben Dedecker*, Ben De Meester and Pieter Colpaert

IDLab, Ghent University – imec, Technologiepark-Zwijnaarde 126, 9052 Gent, Belgium

Abstract

Data integration typically relies on more than raw triples: provenance, quality indicators, usage policies, and signatures are often required as in-band contextual statements. In the Resource Description Framework (RDF), such annotations are expressed through a range of annotation models and method-specific practices. In this space, mature systems such as DQV, nanopublications, RO-Crates, and W3C Verifiable Credentials differ both in how they model annotations and how the annotation target is defined, with contextual information encoded in the annotation system rather than at the data level. This heterogeneity limits the uniform storage, exchange, discovery, and querying of contextual information associated with target statements. We present Context Associations, an approach for uniformly modeling and querying associations between contextual information and statements in an RDF knowledge graph. Our approach enables a lossless and reversible conversion of existing annotations into a single association model based on blank-node graphs. We evaluate Context Associations across the aforementioned annotation systems and show that contextual information can be uniformly associated with target statements and queried across applications. We further show that the original formats can be fully reconstructed when method-specific modeling assumptions are made explicit. By providing a uniform representation of contextual information associated with RDF statements, Context Associations supports the discovery, exchange, storage, and processing of heterogeneous annotations.

Keywords

RDF, Metadata, Annotations, Named Graphs, Interoperability, Context Associations

1. Introduction

Data integration and reuse typically require more than raw triples: provenance, policies, signatures, licensing, uncertainty, and quality indicators are often needed as *in-band* contextual statements that annotate target data. In RDF, both context and target may range from individual statements to graphs or datasets that are exchanged and queried together as a single knowledge graph. There are many possible *annotation models*–i.e., RDF-level models to associate contextual information with target data (e.g., reification, graphs)–and even more *annotation methods* – i.e., application-specific instances of an annotation model. These annotation methods rely on specifications and protocol definitions, as illustrated by mature annotation systems such as DQV, nanopublications [1], RO-Crates [2], and W3C Verifiable Credentials (VCs) [3].

For example, the Data Quality Vocabulary (DQV) [4] uses the term-bound annotation model, where the annotation method relies on DQV-specific relations and classes. Data quality is represented as term-bound associations of subjects of type `dqv:QualityAnnotation` or `dqv:QualityMeasurement` that directly associate contextual information to target datasets or distributions via the predicates `oa:hasTarget` (from the Web Annotation Ontology[1]) or `dqv:computedOn`, respectively. In the Nanopublication specification, contextual information is

QKG@ESWC 2026 — Evaluating, Improving, and Sustaining Knowledge Graph Quality, co-located with ESWC 2026, Dubrovnik, Croatia

*Corresponding author.

ruben.dedecker@ugent.be (R. Dedecker)

https://www.rubendedecker.be/ (R. Dedecker); https://work.de-meester.org/ (B. D. Meester); https://pietercolpaert.be/ (P. Colpaert)

0000-0002-3257-3394 (R. Dedecker); 0000-0003-0248-0987 (B. D. Meester); 0000-0001-6917-2167 (P. Colpaert)

© 2026 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

[1] https://www.w3.org/ns/oa

associated through explicit graph structuring: a nanopublication is composed of four named graphs – Head, Assertion, Provenance, and PublicationInfo – and the Head graph uses predicates such as `np:hasAssertion`, `np:hasProvenance`, and `np:hasPublicationInfo` to relate the nanopublication resource to its constituent graphs [5]. In RO-Crate, contextual information is associated more implicitly through the JSON-LD graph structure of the RO-Crate Metadata Document using existing vocabularies, such as Schema.org[2], where relationships between entities (e.g., dataset and an information resource publishing part of the dataset; or linking to the creator) are encoded as linked properties in the `@graph` [2]. In the VC Data Model specification, contextual information is associated through an explicit credential structure: a verifiable credential is composed of core properties such as `credentialSubject`, which links to the identified subject resource, and `proof`, which binds the credential to a cryptographic verification method [3].

As these examples illustrate, annotation methods are not aligned and are not always explicitly described at the data level. When, for example, asserting data quality, this mismatch in annotation methods across applications limits the uniform storage, exchange, and discovery of contextual information associated with a target set of statements. For example, when integrating both contextual information and target data in an RDF graph store, it becomes difficult to distinguish contextual information from the target data itself (especially when, for example, the quality measurements themselves also have associated contextual information).

In this paper, we present Context Associations: an approach, together with an associated specification and tooling, to uniformly model and query which contextual information is associated with which statements in an RDF knowledge graph. Context Associations is available at https://w3id.org/context-associations/specification. The approach relates to a demonstrator [6]. This work elaborates on the requirements and other options that exist.

With section 2, we give a brief overview on the state of the art on annotation models and methods in RDF. Based on this, we define a set of requirements in section 3 and introduce the Context Associations approach. We then show in section 4 how it can be applied in a practical case study, which is discussed in combination with the limitations of the approach and a summary table discussing how aforementioned requirements are met by different models in section 5. Finally, section 6 contains our conclusions and perspectives.

2. State of the Art

The Resource Description Framework lends itself well to the description of Web resources, using URI references as the abstraction layer for the target concept of the annotation. These target URIs can be Web resources that can be dereferenced from that target URI – following recommendations of vocabularies such as the Data Catalog Vocabulary (DCAT)– or they can be internal entities to the RDF knowledge graph. Given that such dereferencing of resources denoted by a URI for their inclusion in local storage or processing requires additional conventions that vary across systems and are not standardized in RDF itself, we align our evaluation with [7] in restricting the scope of this work to annotation models and methods *within the confines of an RDF dataset*, where target boundaries are represented in-band and Web dereferencing is no longer needed.

2.1. Annotation models

The work of Müller et al. [7] gives an overview of annotation models that are part of the RDF model, namely *reification*, *singleton properties*, and *named graphs*. They conclude that while reification offers fine-grained statement-level annotation, it incurs significant verbosity and complexity, whereas named graphs provide a more practical and widely supported mechanism for grouping statements and attaching contextual information at the graph level. Overall, they

[2] https://schema.org

argue that no single approach is universally optimal, and that the choice of representation depends on the required granularity of annotation and the intended processing environment.

Term-bound annotation is how we define the capability of RDF to annotate resources based on direct referencing of the resource URI as the subject or object term in triple statements. Here, the boundary between domain data and contextual data is not structurally represented and typically depends on conventions and vocabulary-specific interpretation. Additionally, the exact scope of the resource defined by the term URI is abstract within the knowledge graph, and similarly relies on conventions for precise definition.

Reification[3] allows deconstructing triples to a set of triples defining the subject, predicate and object of the reified triple, to associate contextual information. However, RDF 1.0 Semantics state that reification does not entail assertion of the described triple, and asserting a triple does not entail its reification, which complicates their interpretation without additional conventions.

Singleton properties are a proposed method in RDF to overload a triple predicate, similar to the working of Labeled Property Graphs [8], in which the predicate is replaced by an instanced predicate, derived from the original predicate, that can be referenced in other statements to associate contextual information to the original relation.

Named graphs were introduced for defining graphs in an RDF dataset, in the form of a (name, RDF graph) pair called a named graph. However, similar to reification, the semantic relation between the name identifier and RDF graph remains under-specified and is typically fixed by application-level conventions [9].

Triple terms, formerly known as quoted triples in RDF-star, are being standardized as part of ongoing RDF 1.2 work[4] and provide a compact mechanism for statement-level annotation without the verbosity of reification.

2.2. Annotation methods

Annotation methods instantiate these models by introducing method-specific conventions for target identification, target boundary, and processing behavior (see section 1).

Ontologies such as the Data Quality Vocabulary (DQV), The Open Digital Rights Language (ODRL) and the Data Catalog Vocabulary (DCAT) rely on *term-annotation* rely on term-bound annotation with vocabulary-defined semantics to define the target scope of the annotation. Other methods such as nanopublications rely on *graph structuring*, defining concrete interpretation for predicate targets as referencing the named graph with the target URI as graph name. Approaches such as Verifiable Credentials (VC)[10] and trusty URIs[11] combine dataset representations with *canonicalization and cryptographic hashing*, through RDF Dataset Canonicalization [12] and related Data Integrity work.

In practice, annotation methods instantiate the above models through application-specific methods (see examples in Section 1). Some annotation methods build on RDF knowledge graph canonicalization to create a stable identifier to associate contextual information to. This provides a **reificiation-like** pattern that is not limited to a single term. Through JSON-LD expansion and with accompanying W3C CCG Note "RDF Dataset Canonicalization and Hashing" and related Data Integrity specifications, Verifiable Credentials can be interpreted as RDF datasets – with a stable identifier – for the purpose of canonicalization and cryptographic proof generation[5]. Specifications such as trustyURI make use of an RDF knowledge graph canonicalization algorithm to generate a hash, and append this hash value as the resource extension to produce stable (and verifiable) identifiers[6]. Many specifications, such as Nanopublications, make use of these **named graphs** to organize their contents and associated context in these graphs.

[3] https://www.w3.org/TR/rdf-mt/#Reif
[4] https://www.w3.org/TR/rdf12-concepts/
[5] https://www.w3.org/TR/rdf-canon/
[6] https://trustyuri.net/spec/v1.FADQoZWcYugekAb4jW-Zm3_5Cd9tmkkYEV0bxK2fLSKao.md

Within Verifiable Credentials (VCs), the default graph is the target data, and named graphs are used to store the contextual information. This hybrid method of representing contextual information **across default and named graphs** allows to provide a consistent JSON structure while also being interoperable with RDF through JSON-LD. However, this reliance on the default graph breaks the association when multiple credentials are merged into one RDF knowledge graph.

3. Methodology

To address the interoperability issues outlined in Section 1, we derive a set of requirements for a dataset-local, uniformly queryable annotation representation. We then introduce Context Associations as a minimal baseline that satisfies these requirements and that can serve as a compatibility layer for heterogeneous annotation methods.

3.1. Requirements

We derive the following requirements from the need to uniformly retrieve contextual information associated with target statements after merging heterogeneous annotations from multiple applications into a single knowledge graph, represented by a queryable RDF graph store.

- REQ1: the solution is **interoperable** and **implementation-independent**: no extensions to RDF or SPARQL are needed, and any existing RDF graph store can be used.
- REQ2: target data can be **any arbitrary set of statements and graphs**: target data is not bound to a single specific subject or graph, and can contain any existing RDF 1.1 (and upcoming RDF 1.2) statements, i.e., any combination of triples, graphs, and triple terms.
- REQ3: annotations are **explicit**: both the target statements and the associated contextual information are available in the RDF graph store.
- REQ4: annotations are **immutable**: merging contextual information from multiple applications cannot introduce collisions or other side effects.
- REQ5: annotations can be **recursive**: contextual information can itself have associated contextual information.

These requirements define a baseline for annotation methods to support uniform storage, discovery, exchange and processing of heterogeneous contextual annotations of target data from different application in RDF knowledge graphs. They require that annotations can be processed with common RDF tooling (*REQ1*), while annotating any combination of RDF statements that remain unambiguously defined within the knowledge graph (*REQ2*). Their extraction should support method-agnostic processing, where the link between contextual information and its target is represented in-band and is mechanically discoverable and traversable using queries (*REQ3*). As data and annotations is merged from different sources into a knowledge graph for storage and processing, the methods must be robust against unintended co-reference and collisions that would alter either the annotation or its target (*REQ4*). And as contextual information is often layered, in cases such as provenance chains or signatures over policies, recursive expressions of context should not introduce special cases (*REQ5*), which follows from (*REQ2*).

3.2. Context Associations

In this chapter, we are constructing a solution to REQ1-5 which we will call Context Associations, documented in a specification at https://w3id.org/context-associations/specification.

In order to support a closed set of statements (**REQ2**), Context Associations use named graphs. While the semantics of named graphs have been left open in the RDF1.1 specification[7], they

[7]A working group note (2014) on the semantics of RDF Datasets: https://www.w3.org/TR/rdf11-datasets/

are a standard RDF 1.1 pattern (**REQ1**) that can be exploited. Furthermore, annotations are explicit: the identifier associated with the named graph can be used in the same RDF graph store (**REQ3**) by using an RDF term.

However, the use of named graphs in RDF for associating context to target data introduces two potential issues for which we have to work around[8]. The first is that some RDF graph store implementations may automatically merge certain named graphs into the default graph, assuming the named graphs are only used for partitioning. Here we pragmatically advice to be cautious when working with systems that work with this assumption.

Secondly, where dataset merging across boundaries is not standardized in RDF, the merging of named graphs with identical name identifier is common behavior for RDF graph stores. As Context Associations makes use of named graphs to define concrete boundaries over sets of statements, such graph merge operations are to be prevented. For this, we use *blank node identifiers*, also previously applied by Braun et al. [13], as the name identifier of named graphs (**REQ4**). This way, the scope of the context statements and its association to a target set of statements is local to the scope of the storage, exchange, or operation in which they are used. If the use of blank nodes is impractical, skolem identifiers can be used to ensure the unique generation of graph name identifiers at the time of their construction.

To define the exact associations between graphs, we make use of an anchor triple that provides a directed link between both graphs in the form of `_:sourceGraph ca:aboutGraph _:targetGraph`. This also allows for graph chaining, i.e., recursive annotations (**REQ5**).

Context Associations explicitly provide the directed association between contextual information and target data. Existing annotation methods can be *encoded* as Context Associations, though these will typically require customized processing to encode their implicit conventions. The explicit inclusion of such conventions, such as graph names and assertion status, in the Context Association method enables their uniform *decoding*. Encoding typically follows the following steps:

First, based on the annotation method, scope the target data and identify the (different types of) contextual information. For nanopublications, the `AssertionGraph` is the target data, referenced by the `Provenance`, `PublicationInfo`, and `Head` graphs of the nanopublication in their CA format. For VCs, we encode the `credentialsubject` as the target graph, referenced by a `credential` graph defining the credential entity and associated metadata. The `proof` graph links to both these graphs.

Second, based on how the contextual information is attached to the target data, convert into Context Associations. The contextual information is always put in a named graph with blank node label. For **reification**, embed the original triple as target data in a named graph with blank node label. For a **singleton property**, separate the triple containing the singleton property and the accompanying `rdf:singletonPropertyOf` triple, and embed them as target data in a named graph with a blank node label. For (default and named) **graphs**, rename the graph to a named graph with blank node label and add the original graph name through `ca:graphName`. For **term-bound linking**, put target data in a named graph with blank node label.

For each annotation method, customized encodings need to be provided. These, and a general decoding of Context Associations are validated through a SPARQL CONSTRUCT Queries, are made available at https://github.com/KNowledgeOnWebScale/context-associations-encoding-decoding/ under the permissive MIT license.

The original semantics of the contextual information can be reconstructed, but the syntactic constraints of the original annotation models may require additional processing. For example, VC signatures cannot be verified in the Context Association encoding because some signature suites rely on a specific JSON frame. They can only be verified after decoding and reformatting the data in the correct structure.

[8]A blog post with more elaborate explanations: https://pietercolpaert.be/linkeddata/2025/09/30/named-graphs

```
_:graph1 {
  robase:survey-responses-2019.csv schema:contentSize "26452" ;
    schema:encodingFormat "text/csv" ;
    schema:name "Survey responses" ;
    a schema:MediaObject .
}
_:graph2 {
  _:graph2 ca:aboutGraph _:graph1.
  robase:ro-crate-metadata.json <http://purl.org/dc/terms/conformsTo
    ><https://w3id.org/ro/crate/1.1> ;
    schema:about robase: ; a schema:CreativeWork ;
    pav:createdBy ex:Alice .
}
```

Listing 1: Relevant RO-Crate Context Associations

```
_:graph1 {
  ca:graphName "http://example.org/nanopub/np1#" ;
  ex:comment a ex:Comment ; ex:hasCommentText "My survey showed a
    SUS score of 3.4" ;
}
_:graph2 {
  this:Provenance ca:aboutGraph _:graph1 .
  this:Assertion prov:wasAttributedTo ex:Alice ;
    prov:generatedAtTime "2026-02-19T10:30:00Z"^^xsd:dateTime .
}
```

Listing 2: Relevant nanopublication Context Associations

```
_:data { ex:Alice ex:hasDiploma ex:MasterDegree . }
_:proof {
    _:proof ca:aboutGraph _:data ;
        dct:created "2026-02-19T10:32:00Z"^^xsd:dateTime ;
        sec:cryptosuite "eddsa-rdfc-2022"^^sec:cryptosuiteString ;
        sec:proofValue> "[...]"^^sec:multibase.
}
```

Listing 3: Relevant VC Context Associations

4. Case study

To demonstrate the need for Context Associations, we take a use-case of a research output of a researcher. The contextual information consists of an RO-Crate defining the research output, dataset, and author information (Listing 1), a nanopublication of the publication (Listing 2), and a verifiable credential that states the diploma of the researcher (Listing 3). All demonstration data is available at https://github.com/KNowledgeOnWebScale/context-associations-encoding-decoding/.

Below, we include the relevant encoded Context Associations, for the RO-Crate, nanopublication, and verifiable credential. Prefixes are omitted for brevity but can be found at the repository.

```
CONSTRUCT {
  GRAPH ?g { ?s ?p ?o }
} WHERE {
  GRAPH ?target {
    { ?x ?y ex:Alice . } UNION { ex:Alice ?y ?z . }
  }
  GRAPH <urn:x-arq:UnionGraph> {
    ?source ca:about* ?target .
  }
  GRAPH ?source { ?s ?p ?o }
}
```

Listing 4: Extraction of the named graphs annotation chains with a single construct query using Apache Jena ARQ.

This uniform way of associating contextual information with target data allows us to extract all annotations we have about `ex:Alice`. Extracting all annotations using standard SPARQL 1.1 requires the following sequence:

1. We extract all annotation chains, based on the anchor triples `_:sourceGraph ca:aboutGraph _:targetGraph`. As this requires the use of property paths in SPARQL, which are not supported over arbitrary sets of named graphs, we first extract these anchor triples to the default graph.
2. To regenerate the contextual information as Context Associations for a specified target, we must output the relevant named graphs. As this is not natively supported in SPARQL CONSTRUCT queries, we instead extract all named graph identifiers that chain towards a target graph through (1)) a known name identifier, or (2) by matching a target set of statements to a named graph using the `GRAPH` keyword.
3. Finally, for each of the extracted graph identifiers, we need to reconstruct the output graphs individually, since SPARQL construct can only produce one output graph per query.

However, we can make use of the ARQ query engine provided by Apache Jena[9]. This SPARQL processor provides two functions that extend beyond the SPARQL specification, which allow us to perform the extraction in a single iteration: (1) ARQ provides a union graph under the identifier `urn:x-arq:UnionGraph` – a materialized union of the named graphs present in the queried RDF dataset – which allows us to extract the anchor triples from all graphs in one iteration; and (2) ARQ supports the creation of named graphs in the `CONSTRUCT` clause of a SPARQL query, which enables us to directly re-create the named graphs in the annotation chain, without having to iterate separately over each graph. Listing 4 shows the ARQ query that allows us to uniformly gather all contextual information of `ex:Alice` .

5. Discussion

Based on the initial case study performed in section 4, we demonstrated that for a combined case of researcher output, the combination of nanopublications, RO-Crate metadata and university-issued Verifiable Credentials, the Context Associations is capable of providing a uniform discovery layer for all encoded data graphs that include information about `Alice`, and all associated contextual information.

[9] https://jena.apache.org/index.html

The proposed Context Association method does have limitations in serving as an application-independent annotation method for RDF knowledge graphs. Encoding of source methods into the Context Association structure requires prior knowledge about the semantic and syntactic constraints of the source method, such as identifier reuse, fixed JSON-LD frames and URI conventions, that need to be included in the encoding for successful decoding. The identifier relabeling prevents the creation of a uniform cross-method encoding algorithm. Where chains of context graphs are supported, Context Associations does not enforce a unique chaining strategy. When combining different annotation methods, care must be taken that the correct decisions are made when chaining graphs to ensure correct discovery and retrieval of what context should be associated with what target statements. Additionally, this approach does not remove the ambiguity of what constitutes "data" versus "metadata". Different methods and even usages of the same ontology may draw different boundaries, leading to different encoding strategies for the same annotation methods. Finally, the lack of support for named graphs in SPARQL construct severely hinders their practical use in standard SPARQL query engine implementations. Where this can by resolved by extending the query engine, as shown by the ARQ query, this still limits practical usability in standardized RDF environments.

With Table 1, we show how Context Associations compares to existing annotation models and methods, based on the five requirements introduced in subsection 3.1.

	term-bound	reification / triple terms	named graphs	singleton property	hybrid default & named graph	Context Associations
interoperable	**?**	✓	**?**	**?**	**?**	**?**
RDF support	method dependent	1 triple statement	1 RDF graph	1 triple statement	(multiple) triple statement(s)	(multiple) RDF graph(s)
explicit	method dependent	✓	✓	✓	✓	✓
immutable	**?**	**?**	**?**	**?**	default graph collisions	✓
recursive	✓	✓	✓	✓	×	✓

Table 1
Comparison of the requirements of existing annotation models and methods with Context Associations. At *interoperable*, **?** denotes that custom annotation methods are introduced to interpret the annotation model. At immutable, **?** denotes immutability requires globally unique and non-colliding identifiers.

6. Conclusion

Although Context Associations in its current form is a custom annotation method, being a superset of existing models and methods shows the potential to increase interoperability across existing annotation models and methods. By explicitly scoping contextual information from target data in existing annotation methods, and enabling the uniform extraction of contextual information across existing methods with regards to target statements identified using SPARQL queries as shown in the case study, Context Associations provides a more complete picture of what assertions that target the same data in RDF knowledge graphs. Most of the complexity in the approach is captured in the method-specific encoding for generating the Context Associations, requiring prior knowledge of the source methods to perform correct encoding and decoding to the Context Association method, making its applicability dependent on generating such encodings either at the data source, or before merging into local knowledge graphs.

A first application of Context Associations was explored in the context of ODRL [6], and future work involves further application of Context Associations to improve enabling of discovery, exchange, storage, and processing of heterogeneous annotations, assessing the feasibility of their use in larger data spaces.

References

[1] Nanopublication Community, Nanopublication guidelines (working draft), 2026. URL: https://nanopub.net/guidelines/working_draft/, docs index: https://nanopub.net/docs/ (last updated 2026-02-05).

[2] ResearchObject.org Community, Ro-crate metadata specification 1.2.0, 2025. URL: https://w3id.org/ro/crate/1.2. doi:https://doi.org/10.5281/zenodo.13751027, spec landing page: https://www.researchobject.org/ro-crate/specification/1.2/index.html.

[3] Verifiable Credentials Data Model v2.0, W3C Recommendation REC-vc-data-model-2.0-20250515, W3C, 2025. URL: https://www.w3.org/TR/2025/REC-vc-data-model-2.0-20250515/.

[4] Data on the Web Best Practices: Data Quality Vocabulary, W3C Working Group Note NOTE-vocab-dqv-20161215, W3C, 2016. URL: https://www.w3.org/TR/2016/NOTE-vocab-dqv-20161215/.

[5] P. Groth, A. Gibson, J. Velterop, The anatomy of a nanopublication, Information Services & Use 30 (2010) 51–56. doi:https://doi.org/10.3233/ISU-2010-0613.

[6] R. Dedecker, J. De Roo, B. Esteves, P. Colpaert, Demonstrating a pragmatic solution to context associations in RDF using blank node graphs, in: The Semantic Web: ESWC 2025 Satellite Events, Springer, 2025, pp. 58–61. doi:https://doi.org/10.1007/978-3-031-99554-5_11.

[7] J. Frey, K. Müller, S. Hellmann, E. Rahm, M.-E. Vidal, Evaluation of metadata representations in RDF stores, Semantic Web 10 (2019) 205–229. doi:https://doi.org/10.3233/SW-180307.

[8] V. Nguyen, O. Bodenreider, A. Sheth, Don't like RDF reification? making statements about statements using singleton property, in: Proceedings of the 23rd International Conference on World Wide Web, 2014, pp. 759–770. doi:https://doi.org/10.1145/2566486.2567973.

[9] J. J. Carroll, C. Bizer, P. Hayes, P. Stickler, Named graphs, Journal of Web Semantics 3 (2005) 247–267. URL: https://www.sciencedirect.com/science/article/pii/S1570826805000235. doi:https://doi.org/10.1016/j.websem.2005.09.001, world Wide Web Conference 2005 — Semantic Web Track.

[10] M. Sporny, D. Longley, D. Chadwick, J. Andrieu, Verifiable credentials data model v2.0, https://www.w3.org/TR/vc-data-model-2.0/, 2024. W3C Recommendation.

[11] T. Kuhn, M. Dumontier, Trusty uris: Verifiable, immutable, and permanent digital artifacts for linked data, Proceedings of the 13th International Semantic Web Conference (ISWC 2014) (2014) 395–410. doi:10.1007/978-3-319-11964-9_25.

[12] D. Longley, M. Sporny, Rdf dataset canonicalization, https://www.w3.org/TR/rdf-canon/, 2023. W3C Candidate Recommendation.

[13] C. H.-J. Braun, T. Käfer, RDF-based semantics for selective disclosure and zero-knowledge proofs on verifiable credentials, in: The Semantic Web: ESWC 2025, Springer, 2025, pp. 383–402. doi:https://doi.org/10.1007/978-3-031-94575-5_21.