

Towards usable ICDD containers for ontology-driven data linking and link validation

Philipp Hagedorn^{1,*}, Madhumitha Senthilvel², Hans Schevers³ and Lucas B. Verhelst⁴

¹Chair of Computing in Engineering, Ruhr-Universität Bochum, Germany

²Chair of Design Computation, RWTH Aachen University, Germany

³BuildingBits, The Netherlands

⁴BIM-Connected, The Netherlands

Abstract

Delivering data and documents of a building throughout its lifecycle is a common task in construction planning, engineering, and maintenance. The international standard Information Containers for linked Document Delivery (ICDD) has been created to link data and documents together for integrated delivery. This paper describes two independent software prototypes for creating these hybrid linksets in ICDD containers and validating them with the Shapes and Constraint Language (SHACL). This research focuses on ways to link data modeled in the Resource Description Framework (RDF) to non-RDF resources like documents and entities within documents. It argues that current ICDD mechanisms for linking to RDF resources are cumbersome and do not exploit the advantages of Linked Data. It also argues that the ICDD specification regarding the usage of Linked Data is very restrictive, thereby blocking the full benefits of Linked Data. The paper presents two strategies to alleviate these barriers to using ICDD; one approach is within the ICDD standard but adds additional agreements on top of the current ICDD standard. The other approach is technically outside the ICDD standard. Eventually, conclusions and recommendations are drawn from the presented implementations and strategies.

Keywords

ICDD, ISO 21597, Prototype, Deep Linking, Linked Building Data, SHACL, RDF Validation

1. Introduction

Stakeholders of the built environment and the construction industry constantly share heterogeneous information. In the complex landscape of the built environment, there is a need to exchange information in a standard way without losing the context of the information and only relying on vendor-specific formats [1]. Furthermore, with the increasing data volumes and different types of data having to form a unified view, the need to transfer interconnected data is growing [1].


The ISO 21597 [2, 3] *Information Container for linked Document Delivery – Exchange Specification* (ICDD) was released in 2020 to overcome the lack of a vendor-neutral structure for exchanging heterogeneous distributed building data. The ISO standard specifies a container


Proceedings LDAC2023 – 11th Linked Data in Architecture and Construction, June 15–16, 2023, Matera, Italy

*Corresponding author.

✉ philipp.hagedorn-n6v@ruhr-uni-bochum.de (P. Hagedorn); senthilvel@dc.rwth-aachen.de (M. Senthilvel); hans.schevers@buildingbits.nl (H. Schevers); lucas.verhelst@bim-connected.com (L. B. Verhelst)

ORCID 0000-0002-6249-243X (P. Hagedorn); 0000-0003-0733-9157 (M. Senthilvel); 0000-0003-1651-8202 (L. B. Verhelst)

 © 2023 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

 CEUR Workshop Proceedings (CEUR-WS.org)

with a structure that holds the payload that has to be exchanged. In this container, a header file is present to register and describe the contained files with metadata. Link files describe the (deep) links between these files. Both metadata files are formatted using Linked Data (LD) and Semantic Web formats, i.e., the Resource Description Framework (RDF) and the Web Ontology Language (OWL). RDF and OWL are standardized by the World Wide Web Consortium (W3C).

If the payload happens to be (partly) LD, the standard specifies that links should still be created through the linkset files. However, because deep linking between datasets is natively supported in LD, the proposed linking mechanism of ICDD is unnecessarily cumbersome. Furthermore, the overhead of the linking structure in ICDD could lead to redundancy in the data if direct links between the LD payload persist. Further, the standard does not specify how to transfer the data model and requirements that might be encoded using Semantic Web standards, e.g., Shapes Constraint Language (SHACL). Such functionality would make the container even more self-describing and enable the software to verify the exchanged data.

In this paper, we discuss how a revision of the ICDD standard might change the container to use the characteristics of LD better and consequently use the standard languages of LD (e.g., RDF, OWL, SHACL). However, because the conformance paragraph in the standard prohibits any extensions of the structures specified in it, only one of the two proposed solutions will conform to the standard. In the following, we will examine two independent software prototypes for ICDD regarding their usability with LD payloads, their handling of the linkage between RDF data and documents, and their ability to validate the contents of ICDD containers with SHACL. After concluding problem statements from the analysis of the prototypes, strategies for extensions of the ICDD standard are defined, and general recommendations are given in the conclusions.

2. The ICDD standard and Linked Data

Information exchange in the construction industry is conducted according to ISO 19650 [4]. Information containers are referred to for exchanging information as the smallest addressable unit in a software environment. However, it is not defined what an information container is and how it should be structured. A standardized information container is supplied with the ICDD defined by ISO 21597-1:2020 [2]. The standard contains a structured container schema in a ZIP archive built upon Semantic Web technologies. ICDD containers can exchange interconnected documents such as GML files, IFC files, Excel files, PDF files, CAD drawings, etc. Information about documents, including links between them, can be asserted using LD in the container. Use cases for the ICDD container are presented in recent research [5, 6, 7]. LD makes use of a part of the Semantic Web technologies. There are increasingly more examples in research where building data is converted to LD or building data is referred to via LD [8]. The construction industry increasingly uses these Semantic Web technologies to model knowledge in RDF and OWL to facilitate a web-based provision and exchange of building data [9].

LD is provided through publicly available base and domain-specific ontologies that can be leveraged to represent building data in a modular ontology combination. Base ontologies contain commonly agreed terminology that is valid for multiple domains, while domain ontologies extend the base ontologies with domain-specific knowledge. These ontologies are also referred

to as Terminology-Box (T-Box). The W3C Linked Building Data (LBD) Community Group¹ is one instance that proposes well-established base ontologies such as the Building Topology Ontology (BOT) or the Ontology for Property Management (OPM) [9]. LBD is thus an approach to employ linked data-based applications across the life cycle of buildings using shared ontologies. Besides these shared ontologies, Object Type Libraries (OTL) are increasingly used to describe the semantics of buildings, including standardized object types and property specifications of objects.

The ICDD container is a successor of the Dutch COINS container, as described by Hoeber et al. [10]. The container supplies an index file defined by the vocabulary from the *Container.rdf*² and includes a folder *Ontology resources* for ontologies, *Payload documents* for arbitrary documents, and *Payload triples* for RDF-serialized data and linksets.

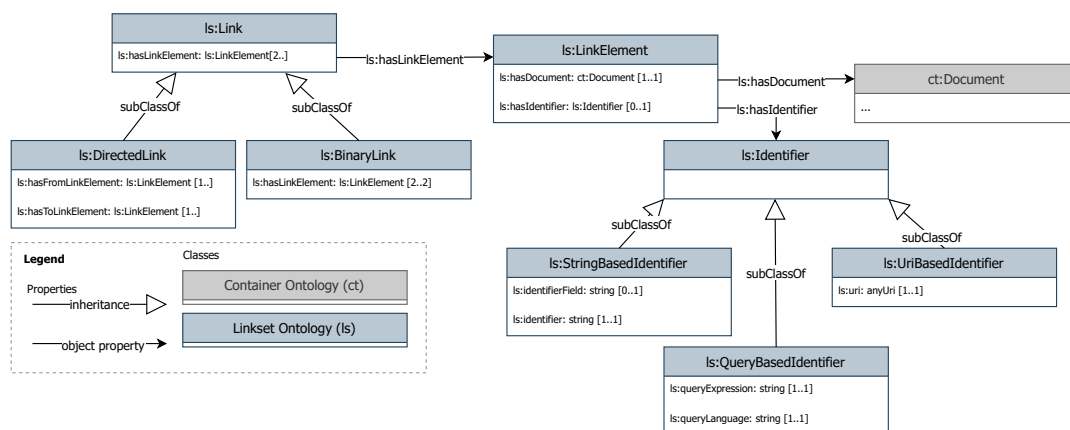


Figure 1: Overview about the main classes in the ICDD Container ontology and Linkset ontology

Documents registered in the container index can be linked using the vocabulary from the *Linkset.rdf*³. Specific link types can be used with the *ExtendedLinkset.rdf*⁴ vocabulary, which is standardized in the second part of the ISO standard [3]. The general structure of defining links between documents is reported in Figure 1.

Linking between two or more documents is provided through the link structure shown in Figure 1, defining a `Is:LinkElement` class that can be referenced in several link types. The `Is:LinkElement` class contains exactly one document reference and optionally one identifier of the identifier types `Is:StringBasedIdentifier`, `Is:UriBasedIdentifier`, and `Is:QueryBasedIdentifier`. These identifiers allow the deep linking of entities inside documents. Deep linking uses the above-mentioned identifiers, which form a generic mechanism that arguably needs supplementary specifications for each file type. For example, deep linking to a GML file could mean that ICDD’s identifier refers to a `gml:id` within a GML file, although this is not documented or specified. It could also mean that the ICDD identifier refers to the

¹W3C LBD Community Group, URL: <https://www.w3.org/community/lbd/>, accessed: 29.01.2023

²Container.rdf, URL: <https://standards.iso.org/iso/21597/-1/ed-1/en/Container.rdf>, accessed: 27.10.2022

³Linkset.rdf, URL: <https://standards.iso.org/iso/21597/-1/ed-1/en/Linkset.rdf>, accessed: 27.10.2022

⁴ExtendedLinkset.rdf, URL: <https://standards.iso.org/iso/21597/-2/ed-1/en/ExtendedLinkset.rdf>, accessed: 27.10.2022

objectID property within a GML file. For linking to IFC files, ICDD's identifier could use the IFC-GUID property, but the line number (in the case of an IFC step physical file) is also valid. The deep linkage of drawings and IFC models in ICDD containers has also been analyzed by Borrmann et al. [5].

Validation of ICDD containers is provided by the second part of the ISO 21597-2:2020 [3]. Therein, SHACL [11] is used to validate every instance of the ICDD ontologies inside a container. To conduct the validation, the appendix delivers SHACL shapes to verify the container. SHACL validation for LBD using generated SHACL shapes has been investigated by Senthilvel et al. [12]. The documentation of SHACL can be found in [11]. The usage of SHACL for ICDD container conformity validation is also addressed by Schevers [13]. However, the approaches in [3] and [13] only validate the conformity with the ISO 21597 standard and do not touch the contents of a container. The conformity criteria of an ICDD container are specified in ISO 21597 and forbid the extension of any class or property from the given ICDD ontologies. According to the conformance criteria in 6.2 f) of ISO 21597-2 [3], RDF data must be placed in the *Payload triples* folder and is thus not registered in the index file of the container. A consequence of this is that no links to entities can be made in the RDF data, as a `ct:Document` instance is mandatory for this (see Figure 1).

An approach for validating the linksets inside ICDD containers content-wise to ensure the link validity based on the linked entities is provided by Hagedorn and König [14]. In their approach, building data is transferred into RDF data and validated in the synopsis of all resources.

3. Prototype software systems

In the following section, two independent software prototypes for viewing and editing ICDD containers are presented and analyzed regarding their handling of RDF-based data and linking between these RDF-based data and documents inside the container.

3.1. RUB ICDD Platform (RUB-IP)

The RUB-IP is a viewing, editing, and collaboration platform based on ICDD information containers. Hagedorn et al. [15] provide the underlying software architecture for this web platform. ICDD containers are maintained on the platform in projects. According to information management using ISO 19650, projects can contain ICDD containers with specific states and purposes. The platform implements rights management for projects and containers. Using RUB-IP, ICDD containers can be queried using SPARQL and validated using SHACL. Therefore, payload documents available as RDF data, container metadata, linksets, and all other RDF data from the *Payload triples* folder are integrated into a triples store. Thus, validation can be performed not only on the schema level but also on the content level and regarding the integrity and plausibility of linksets. Moreover, RUB-IP renders IFC files into a 3D viewer, to enable a visual selection of IFC elements and to query, e.g., linked documents or entities, for the selection by using SPARQL queries. The platform relies on the open-source published *IIB.ICDD* framework [16] for handling ICDD containers using *C#* and *.NET*.

RUB-IP shows the contents of an ICDD container featuring all the links between documents following the ICDD specifications. Extra functionality is present when ICDD links refer to

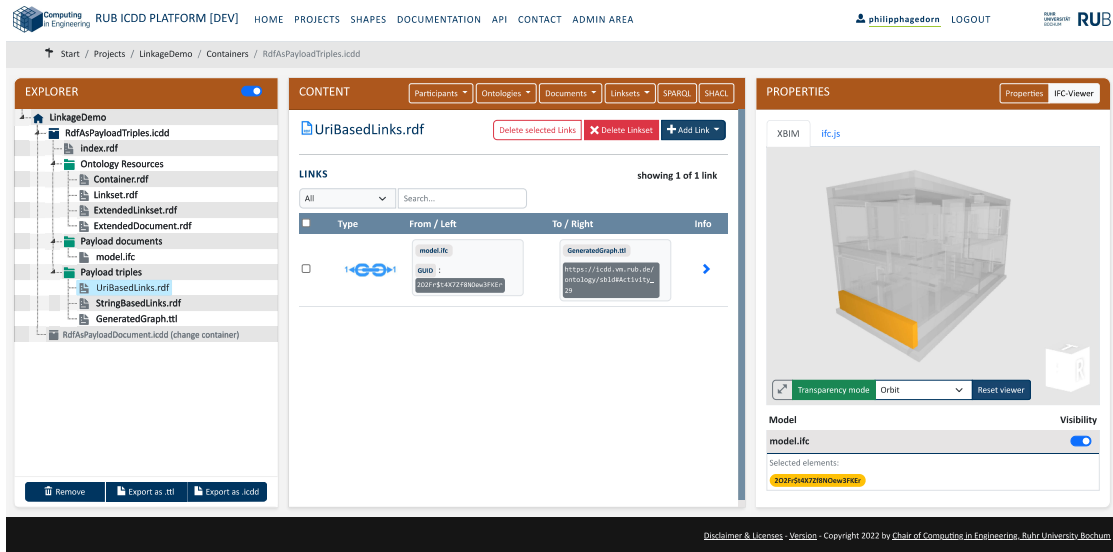


Figure 2: RUB ICDD Platform (RUB-IP) showing a link between an IFC file and an RDF entity

other RDF files, e.g., showing the graph structure of the link data structure, the linked RDF individuals, and direct L1 level properties of the entities. Rasmussen et al. [17] define these property levels for LBD from L1 to L3 according to the respective steps (number of triples) from a feature of interest to the value of a property.

3.1.1. Handling of RDF-based data

Loading RDF files into ICDD containers on the RUB-IP can be done threefold. First, RDF documents can be stored in the *Payload documents* folder as `ct:InternalDocument` instance. Moreover, RDF files can also be referenced as `ct:ExternalDocument` instances via their respective URI, under which an RDF serialized format is retrievable. Both options store the LD in the *Payload documents* folder. The third option to add LD to ICDD containers in the RUB-IP is to store it under *Payload triples* as defined in Part 2 of ISO 21597. As of the standards definition, only the linksets from the *Payload triples* are registered in the container index file, but neither the other RDF files in the *Payload triples* folder nor the ontologies in the *Ontology resources* folder are considered. For registering RDF files from the *Payload triples* folder in the container, the ICDD ontologies are extended, as described in [15]. A class `exdoc:PayloadProxy` is defined as a subclass of `ct:ExternalDocument` with the derived property `ls:uri` from the `ct:ExternalDocument` class. With this, an indication is made that an RDF document is added in the *Payload triples* folder referring to the respective `ls:uri` as a base URI of the contained RDF graph. All uploaded RDF files on the RUB-IP are automatically generated as `exdoc:PayloadProxy` instances and registered in the index file. Through RDFS reasoning, all instances of `exdoc:PayloadProxy` can be inferred to `ct:ExternalDocument` to conform to ISO 21597 again.

3.1.2. Handling of the linking between RDF data and documents

Depending on how RDF data is stored in containers on the RUB-IP, there are possible ways to link RDF data with other documents in the container using `ls:StringBasedIdentifier` or `ls:UriBasedIdentifier`. The identifier type `ls:QueryBasedIdentifier` is not yet implemented into the system. For RDF data stored in the *Payload documents*, links can easily be created using a `ls:LinkElement` referring to the internal or external document instance and either a URI-based identifier to address the RDF entity directly via its URI or a string-based identifier to refer to an RDF entity via a combination of an identifier field and identifier value. The latter option can be used to look up respective triples where the identifier field is used as a predicate and the identifier value as an object. This delivers one possible subject if the identifier value is unique in the RDF data (e.g., the IFC-GUID when using ifcOWL) or multiple values if the identifier is not unique (e.g., a property value of entities with a particular label or numeric value). If the RDF data is stored in *Payload triples*, both methods to identify instances can be used as described before. The only difference is that the `ls:LinkElement` refers to the generated `PayloadProxy` instance reflecting the RDF document in the container index file. However, none of these two solutions for handling RDF data and linking to it is intended by the ISO 21597 standard.

3.2. Wistor ICDD Viewer

The Wistor ICDD Viewer [18] enables users to upload an ICDD container and browse through the data. An IFC ‘Adaptor’ enables visualization of IFC files within the ICDD container. The adaptor resolves deep links towards IFC elements and other documents. Related content can be found using the 3D IFC viewer; consequently, extra documents can be attached to IFC elements. The goal of this ICDD viewer, besides handling ICDD content, is also to handle extra LD based on a Object Type Library (OTL) specified in SHACL. ‘Instances’ of OTL classes are present in an additional RDF file within the ICDD Container, enabling the exchange of objects, properties, and their relations connected to other documents using ICDD data structures. Figure 3 shows the Wistor ICDD Viewer of a substation design document in multiple IFC files where objects in red have extra information available using an RDF file in the *Payload documents* folder.

The properties panel (bottom left widget) shows the extra properties of the selected object. These properties are gathered from an RDF file. The connection between the selected IFC element and the RDF file resources is specified using the ICDD classes.

Multiple verification steps are available to ensure the exchanged data complies with the specifications. First, the ICDD container is verified to meet the ISO 21597 conditions. As the client primarily wants IFC files using their specifications related to (necessary) documents and wants extra object information according to their SHACL OTL, additional verification steps are required. IFC files are checked to ensure they comply, and the extra LD is validated against the SHACL OTL.

3.2.1. Handling of RDF-based data

The Wistor ICDD Viewer had to make its own extra decisions to enable a more holistic verification of ICDD data encompassing IFC files, ICDD links, and additional LD based upon an

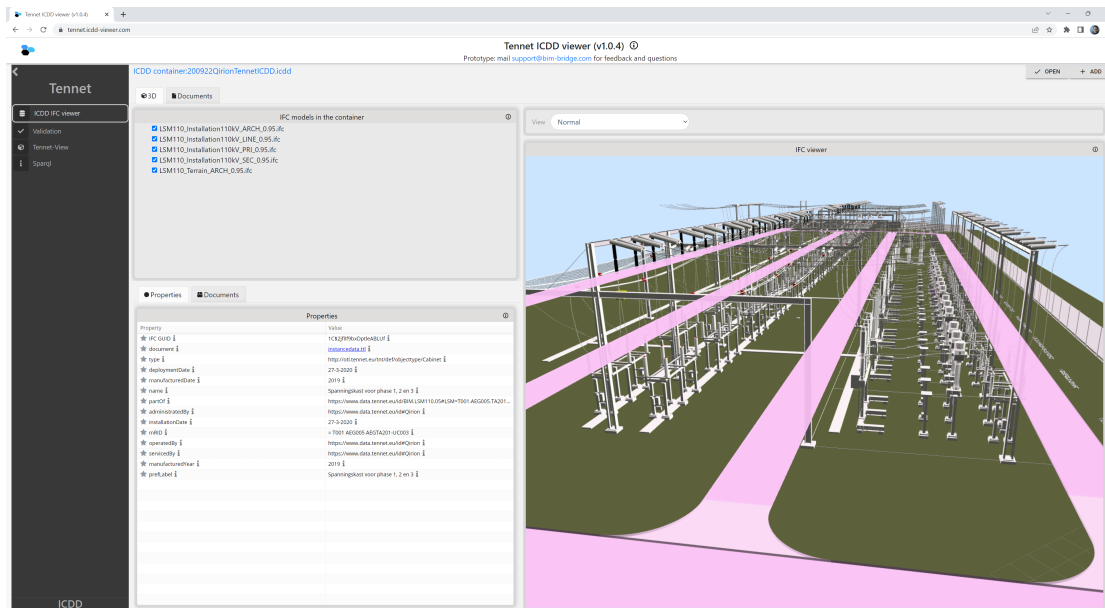


Figure 3: Wistor ICDD Viewer

OTL (in SHACL). The first decision was to load extra LD in the *Payload documents* folder within the ICDD specification. Primarily the idea was to put additional RDF files in the *Payload triples* folder, but Wistor’s interpretation was that this was not according to the ISO standard. A second decision was to load an LD OTL to make more information on object types available to the viewer. As the OTL was based upon SHACL, the extra LD set could be validated, ensuring that the exchanged data conforms. This was the third decision (validating the supplementary dataset against a SHACL/OTL).

3.2.2. Handling of the linking between RDF data and documents

Another decision was to use `ls:LinkElement` and `ls:StringBasedIdentifier` to link to an LD data resource. Instead, the foundation of LD proposes to use IRIs to link to resources. However, this approach was at least complying with the ICDD standard. As multiple ways are present to handle a link to an LD resource (e.g., `ls:UriBasedIdentifier`), extra guidelines are necessary for ICDD container creators to ensure they use the same approach. Data structures for linking to external documents were also present within the OTL, enabling cardinality restrictions or simply enabling the specification of mandatory documents. However, these data structures are present in the supplementary RDF files and not related to ICDD links, consequently introducing redundancy and potential conflicts.

3.3. Problem statements

Based on the two prototype implementations, remarks and problem statements on the usability of ICDD for ontology-driven data linking and link validation for the construction industry

are drawn. Both prototypes evaluated use cases where RDF data was available inside the container in addition to the container metadata and linksets, allowing users to load RDF files into the container. Both prototypes demonstrated a cumbersome way to link to an RDF resource using the ICDD linking structure with either the `ls:StringBasedIdentifier` or the `ls:UriBasedIdentifier` according to the standard and consequently ignoring very basic LD solutions. In the second prototype, this leads to a definition of links in the LD itself and not in the linksets of the ICDD, so the advantage of separating links in the linksets from the original data in the payload is not maintained anymore. This can lead to duplicate linking and additional effort in maintaining the container and linksets. Using the ICDD linking structure combined with LD in a linkset or payload LD is not allowed by the standard and would fail the SHACL validation provided by ISO 21597-2 [3].

Moreover, both prototypes use SHACL for validating and rule execution to validate and visualize the interconnections to additional LD in the ICDD containers. However, link validation is not genuinely possible for deep linking, as especially the `ls:StringBasedIdentifier` as a key-value-pair representing the identifier does not convey a unique meaning or identification. Thus, it is only usable within the defining system unless there is an agreement on how the identifiers are used. As the standard provides schema validation, the next step is validating the linking and context, which the standard does not cover yet. The standard does not deliver a solution where to store the SHACL shapes that should be evaluated for the specific container.

4. Extending ICDD for linking and validation

After comparing the implementations and considering the lack of solutions for deep linking RDF data in ICDD, the challenges are: (1) to overcome the quite complex standard for simple data linking, (2) to uniquely identify deep linked elements with the built-in identifiers, (3) to exploit the added value of LD and (4) to enable identification of SHACL files that are meant for data validation. To overcome these challenges, it is necessary to acknowledge that RDF data differs from other payloads in the container. The authors identified potential solutions in two strategies to put the strengths of the ICDD data structure together with the full power of LD.

The first strategy (stay-within-the-standard) contains solutions within the ICDD standard resulting in best practices or even additional agreements on dealing with LD datasets within ICDD containers. This could result in LD modeling agreements and an LD adapter implementation agreement. The second strategy (extend-the-standard) contains solutions that are technically not compliant with the ICDD standard because they extend or change the schema provided by the standard. These two strategies will be discussed further in the following paragraphs.

4.1. Compliant solutions (stay-within-the-standard)

Compliant solutions can be met by storing RDF data and SHACL files in the *Payload documents* folder without having them treated specially. Linking to RDF data must then be done via the `ls:URIBasedIdentifier` provided by the standard to make the linking resolvable for other container recipients. This linking mechanism is present in the current standard and is the closest way to bridge the gap to a handy and usable application of LD in ICDD. Regarding the unique identification of entities in other file types, e.g., IFC models, GIS data, or calculation spreadsheets,

a common understanding of how to use the `ls:StringBasedIdentifier` using adapters and implementation guidelines is necessary. The two prototypical implementations have shown that even for IFC documents, different key-value pairs for referring to the IFC GUID can be used for linking, e.g., when utilizing different keys: `"GUID"`, `"globalID"`, `"IFC GUID"`. This makes it hard for software systems to implement ICDD in an interoperable way.

4.2. Ideal solutions (non-compliant, extend-the-standard)

In the above section, solutions compliant with the current specifications in ISO 21597 were elaborated. This section presents approaches that deviate from the specifications of ISO 21597 while retaining the concept of information containers.

The first solution could be to allow extensions of the linkset ontology, including the necessary modifications to accommodate this. For instance, the ranges and restrictions on the predicates `ls:hasLinkElement`, `ls:hasFromLinkElement`, and `ls:hasToLinkElement` could be adjusted to not only allow instances of `ls:LinkElement`. One possibility would be a generic restriction to any `rdfs:Resource` so that either `ls:LinkElement` referencing the documents or RDF entities could directly be addressed, which is depicted in Figure 4.

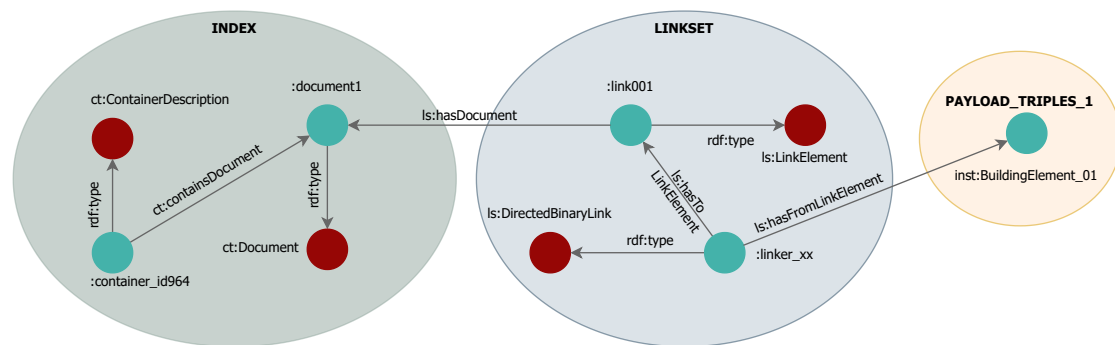


Figure 4: Referring to an RDF entity in the payload triples directly from a `ls:DirectedBinaryLink`

The advantage of this approach is that the `ls:LinkElement`, `ls:Identifier`, and the identifier declaration can entirely be omitted for one side of the link, and only the reference to the entity is referred to via `ls:hasFromLinkElement`. With this change in the modeling principle of the links, RDF data must not necessarily be registered in the index but must be present in the *Payload triples* folder.

T-Box files, like SHACL shapes and extensions of the linkset ontology, must be placed in the *Ontology resources* folder. RDF datasets (A-Box) must be put into the *Payload triples* folder. Company-specific OTLs or T-Box datasets like the upcoming Semantic Modeling and Linking (SML) standard draft could arguably be used with the ICDD standard. This could bridge the gap between hybrid modeling of semantics and documents at this point [19].

Eventually, a more generic solution is an approach that uses a combination of the above proposals and also considers the usage of the link type classes (defined in part 2) as property values. In this solution, the original container structure (of the three folders for *Ontology resources*, *Payload documents*, and *Payload triples* along with the overarching index file) is

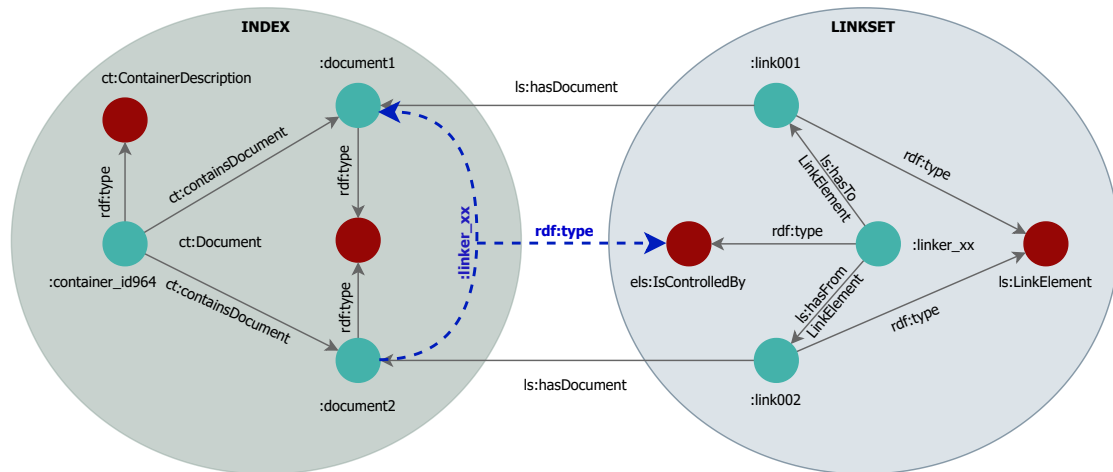


Figure 5: Direct linking mechanism between documents using the extended linkset vocabulary

retained as the essential information container structure. To illustrate this, Figure 5 showcases an example container index file snippet (green oval), with the linkset file (purple oval), in which two documents `:document1` and `:document2` are linked using the two hierarchy levels: first declaring documents as `ls:LinkElement` and this is then used to link (by using `ls:hasToLinkElement` and `ls:hasFromLinkElement`) to an instance `:linker_xx` which is finally assigned to a link class `els:IsControlledBy`. If this mandatory declaration of both `ls:Link` and `ls:LinkElement` can be omitted, and the current restrictions on the usage of link types are removed, it is possible to assign a link class as a property value directly (this is illustrated by the blue dotted arrows in the figure) [20]. However, in this solution, though the linkset structure is partially retained, the usage of link classes beyond the assigned domain and range will be recognized as a non-conformance.

5. Conclusions and Recommendations

Adopting LD for the built environment results in the need to exchange data using LD in conjunction with other documents. The development and adaption of the ICDD standard for the construction industry is a logical step toward data-driven storage and exchange of information. Two independent prototype developments were presented within this research focusing on the linking of data, the storage of LD datasets, and the validation of the container contents. However, the authors have recorded that the ICDD linking structure for RDF resources is cumbersome and does not exploit the full potential of LD. In the current standard state, additional agreements are necessary to facilitate an ICDD software ecosystem that maintains and verifies ICDD containers containing extra LD datasets. Possible solutions for improving the ICDD standards were given. The recommendations are summarized, and an outlook on future research is shown in the following. Based on the research carried out in this paper, several recommendations for action are outlined:

- When using ICDD in its current standardization, supplemental agreements for structuring link elements and identifiers for specific file types, e.g., IFC, GIS, or spreadsheets, could help interpret the links inside ICDD in different systems uniquely.
- The basic idea of ICDD should be that the potential of LD is not limited by its use as a simple ZIP archive without touching the benefits of LD.
- For extending the ICDD standard for an ideal interaction with LD, the extensions of classes and properties of the ICDD ontologies should be allowed.
- The storage of internal and external RDF files should be registered in the index file.
- SHACL shapes residing in a dedicated folder or referenced from a web resource should be registered inside the index file as requirements for container exchange.

Regardless of the presented solutions for improving the linking of LD datasets in ICDD, the validation of containers is needed and should be supported by the standard for the conformity check and the validation of content. Regarding further research, there is also ongoing standardization regarding the use of LD for modeling and linking building data as SML resulting in a standard draft of BS EN 17632 [19]. In addition, there are overlaps and touching points between modeling documents and non-document data. Further overlap is between general LD concepts such as the *Linked Data Platform*, which allows defining containers for maintaining heterogeneous data [21]. Both approaches can be analyzed in further work and could be adapted for implementing a usable container-based information exchange in the construction industry.

References

- [1] J. Werbrouck, P. Pauwels, J. Beetz, E. Mannens, Mapping Federated AEC projects to Industry Standards using dynamic Views, in: Proceedings of the 10th Linked Data in Architecture and Construction Workshop, 2022, pp. 65–76.
- [2] ISO 21597-1, Information container for linked document delivery: Exchange specification - Part 1: Container, 1 ed., International Organization for Standardization, Geneva, Switzerland, 2020. URL: <https://www.iso.org/standard/74389.html>.
- [3] ISO 21597-2, Information container for linked document delivery: Exchange specification - Part 2: Link types, 1 ed., International Organization for Standardization, Geneva, Switzerland, 2020. URL: <https://www.iso.org/standard/74390.html>.
- [4] ISO 19650-1, Organization of information about construction works - Information management using building information modelling. Part 1: Concepts and principles, 1 ed., International Organization for Standardization, Geneva, Switzerland, 2018.
- [5] A. Borrmann, J. Abualdenien, T. Krijnen, Information containers providing deep linkage of drawings and BIM models, The 38th CIB W78 conference on Information and Communication Technologies for AECO (2021). URL: <http://itc.scix.net/paper/w78-2021-paper-082>.
- [6] A. Göbels, J. Beetz, Using ICDD Containers for documentation data archives: Semi-automated creation of linked documentation data archives using an Information Container for linked Document Delivery, Proceedings of the 27th International Conference on Cultural Heritage and New Technologies, November 2022 (2022).

- [7] J. Karlapudi, V. Prathap, K. Menzel, An explanatory use case for the implementation of Information Container for linked Document Delivery in Common Data Environments, in: EG-ICE 2021 Workshop on Intelligent Computing in Engineering, 2021, pp. 76–86.
- [8] H. Schevers, J. Mitchell, P. Akhurst, D. Marchant, S. Bull, K. McDonald, R. Drogemuller, C. Linning, Towards digital facility modelling for Sydney opera house using IFC and semantic web technology, *ITcon 12 (2007)* 347–362. URL: <https://www.itcon.org/2007/24>.
- [9] P. Pauwels, K. McGlenn (Eds.), *Buildings and Semantics: Data Models and Web Technologies for the Built Environment*, 1 ed., CRC Press, London, 2022. doi:10.1201/9781003204381.
- [10] H. Hoeber, D. Alsem, Life-cycle information management using open-standard BIM, *Engineering, Construction and Architectural Management* 23 (2016) 696–708. doi:10.1108/ECAM-01-2016-0023.
- [11] H. Knublauch, D. Kontokostas, *Shapes Constraint Language (SHACL): W3C Recommendation 20 July 2017*, 2017. URL: <https://www.w3.org/TR/shacl/>.
- [12] M. Senthilvel, J. Beetz, A Visual Programming Approach for Validating Linked Building Data, in: L.-C. Ungureanu, T. Hartmann (Eds.), *EG-ICE 2020 proceedings: International Workshop on Intelligent Computing in Engineering*, Berlin, 2020, pp. 403–411.
- [13] H. Schevers, Towards a webbased service for NEN-EN-ISO 21597-1:2020 container validation: Bimloket Governance documentation Draft 19 september 2022, 2022. URL: <https://bimloket.github.io/ICDD-NL/validation/>.
- [14] P. Hagedorn, M. König, Rule-Based Semantic Validation for Standardized Linked Building Models, in: *Proceedings of the 18th International Conference on Computing in Civil and Building Engineering*, 2021, pp. 772–787. doi:10.1007/978-3-030-51295-8_53.
- [15] P. Hagedorn, L. Liu, M. König, R. Hajdin, T. Blumenfeld, M. Stöckner, M. Billmaier, K. Grossauer, K. Gavin, BIM-enabled Infrastructure Asset Management using Information Containers and Semantic Web, *ASCE Journal of Computing in Civil Engineering* 37 (2023). doi:10.1061/(ASCE)CP.1943-5487.0001051.
- [16] P. Hagedorn, IIB.ICDD framework: a framework to open, create, validate, edit, and export containers defined by ISO 21597-1:2020: MIT license, 2022. URL: <https://doi.org/10.5281/zenodo.7256174>. doi:10.5281/zenodo.7256174.
- [17] M. H. Rasmussen, M. Lefrancois, M. Bonduel, C. A. Hviid, J. Karlshøj, OPM: An ontology for describing properties that evolve over time, *Proceedings of the 6th Linked Data in Architecture and Construction Workshop (2018)*.
- [18] BIM-Connected and BuildingBits, Wistor platform, 2023. URL: <https://www.wistor.nl/en/>, last accessed: 29.03.2023.
- [19] BS EN 17632, *Building Information Modelling (BIM) - Semantic Modelling and Linking (SML): Part 1: Generic modelling patterns [Draft]*, 1 ed., British Standards Institution, London, United Kingdom, 2022.
- [20] N. Noy, M. Uschold, C. Welty, Representing Classes As Property Values on the Semantic Web, 2005. URL: <https://www.w3.org/TR/2005/NOTE-swbp-classes-as-values-20050405/>.
- [21] S. Speicher, J. Arwe, A. Malhotra, *Linked Data Platform 1.0*, W3C Recommendations (2015). URL: <http://www.w3.org/TR/2015/REC-ldp-20150226/>.