# Common Data Environments for the Information Container for linked Document Delivery

Madhumitha Senthilvel[1], Jyrki Oraskari[1], and Jakob Beetz[1]

Design Computation, Faculty of Architecture, RWTH Aachen University, 52062 Aachen, Germany

**Abstract.** Unstructured and poorly managed information is a major cause of time delays in construction projects. Availability of relevant information at the required time has a considerable impact on decision making in the project's lifecycle. Multi-models containers, and a more recent approach, Information container for linked document delivery (ICDD), aim to facilitate construction data management and sharing information. Containers help in structuring and linking of heterogeneous data. Such container models are relevant in and align with Common Data Environments (CDE) which facilitate a centralised environment for managing both information and services. Presently, three approaches fit in the vision of CDE: the DIN SPEC 91391-2, OpenCDE-API which is loosely based on DIN SPEC 91391-2, and the W3C Linked Data Platform (LDP), a generic data container-based approach to managing linked data graphs online. In this paper, we investigate how ICDD, an approach to link information, can be represented using the above three frameworks. A comparison is developed between the three approaches based on a sample use-case. The required conversion steps are analysed, and the limitations of the mapping are evaluated.

**Keywords:** CDE · OpenCDE · LDP · ICDD · DIN SPEC 91391-2

## 1 Introduction

In building projects, decisions are made based on the best information available at any given time. If any piece of data is missing or not accessible at the required time, it affects the decision, and subsequently, the quality of the project. Digitization has connected data across a federated Architecture Engineering and Construction (AEC) environment, making it increasingly easier to access the required information at the required time. However, the quality of the information available is also a deciding factor in determining its usefulness. Often, unstructured and poorly managed information, or missing links between documents/data is estimated to be the primary cause of time delays in construction [13].

Multi-models allow packaging and sharing heterogeneous building models and relevant application models in a single container. The first version of Multi-model-Container (MMC) [20] was originally defined by Scherer et al.[30, 29] in

Copyright © LDAC2020 for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).



CEUR-WS.org/Vol-2636/10paper.pdf

the Mefisto project for multi-model exchange [28]. They play an essential role in building projects where manifold file-formats and the variety of application domains are common.

Information Container for Document Delivery[1] (ICDD) [22, 23] is an upcoming standard for multi-model container approach that also allows using Linked Data (LD) [7, 8] and also Linked Building Data[2] (LBD) to interlink the models and to enable connecting the data to external sources. Linked Data provides commonly used Semantic Web methods and technologies to facilitate access to information. Representing the data in the widely used Resource Description Framework (RDF) [3] along ontology descriptions is expected to facilitate data management.

However, ICDD is a file-based container system, which means that it has the same limitations of file sharing. Without extra measures, the availability of the data for every partner and the possibility to access the most up-to-date data may not be insured. Common Data Environment (CDE) [11] addresses these issues, providing a single platform digital access to the containers and their relevant documents. At present, there are three approaches that fit in the vision of CDE: DIN SPEC 91391-2, OpenCDE-API which is loosely based on DIN SPEC 91391-2, and the W3C Linked Data Platform (LDP)[25], a generic data container-based approach to managing linked data graphs online.

The above CDE frameworks can express semantic links, much like ICDD, which proposes a linked data approach to linking documents. The question that then arises is "How CDEs which facilitate the broad architecture of data storage, access and retrieval, benefit from ICDD containers, which specifically focus on structuring and linking of heterogeneous data on meta and sub-document level".

In this paper, we investigate how ICDD can be represented using the above CDE frameworks. A comparison is developed between the three approaches based on a sample use-case. The required conversion steps are analysed, and the limitations of the mapping are evaluated. To establish the basic concepts of ICDD, OpenCDE, and LDP, a brief overview is given in Section 2. Section 3 introduces how ICDD data content can be served in the selected frameworks. Section 4 presents the results of comparing the solutions, with the conclusions focusing on summarizing the findings and recommendations of this research.

[1] formerly called "Information Container Data Drop" which has been adapted in the ISO standardization process to reflect more continuous processes as opposed to mere file-based "drops"

[2] Linked Building Data (LBD) is defined by W3C Linked Building Data Community Group https://www.w3.org/community/lbd/

## 2 Concepts and Related work

### 2.1 Information container for linked document delivery

ICDD is described in ISO 21597 [22, 23] a two-part standard, presently under development. Part 1 of the standard defines the container structure and the general linking concepts through the definition of a container ontology, corresponding data types and object properties, along with a linkset ontology with corresponding data types and properties. Part 2 defines additional types of links which form the extended linkset. ICDD follows a folder structure (refer Fig. 1): which three major components: an Ontology folder containing the schema of the files in the ICDD Zip file format [24], a linkset folder containing the links, and a payloads folder containing the documents themselves. A meta-file called index contains a summary of the ICDD container documents and how they are linked. In brief, the ICDD ontologies together with data types and properties define what kind of meta-information can be used for a file and the links between different files. ICDD includes both the Multi-model approach and the Linked Data approach [6].

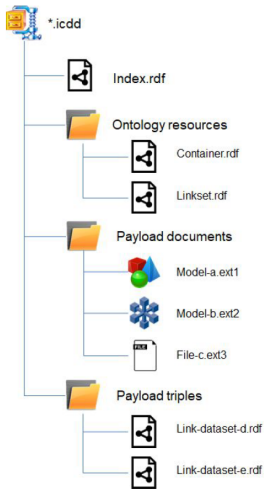


**Fig. 1.** The ICDD folder structure

### 2.2 Common Data Environment

According to the yearly industry report [16] of PlanGrid and FMI, based on their survey, construction professionals spend 5.5 hours per week looking for project data or information. Poor structuring of information, coordination and difficulty to find information results loss of the efficiency in the projects. CDEs provide an

interface for centralised management of information and services. The approach of a CDE is to give all project members with access to digital information and the ability to categorize information as needed. CDE implementations can be, for example, a project server, an extranet, or a cloud service. CDE is intended as a single source of information and a concerted network location for building project data collection, management, and distribution. It is based on the British standards PAS 1192-2 [11] and BS 1192 [27]. The documents describe the BIM Level 2 compliance requirements for construction company projects. [9]

**OpenCDE** German DIN SPEC 91391-2 [15] standard specifies OpenCDE, a list of conceptual requirements for the interface for communication between CDE, and between them and various software. It follows the pure client-server interaction design pattern [12]. The interface is specified in OpenAPI 3.0 format and defines a RESTful [17] API interface.

On the other hand, OpenCDE-API initiated by buildingSMART focus on simplifying the exchange between programs used in a construction project. DIN SPEC 91391-2 OpenCDE and buildingSMART OpenCDE-API initiative are separate works (Yoram Kulbak, personal communication, March 24, 2020). The main difference between the approaches is that the latter requires user interaction. In the latter's project's interface supports two flows: interactive and non- interactive. In the interactive flow, the user interacts with a web page (the client environment) directly. For example, a model author uses a client tool to upload model versions to the server manually. In the non-interactive flow, greater flexibility is available for implementing more dynamic features and workflows. The non-interactive server-to-server flows are still at the concept level. There is a publicly available OpenAPI 3.0 specification for the server interface available on the buildingSMART/OpenCDE-API GitHub repository[3] .

**Linked Data Platform** LDP is a specification containing definitions for reading-writing Linked Data architecture. As Bizer summarized in [7], "Linked Data is simply about using the Web to create typed links between data from different sources." However, it is also a collection of best practices. The core concept is based on the list of Linked data Rules stated by Tim Berners-Lee in [4]. The rules are: 1. Uniform Resource Identifiers (URIs) are used to name things uniquely. 2. Hypertext Transfer Protocol (HTTP) [18] is used so that people can find information about things in the same manner as they open web pages. 3. When following a link, useful information is provided using standards like Resource Description Framework (RDF) [26] format and SPARQL Query Language [31]. 4. In the reply, links to other URI's are provided to help to discover more information. Furthermore, Linked Data is based on the Restful stateless communication pattern presented in [17, 21, 25].

[3] https://github.com/buildingSMART/OpenCDE-API

An LDP can be a client or server or a combination of both while conforming to the specification set forth by the W3C Working Group of the same name. It focuses on the use of HTTP to Create, Read, Update, and Delete Linked Data Resources (files/documents) that are part of a collection (folder). For this, the W3C recommendation contains information on what resource content notations and content serialization formats should be used, how a client can handle changes to resources, container issues including data and meta-data retrieval using GET updating containers using POST.

Numerous available implementations of LDP exists such as Apache Marmotta, OpenLink Virtuoso, TopBraid Live etc. [1]. A very recent initiative, SOLID [5] is loosely based on the LDP server, which aims to promote a platform where users can store data in pods, which can be accessed by applications through authorizations. In this paper we have chosen to demonstrate LDP with SOLID.

## 3 ICDD in the CDE Context

### 3.1 ICDD in DIN SPEC 91391-2

Multi-containers are specified as a container type in DIN SPEC 91391-2. BIM-LV containers [14] are explicitly mentioned in the standard [15] and according to DIN SPEC 91350 are specific version of the multi-model container and correspond to the multi-model scheme published at [28].

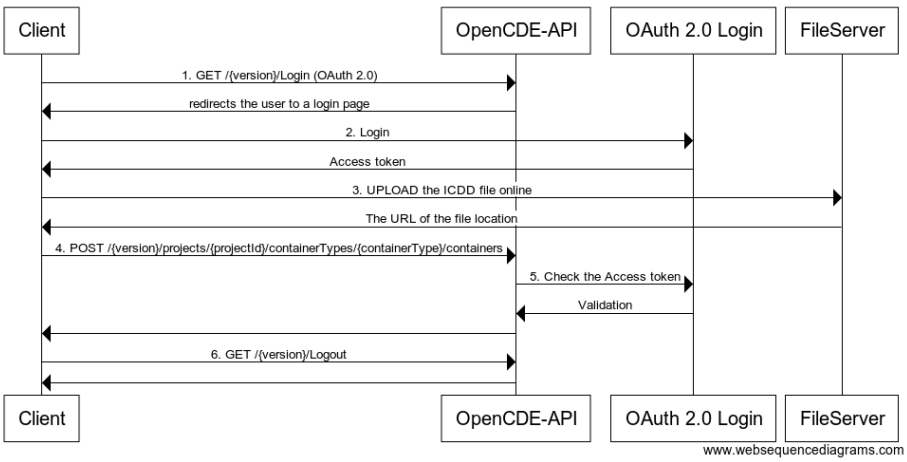


**Fig. 2.** Publish an ICDD container into a DIN SPEC 91391-2 Server

ICDD is also a multi-model and can be interpreted addressed implicitly at the standard [29]. For consistency, all multi-models should be published in the same

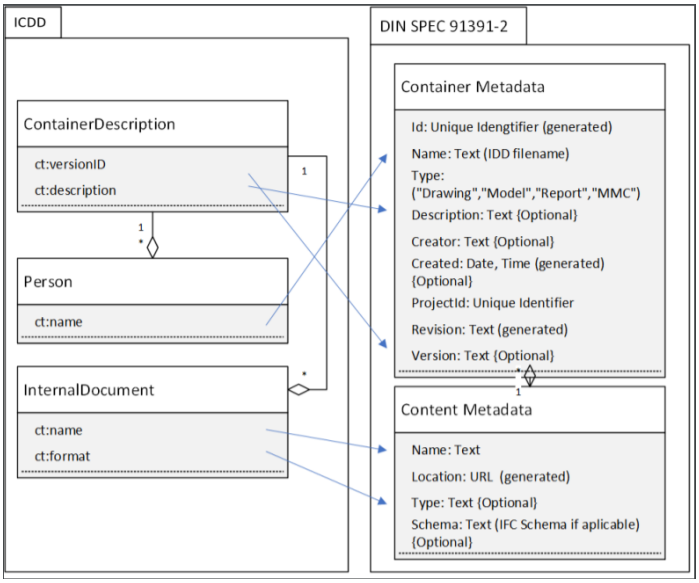


**Fig. 3.** Mapping between ICDD and the DIN SPEC 91391-2 OpenCDE metadata

way. There are also no restrictions for that. The standard defines that OpenCDE container types can and should be defined in the BIM Execution Plan (BEP) of the construction project. When set in BEP, ICDD is a legitimate OpenCDE container type. An advantage is that while published as an unmodified ICDD file container, there will be no information loss associated. The drawbacks is that there would be an additional metadata layer for the models, and the contents of the ICDD container will not be directly accessible online due to Zip encoding. The publication procedure is shown in Fig. 2. The required steps are 1. The user initiates OAuth 2.0 login. 2. Login is made using an OAuth 2.0 login server. This can be the authentication server of the user's employer. 3. The ICDD file is uploaded into a publicly available web address. The server returns a URL for the file. 4. Metadata for the container is created and uploaded to the OpenCDE server. 5. the OpenCDE checks the validity of the OAuth 2.0 token. 6. At the end, the user is logged out from the server.

The mapping shown in Fig. 3 can be used to create the contained metadata for the DIN SPEC 91391-2 OpenCDE container. Most of the mandatory data can be copied directly from the ICDD container. The file location URL is implementation-specific, and fields like the container ID can be generated. The release number can be inferred from the version number or created automatically. Only the project ID needs to be given by the user.

OData 4.0 [10] filters specified in DIN SPEC 91391-2 [15] offers an implementer specific ways to filter API results by metadata. OData contains operator to

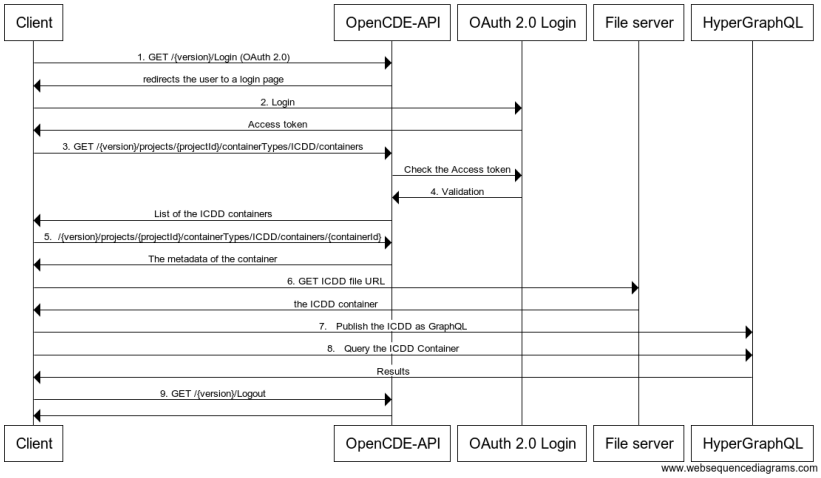


**Fig. 4.** Query an ICDD container published in a DIN SPEC 91391-2 Server

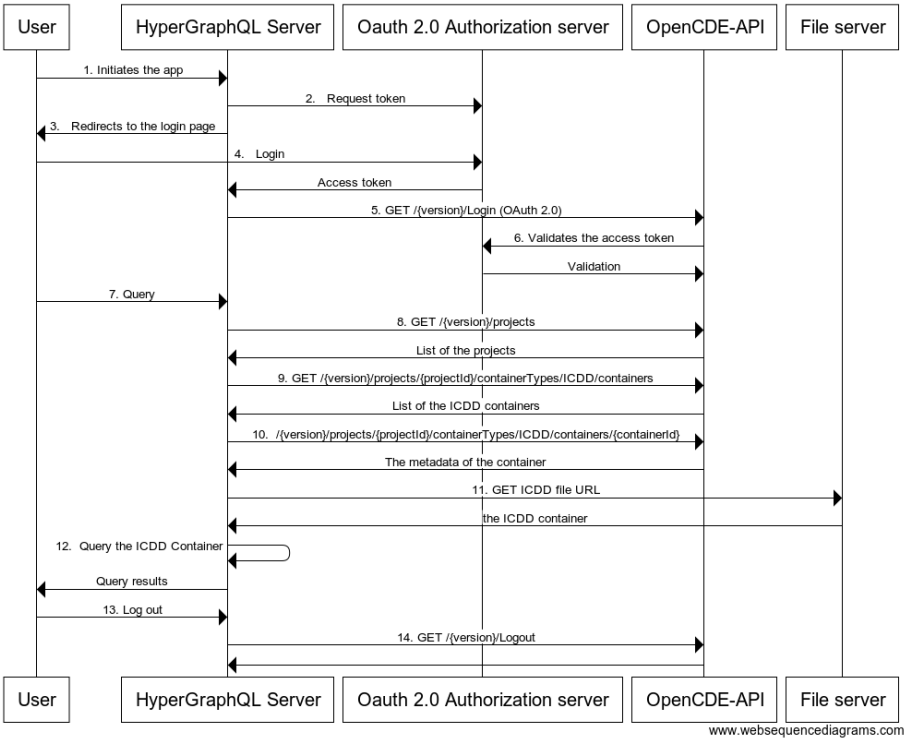


**Fig. 5.** Query an ICDD container published in a DIN SPEC 91391-2 Server

compare metadata parameters, operator for search and to sort the output. For example, it allows querying containers by upload time. Furthermore, as shown earlier in [34], the queried ICDD containers can be queried using the GraphQL[4]. The setup is shown in Fig. 4. Steps 1, 2, and 4 are for OAuth 2.0 user authentication. Step 3 gets a list of the ICDD containers in the project. The query can be extended with OData 4.0 filters. Steps 5 and 6 is to get the ICDD container. In steps 7-8, the ICDD container is published using HyperGraphQL server that was used earlier in [34] to query ICDD data. Since the above-described querying method has two phases and is not consistent, a more coherent approach was designed. The outline of the method is visualized in Fig. 5. In steps, 1-6 OAuth 2.0 login is handled. GraphQL query is given in 7. The steps 8-12: this part is using three facts: first, like shown by Wittern et al. in [35], a wrapper can be implemented to expose REST APIs as GraphQL, GraphQL can federate other GraphQL endpoints, and HyperGraphQL can be used to expose ICDD data content [34].

### 3.2 ICDD in buildingSMART OpenCDE-API

As illustrated in Fig. 6, the metadata model of the interactive flow mode of the buildingSMART OpenCDE-API has a flexible structure. It allows expressing any property-value pairs of standard data types. The data types are date-time, boolean, string, integer, enum, and number. Compared to DIN SPEC 91391-2 OpenCDA, the challenge is that OpenCDE-API assumes a user feeding metadata interactively. Thus they cannot be copied or mapped from the ICDD container like they can in the DIN SPEC 91391-2 OpenCDE case.

The Sequence graph (Fig. 7) shows the steps for uploading an ICDD container using the OpenCDE-API interface. Step 1 starts the upload document flow. In step 2, the ICDD file is registered. Then, in step 3, the user manually inputs all the metadata, and, in step 4, the file is finally uploaded to the server.

Fig. 8 shows the needed actions to fetch an ICDD file from the OpenCDE-API interface. In step 1, the user selects the ICDD container. This part could be skipped if the document reference URL for the file is already known. Then the document reference is got. It contains a list of versions of the document. In the last step, the actual file is retrieved.

### 3.3 ICDD in Linked Data Platform

LDP defines containers as a special resource, which has the capability to respond to HTTP requests [19] and modifications including, including the addition of new resources. There are two containers: *Basic Container*, and a *Direct Container*. The *Basic Container* can define links to a *Document resource* (both

[4] https://graphql.org/

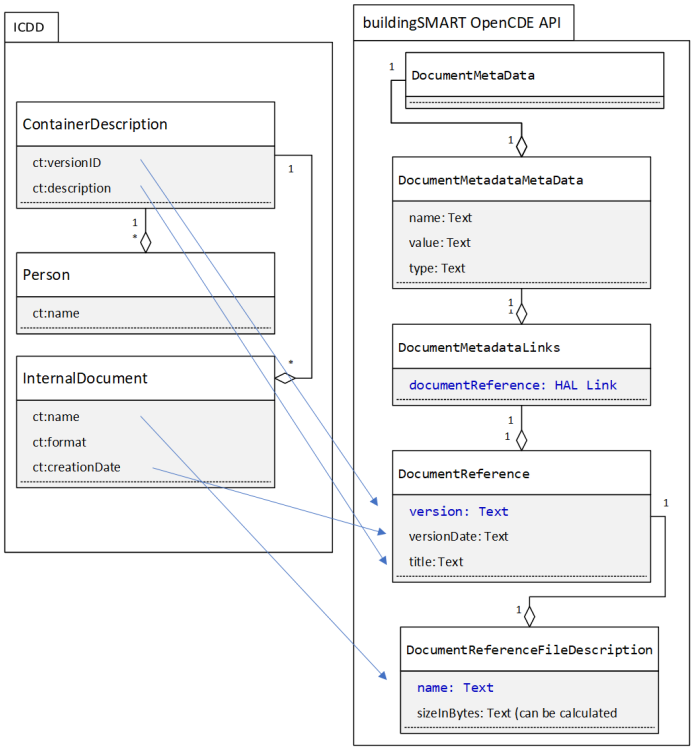


**Fig. 6.** Mapping between ICDD and the buildingSMART OpenCDE-API metadata

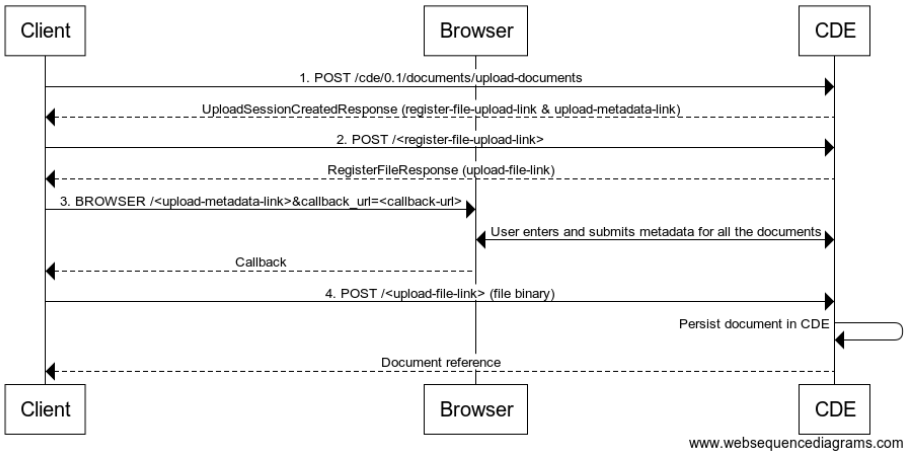


**Fig. 7.** Steps to upload an ICDD into buildingSMART OpenCDE-API

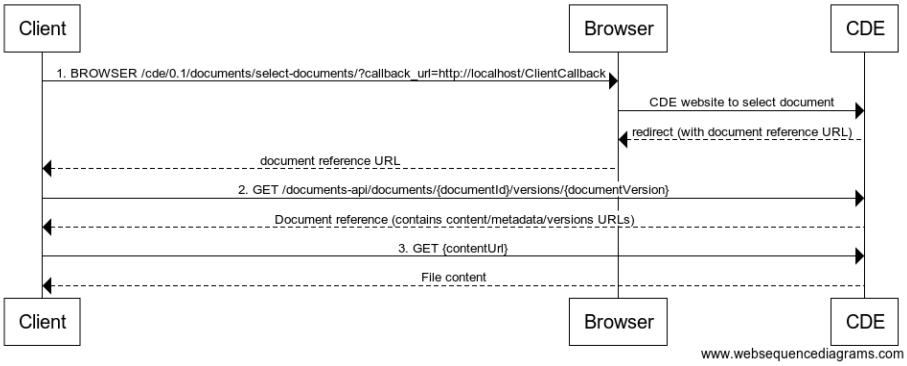


**Fig. 8.** Actions needed to fetch an ICDD from the OpenCDE-API

RDF and non-RDF) using *ldp:contains*. A Direct Container, features some additional features: it can be used to insert additional assertions (also called membership triples) using user-determined domain-specific vocabulary; it's membership triples can also be subject of another container resource; and a resource's facet can be managed using multiple containers. Hence, the user has the flexibility to define relationships beyond the ones defined in LDP, using any vocabulary.

Fig. 10 shows the workflow for publishing ICDD containers in the SOLID ecosystem, while Fig. 9 shows querying of information using the same system. In both these, a separate *ICDDShema.graphql* file has to be implemented, which contains all the Container and linkset ontologies from the ISO 21597 Part 1 and 2 [32].

In Fig. 2 and Fig. 9, the general operations for publishing and accessing data for further operations like querying were shown. Fig. 11 illustrates how the mapping of containers themselves differs. As explained in the beginning, LDP also contains the Container concepts, however ICDD explicitly defines the objects and properties which can be defined, along with type of links between documents in the container.

## 4 Comparison and Conclusions

Publishing ICDD information can be done using DIN SPEC 91391-2, OpenCDE-API, and LDP frameworks. There are a couple of LDP implementations available [2], while Oracle Aconex[5] is an buildingSMART OpenCDE implementation prototype, and there is none that precisely implements DIN SPEC 91391-2. DIN SPEC 91391-2 also leaves many things for the implementer to decide, including the syntax of the values returned by a REST call, and the set of supported

[5] https://bim.aconex.com/

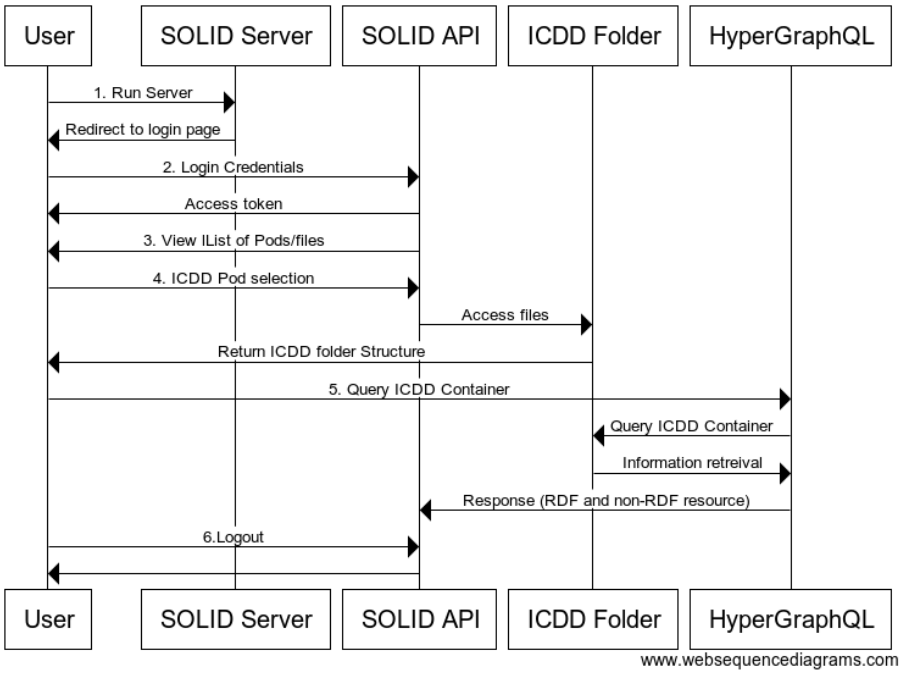


**Fig. 9.** Querying ICDD Container in SOLID

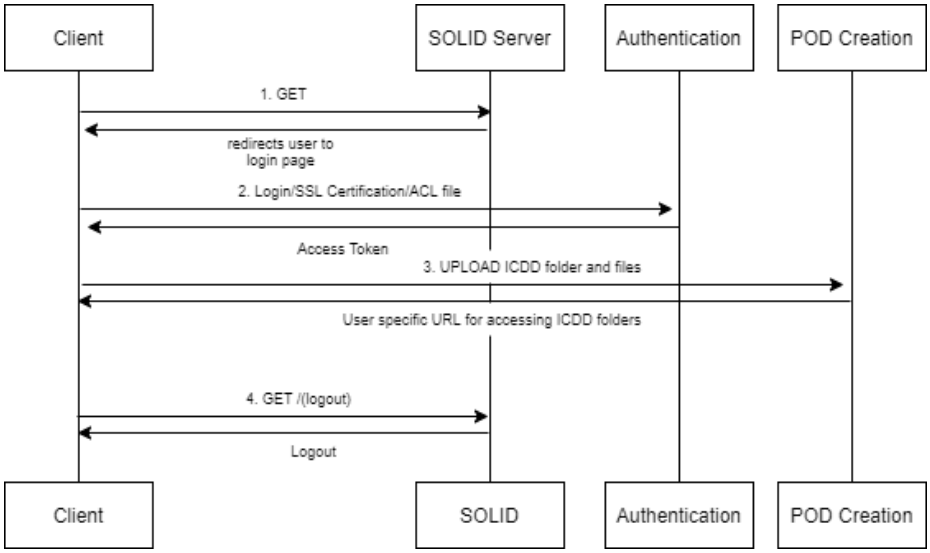


**Fig. 10.** Publishing ICDD Container in SOLID

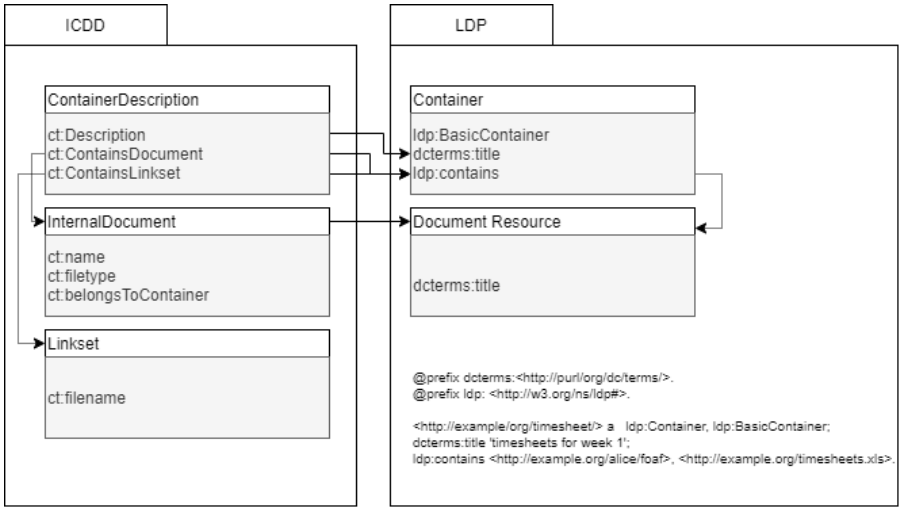


**Fig. 11.** Mapping between ICDD and LDP Container Concepts

OData filters.

The DIN SPEC 91391-2 container model is metadata-based. Compared to LDP, OpenCDE use external servers to save the data content. Furthermore, DIN SPEC 91391-2 has no mention of linked data, while "LDP defines a set of rules for HTTP operations, some based on RDF, to provide an architecture for read-write Linked Data on the Web." [33]. Additionally, DIN SPEC 91391-2 includes concepts for building projects, such as the project identifier, contains an explicit version handling for documents, and the multi-model container is well defined. OpenCDE-API interactive flow limits the possible automation that can be implemented. The metadata is flexible and can easily express ICDD metadata. Like LDP, it is very generic and does not have anything that is construction industry-specific.

The Container ontology and specification in LDP is broad. Anything can be a Container, and have corresponding membership links to resources. However, in building projects, more complex information is usually encoded such as "Type of link between files/documents", "sub-document level linkages" etc. Due to LDP's flexibility in defining link relationships on any domain-specific vocabulary, links in a Container and their interpretation are left to the discretion of the creator. Hence, a schema for interpretation of the links also has to be supplied by the creator. In ICDD, a set of standard types of links, what information should each resource at-least contain is defined by the standard. ICDD's definition of container ontologies, the specific data types and properties which can be associated with them along with the linkset ontology which provides the type of linking facilitates a well-structured approach for linking heterogeneous data us-

ing multi-model Containers. ICDD schema can be seen as an extension of the existing container ontologies defined in the DIN spec and LDP.

The success of any CDE implementation ultimately rests with the CDE vendors who will have to agree on how their systems exchange data. Approaches such as the OpenCDE-API and LDP provides a road map for standardizing CDEs and the features that will have to be taken care of while implementing them. The standardized structure of ICDD can help in extending the initial definitions of Containers in CDE, thus making it easy to implement basic functions such as querying, modifying and deleting heterogeneous data.

## 5 Acknowledgements

This research was funded by the EU through the H2020 project BIM4REN.

## References

1. LDP implementations - w3c wiki, https://www.w3.org/wiki/LDP_Implementations
2. Bader, S.R., Wolf, A., Keppmann, F.L.: Evaluation environment for linked data web services. In: SEMANTICS Workshops (2017)
3. Beckett, D., McBride, B.: Rdf/xml syntax specification (revised). W3C recommendation **10**(2.3) (2004)
4. Berners-Lee, T.: Linked Data - Design Issues (Jun 2008), https://www.w3.org/DesignIssues/LinkedData.html
5. Berners-Lee, T., Prud'hommeaux, E., Carvalho, M., Verborgh, R.: Solid (2017), https://solid.mit.edu/
6. Bizer, C., Heath, T., Berners-Lee, T.: Linked data - the story so far. International Journal on Semantic Web and Information Systems **5**, 1–22 (2009)
7. Bizer, C., Heath, T., Berners-Lee, T.: Linked data-the story so far. Semantic services, interoperability and web applications: emerging concepts pp. 205–227 (2009)
8. Bizer, C., Heath, T., Berners-Lee, T.: Linked data. Evolving the Web into a Global Data Space (2011)
9. Boxall, E.: Common data environment (cde): What you need to know for starters. https://blogs.oracle.com/construction-engineering/common-data-environment-cde-tutorial (2018), [Online; accessed 11-Fev-2029]
10. Chappell, D.: Introducing odata. Data Access for the Web, The Cloud, Mobile Devices, and More pp. 1–24 (2011)
11. Council, C.I.: Pas1192-2: 2013 specification for information management for the capital/delivery phase of construction projects using building information modelling (2013)
12. Daigneau, R.: Service Design Patterns: fundamental design solutions for SOAP/WSDL and restful Web Services. Addison-Wesley (2012)
13. D'Esposito, H.: The state of construction technology - 2019 (2019)
14. Din spec 91350:2016-11, verlinkter bim-datenaustausch von bauwerksmodellen und leistungsverzeichnissen (Nov 2016). https://doi.org/https://dx.doi.org/10.31030/2581152

15. Din spec 91391-2:2019-04, gemeinsame datenumgebungen (cde) für bimprojekte – funktionen und offener datenaustausch zwischen plattformen unterschiedlicher hersteller – teil 2: Offener datenaustausch mit gemeinsamen datenumgebungen (Apr 2019). https://doi.org/10.31030/3044838
16. Erik Thomas, Peter Schott, J.B.J.S., Spare, N.: 2018 industry report: Construction disconnected (2018)
17. Fielding, R.T.: Architectural Styles and the Design of Network-based Software Architectures. Doctoral dissertation, University of California, Irvine (2000)
18. Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., Berners-Lee, T.: Hypertext transfer protocol–http/1.1 (1999)
19. Fielding, R.T., Gettys, J., Mogul, J.C., Nielsen, H.F., Masinter, L., Leach, P.J., Berners-Lee, T.: Hypertext transfer protocol – http/1.1. RFC 2616, RFC Editor (June 1999), http://www.rfc-editor.org/rfc/rfc2616.txt, [Online; accessed 06-June-2019]
20. Fuchs, S., Katranuschkov, P., Scherer, R.: A framework for multi-model collaboration and visualisation. Proc. ECPPM 2010 pp. 115–120 (2010)
21. Hausenblas, M.: Linked data applications. First Community Draft, DERI (2009)
22. Information container for linked document delivery — Exchange specificaton - Part 1: Container (Mar 2018)
23. Information container for linked document delivery — Exchange specificaton - Part 2: Dynamic semantics (Mar 2018)
24. Katz, P.: Appnote. TXT—. ZIP file format specification, version **2** (1993)
25. Malhotra, A., Arwe, J., Speicher, S.: Linked data platform specification. W3C Recommendation (2015)
26. Manola, F., Miller, E., McBride, B., et al.: Rdf primer. W3C recommendation **10**(1-107), 6 (2004)
27. Richards, Richards, M.: Building information management: A standard framework and guide to bs 1192, bsi standards (2010)
28. Schapke, S.E.: jyrkioraskari/MMC: buildingSMART MMC Multimodell-Container (Mar 2020). https://doi.org/10.5281/zenodo.3727286, https://doi.org/10.5281/zenodo.3727286
29. Scherer, R.J., Katranuschkov, P.: Context capturing of multi information resources for the data exchange in collaborative project environments (2019)
30. Scherer, R.J., Schapke, S.E.: A distributed multi-model-based management information system for simulation and decision-making on construction projects. Advanced Engineering Informatics **25**(4), 582–599 (2011)
31. Seaborne, A., Manjunath, G., Bizer, C., Breslin, J., Das, S., Davis, I., Harris, S., Idehen, K., Corby, O., Kjernsmo, K., et al.: Sparql/update: A language for updating rdf graphs. W3c member submission **15** (2008)
32. Senthilvel, M.: Icdd hypergraphql (May 2020). https://doi.org/10.5281/zenodo.3846727, https://github.com/Design-Computation-RWTH/ICDD-HyperGraphQL
33. Speicher, S., Arwe, J., Malhotra, A.: Linked data platform 1.0. W3C Recommendation, February **26** (2015)
34. Werbrouck, J., Senthilvel, M., Beetz, J., Pauwels, P.: Querying heterogeneous linked building datasets with context-expanded graphql queries. In: 7th Linked Data in Architecture and Construction Workshop. pp. 21–34 (2019)
35. Wittern, E., Cha, A., Laredo, J.A.: Generating graphql-wrappers for rest (-like) apis. In: International Conference on Web Engineering. pp. 65–83. Springer (2018)