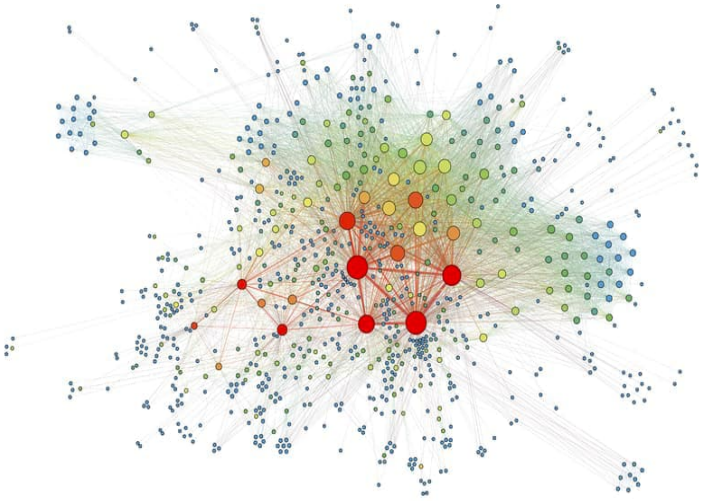
RWTH Aachen University

DISSERTATION
Linked Building Data Management



Graph image: Martin Grandjean

Jyrki Oraskari
2024

Linked Building Data Management

Von der Fakultät für Architektur der Rheinisch-Westfälischen Technischen Hochschule Aachen zur Erlangung des akademischen Grades eines Doktors der Ingenieurwissenschaften genehmigte Dissertation

vorgelegt von

Jyrki Oraskari

Berichtende: Univ.-Prof. Dr. Jakob Beetz

Univ.-Prof. Dr. techn. Sigrid Brell-Cokcan

Tag der mündlichen Prüfung: 19.07.2024

Diese Dissertation ist auf den Internetseiten der Universitätsbibliothek online verfügbar.

Author's Rights Declaration

Mathias Bonduel, Jyrki Oraskari, Pieter Pauwels, Maarten Vergauwen, and Ralf Klein. The ifc to linked building data converter - current status. *Proceedings of the 6th Linked Data in Architecture and Construction Workshop*, 2159:34–43, 2018. ISSN 1613-0073. URL https://ceur-ws.org/Vol-2159/04paper.pdf. International Workshop on Linked Data in Architecture and Construction, LDAC ; Conference date: 19-06-2018 Through 21-06-2018, CEUR-WS Team, Redaktion Sun SITE, Informatik V, RWTH Aachen, Germany. The authors of the material have preserved all their copyrights and publication rights in agreement with the organizers of the Linked Data in Architecture and Construction workshop.

Pierre Bourreau and Jyrki Oraskari. Bim format conversion as inference-based ontology alignment. In *Proceedings of the 9th Linked Data in Architecture and Construction Workshop - LDAC2021, CEUR-WS Team, Redaktion Sun SITE, Informatik V, RWTH Aachen, Germany.*, 2021b. URL https://ceur-ws.org/Vol-3081/11paper.pdf. Online; accessed Oct. 15. 2023. The published work was made available as an open-access publication under the Creative Commons License Attribution 4.0 International (CC BY 4.0). The Linked Data Architecture and Construction Workshop 2020 editors, RWTH Aachen, CEUR-WS and archiving partners were given irrevocable rights to make the work available.

Vincent Kuo and Jyrki Oraskari. A predictive semantic inference system using bim collaboration format (bcf) cases and machine learning. In Arto Saari and Pekka Huovinen, editors, *WBC16 Proceedings : Volume III*, volume 3 of *Raportti / Tampereen teknillinen yliopisto, rakennustekniikan laitos, rakennustuotanto ja -talous*, pages 368–381. Tampere University of Technology, Tampere, Finland, 2016. URL https://trepo.tuni.fi//bitstream/handle/10024/128445/WBC16_Vol_3.pdf. CIB World Building Congress, WBC ; Conference date: 30-05-2016 Through 03-06-2016, Online; accessed Oct. 15. 2023. The authors kept the rights of the work (for the proceedings, copyright 2016 TUT Tampere University of Technology with the comment: "permission from the publishers or, in the case of individual papers, from the author(s) of that paper.".

J Oraskari and S Törmä. Access Control for Web of Building Data: Challenges and Directions. In Symeon Christodoulou and Raimar Scherer, editors, *eWork and eBusiness in Architecture, Engineering and Construction: ECPPM 2016*. CRC Press, London, UK, 2016b. ISBN 9781138032804, https://doi.org/10.1201/9781315386904. The author has retained the right to republish the Contribution in any printed collection consisting solely of the Author's Works without charge and subject only to notifying the Publisher of the intent to do so and ensuring that the publication by the publisher is appropriately credited and that the relevant copyright notice is repeated verbatim. The publisher, Taylor & Francis Groups, has been notified, and they have affirmed the use as long as the following text is added: This is an Accepted Manuscript of an article published by Taylor & Francis in eWork and eBusiness in Architecture, Engineering and Construction: ECPPM 2016 on 28 March 2017, available online: https://www.taylorfrancis.com/chapters/edit/10.1201/9781315386904-10/access-control-web-building-data-challenges-directions-oraskari-t.

Jyrki Oraskari. Computationally efficient ifc file comparison using rdf graph paths. In *ITC Digital library*, pages 599–608, 2015. URL https://itc.scix.net/paper/w78-2015-paper-062. CIB W78 Conference , CIB W78, Eindhoven, The Netherlands; Conference date: 27-10-2015 Through 29-10-2015, Online; accessed Oct. 15. 2023. The copyright of the full-text papers in this digital library is with the authors or with the original publishers of the proceedings and journals.

Jyrki Oraskari. Live web ontology for buildingsmart data dictionary. In *32. Forum Bauinformatik 2021. Institut für Numerische Methoden und Informatik im Bauwesen, TU Darmstadt (2022)*, volume 32, pages 166–173, July 2021. doi: 10.26083/tuprints-00019496. URL https://tuprints.ulb.tu-darmstadt.de/21521/. Forum Bauinformatik, FBI, Darmstadt, Germany; Conference date: 09-09-2021 Through 10-09-2021, Online; accessed Oct. 15. 2023. The work was made available as an open-access publication under the Creative Commons License Attribution 4.0 International (CC BY 4.0).

Jyrki Oraskari and Seppo Törmä. Rdf-based signature algorithms for computing differences of ifc models. *Automation in Construction, Elsevier B.V. Amsterdam, Netherlands*, 57:213–221, 2015b. ISSN 0926-5805. doi: https://doi.org/10.1016/j.autcon.2015.05.008. URL https://www.sciencedirect.com/science/article/pii/S0926580515001181. Online. Accessed Oct. 20. 2023. Elsevier allows the inclusion of the paper in the dissertation for non-commercial purposes.

Jyrki Oraskari, Jakob Beetz, and Seppo Törmä. Federated Building Models in collaborative AEC/FM settings using IPFS. In *[17th International Conference on Computing in Civil and Building Engineering, 2018-06-05 - 2018-06-07, Tampere, Finland]. Galway, Ireland: Galway, Ireland: Ex Ordo*. 17th International

Conference on Computing in Civil and Building Engineering, Tampere (Finland), 5 Jun 2018 - 7 Jun 2018, Ex Ordo, Galway, Ireland, Jun 2018b. URL http://programme.exordo.com/icccbe2018/delegates/presentation/354/. Accessed on Sep. 05. 2023. The authors have kept the rights of the work. This was ensured by the conference organizer, Ville Raasakka, RILFinnish Association of Civil Engineers, during a phone call on November 8, 2024.

Jyrki Oraskari, Madhumitha Senthilvel, and Jakob Beetz. Shacl is for lbd what mvdxml is for ifc. In *Proceedings of the 38th International Conference of CIB W78, Luxembourg, 13-15 October*, CIB W78 conference series, pages 693–702. CIB International Council for Research and Innovation in Building and Construction, 2021b. URL https://www.cibw78-ldac-2021.lu/. CIB W78 Conference ; Conference date: 11-10-2021 Through 15-10-2021, Online; accessed Oct. 15. 2023. The copyright of the full-text papers in this digital library is with the authors or with the original publishers of the proceedings and journals.

Jyrki Oraskari, Oliver Schulz, and Jakob Beetz. Towards describing version history of BCF data in the semantic web. In *Proceedings of the 10th Linked Data in Architecture and Construction Workshop co-located with 19th European Semantic Web Conference (ESWC 2022). CEUR-WS Team, Redaktion Sun SITE, Informatik V, RWTH Aachen, Germany*. CEUR-WS, 2022a. https://ceur-ws.org/Vol-3213/paper08.pdf. Accessed November 8, 2024. As agreed with the LDAC 2022 conference organizers, the work was made available as an open-access publication under the Creative Commons License Attribution 4.0 International (CC BY 4.0)

Jyrki Oraskari, Oliver Schulz, Jeroen Werbrouck, and Jakob Beetz. Enabling federated interoperable issue management in a building and construction sector. In *Proceedings of the 29th EG-ICE International Workshop on Intelligent Computing in Engineering. Aarhus University Library/Det Kgl. Bibliotek - AU Library Scholarly Publishing Services. Aarhus, Denmark*, pages 92–101. EG-ICE, 2022b. ISBN 978-87-7507-521-8. URL https://ebooks.au.dk/aul/catalog/view/455/312/1848-2. http://doi.org/10.7146/aul.455.c200, Online; accessed Oct. 15. 2023. The authors have kept the copyright for the work: The copyright statement is here: https://ebooks.au.dk/aul/catalog/book/455.

Jyrki Oraskari, Seppo Törmä, Mehmet Yalcinkaya, Vishal Singh, and Jakob Beetz. Sharing linked building data in a peer-to-peer network: ifcowl meets interplanetary file system. *Journal of Computing in Civil Engineering. 1801 Alexander Bell Drive Reston, VA 20191-4400*, 38(1):04023037, 2024. doi: 10.1061/JCCEE5.CPENG-5381. URL https://ascelibrary.org/doi/abs/10.1061/JCCEE5.CPENG-5381. published online on September 28, 2023. ASCE copyrighted content includes all ASCE publications. "Thesis or Dissertation: Reuse is

permitted, provided the content in total does not account for more than 25% of the new work.".

bcfOWL: A BIM collaboration ontology, 2021. URL https://ceur-ws.org/Vol-3081/12paper.pdf. International Workshop on Linked Data in Architecture and Construction, LDAC; Conference date: 11-10-2021 Through 15-10-2021, Online; accessed Nov. 08. 2024. The article was made available as an open-access publication under the Creative Commons License Attribution 4.0 International (CC BY 4.0)

Madhumitha Senthilvel, Jyrki Oraskari, and Jakob Beetz. Common data environments for the information container for linked document delivery. *Proceedings of the 8th Linked Data in Architecture and Construction Workshop - LDAC2020. CEUR-WS Team, Redaktion Sun SITE, Informatik V, RWTH Aachen, Germany*, 2636:132–145, June 2020. ISSN 1613-0073. URL https://ceur-ws.org/Vol-2636/10paper.pdf. International Workshop on Linked Data in Architecture and Construction, LDAC; Conference date: 17-06-2020 Through 19-06-2020, Online; accessed Oct. 20. 2023. The work was made available as an open-access publication under the Creative Commons License Attribution 4.0 International (CC BY 4.0)

Madhumitha Senthilvel, Jyrki Oraskari, and Jakob Beetz. Implementing information container for linked document delivery (icdd) as a micro-service. In *EG-ICE 2021 Proceedings*, page 6675. Universitätsverlag der TU Berlin, ISBN 978-3-7983-3212-6, August 2021. doi: 10.14279/depositonce-12021. URL https://berlin-2021.eg-ice.org/. International Workshop on Intelligent Computing in Engineering, EG-ICE ; Conference date: 30-06-2021 Through 02-07-2021, Online; accessed Oct. 15. 2023. The authors were granted the non-exclusive right of use to the Technische Universität Berlin, which does not affect the author's right to publish the work in other ways. The work is published under the Creative Commons license CC BY 4.0.

Jeroen Werbrouck, Oliver Schulz, Jyrki Oraskari, Erik Mannens, Pieter Pauwels, and Jakob Beetz. A generic framework for federated cdes applied to issue management. *Advanced Engineering Informatics. Elsevier, Amsterdam, Netherlands*, 58:102136, 2023. ISSN 1474-0346. doi: https://doi.org/10.1016/j.aei.2023.102136. URL https://www.sciencedirect.com/science/article/pii/S1474034623002641. Elsevier allows the inclusion of the paper in the dissertation for non-commercial purposes.

Appropriate references are given in the respective figure captions of the data presented. D 82 (Diss. RWTH Aachen University, 2024)

I hereby declare that I have created this work completely on my own and used no other sources or tools than the ones listed and that I have marked any citations accordingly.

Hiermit versichere ich, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie Zitate kenntlich gemacht habe.

Aachen, August 2023
Jyrki Oraskari

Contents

Rights I

Abstract XXI

Überblick XXIII

Acknowledgements XXV

1 Introduction 1

1.1 The Building Information model (BIM) 8

1.2 The Open BIM 10

1.3 Linked Data 12

1.4 Linked Building Data 13

1.5 Linked Data Management 16

2 Conversion and findability of Linked Building Data 23

2.1 The IFC to Linked Building Data converter-current status (LDAC 2018) 26

Bibliography 43

2.2 Dynamic BIM format conversion as inference-based ontology alignment (LDAC 2021) 47

Bibliography 64

2.3 SHACL is for LBD what mvdXML is for IFC (CIB W78 2021) 70

Bibliography 88

2.4 A Predictive Semantic Inference System Using BIM Collaboration Format (BCF) Cases and Machine Learning (WBC 2016) 93

2.5 Discussion and Conclusions 111

3 Linked Building Data and Ontologies 115

3.1 bcfOWL: A BIM collaboration ontology (LDAC 2021) 117

Bibliography 135

3.2 Towards describing version history of BCF data in the Semantic Web (LDAC 2022) 138

Bibliography 155

3.3 Live Web Ontology for buildingSMART Data Dictionary (FBI 2021) . 158

Bibliography 172

3.4 Related publications 176

3.5 Discussion and Conclusions 179

4 Publish Data as Common Data Environment 182

4.1 Common Data Environment Concepts 185

4.2 Common Data Environments for the Information Container for linked Document Delivery (LDAC 2020) 187

Bibliography 204

4.3 Implementing Information Container for linked Document Delivery (EG-ICE 2021) 208

4.4 Enabling Federated Interoperable Issue Management in a Building and Construction Sector (EG-ICE 2022) 225

4.5 A Generic Framework for Federated CDEs applied to Issue Management (AEI 2023) 244

Bibliography 281

4.6 Discussion and Conclusions 291

5 Information Security Aspects: Confidentiality 293

5.1 Access control for the web of building data: Challenges and directions (ECPPM 2016) 294

5.2 Discussion and Conclusions 318

6 Information Security Aspects: Integrity 320

6.1 RDF-based signature algorithms for computing differences of IFC models (AC 2015) 323

Bibliography 351

6.2 Computationally Efficient IFC File Comparison Using RDF Graph Paths (CIB W78 2015) 357

6.3 Discussion and Conclusions 377

7 Information Security Aspects: Availability 380

7.1 Federated Building Models in collaborative AEC/FM settings using IPFS (ICCCBE 2018) 383

Bibliography 399

7.2 Sharing Linked Building Data in a Peer-to-Peer Network — ifcOWL meets InterPlanetary File System (J. Comput. Civ. Eng.) 408

Bibliography 437

7.3 Discussion and Conclusions 445

8 Discussion and Conclusions **448**

List of Publications **454**

Bibliography **458**

Index **475**

List of Figures

1.1 Summary of use cases in the work. 6

1.2 Traditional construction handover documentation (Photo by Cornhenge in Wikimedia Commons). 6

1.3 Semantic connection between published articles and the overall structure of my dissertation work. 20

1.4 An overview of my published dissertation research work. 21

2.1 How the research questions, hypotheses, and their handling in articles and the sub-sections are connected. 25

2.2 My articles related to publishing and facilitating queries at a glance (For the term Abox, see site https://www.lesliesikos.com/knowledge-base/) 26

2.3 Implemented IFCtoLBD conversion process regarding BOT 44

2.4 The BIM4Ren data model architecture 56

2.5 Example of alignment with IFD through bsDD 58

2.6 Example of deobjectification of relations 58

2.7 The schema mapping from IFC to BOT. 81

3.1 How the research questions, hypotheses, and their handling in articles and the sub-sections are connected. 116

3.2 My articles related to publishing ontologies at a glance 117

3.3 General structure of the BCF. The dark blue boxes form the *Issue*. Each hierarchy level represents a request to the server in the BCF API. 127

3.4 A snippet from the ontology, displaying the *Topic* as a graph 129

3.5 Jyrki creates a new Topic, which is the most current version of it now. The Topic contains parameters such as the creation date, a title, two labels (Architecture and Heating) and a status set to "Active". On the next day, Oliver sets the Topic's status to "Closed" and removes the label "Heating". Another day later, the Topic gets reopened (set to "Active") by Jyrki and the label "Documentation" gets added. 147

3.6 The Event System approach is based on the BCF API. A Topic Event points to the related Topic and its operations that specifically define what type of interactions have happened. For example, if a property was removed, updated or created. Next to the type of operation, the attribute's value is stated. 148

3.7 The different states are referenced by the Topic with the property bcfOWL:hasTopicState. The states provide the complete picture of the Topic during a specific time. In order to find the current version of a Topic we have to query for the newest date. An example query is provided on the GitHub repository. 149

3.8 Statements can be marked as added or removed with a time-stamp. The author of this change is added as a statement of the first statement. Even though the word itself states a deletion, resources are never removed from the graph to allow retrieval of the complete history of the Topics. 150

3.9 Object property annotations only contain one level of statements and point to an object that contains all change-related information. 151

3.10 bSDD OWL OpenAPI implementation. When the user (on the left) uses client software to send a query to the ontology generator API (at the middle), the query is translated and conveyed to the buildingSMART API back-end. The result is then mapped into the OWL ontology description language and send to the client. 167

3.11 The OpenAPI SWAGGER user interface of the solution. The first screen on the left shows the introductory page. In the middle is shown the result for the query on the user interface. On the right, the API output in OWL in Turtle - Terse RDF Triple Language is shown when the API is accessed directly. 168

3.12 The software architecture for the implementation. 169

3.13 The Widoco generated and human-readable description of the ontology results from an automated process that uses a generic document template to compose a human-friendly version of the on-the-fly generated OWL ontology description from the API. The automated process allows the data to be up to date and align with the machine-readable OWL. 170

3.14 The generated ontology description for IfcDoor. 171

3.15 The generated ontology description for a property set associated with IfcDoor. 171

4.1 How the research questions, hypotheses, and their handling in articles and the sub-sections are connected. 183

4.2 My articles related to publishing data as CDE at a glance 184

4.3 The ICDD folder structure 194

4.4 Publish an ICDD container into a DIN SPEC 91391-2 Server 196

4.5 Mapping between ICDD and the DIN SPEC 91391-2 OpenCDE metadata 197

4.6 Query an ICDD container published in a DIN SPEC 91391-2 Server . 198

4.7 Query an ICDD container published in a DIN SPEC 91391-2 Server . 198

4.8 Mapping between ICDD and the buildingSMART OpenCDE-API metadata 200

4.9 Steps to upload an ICDD into buildingSMART OpenCDE-API 200

4.10 Actions needed to fetch an ICDD from the OpenCDE-API 201

4.11 Querying ICDD Container in SOLID 202

4.12 Publishing ICDD Container in SOLID 202

4.13 Mapping between ICDD and LDP Container Concepts 203

4.14 Abstraction of buildingSMART's vision of OpenCDE APIs, originally presented by Yoram Kulbak and Pasi Paasiala in October 20191. The BCF API is part of buildingSMARTs OpenCDE APIs. 230

4.15 LBDserver patterns for project discovery. The project access point allows finding contributions in the federated project network. . . . 235

4.16 Comparison between the structure of the BCF API and the container-based approach using bcfOWL. 236

4.17 Abstraction of buildingSMART's OpenCDE APIs, originally presented by Kulbak and Paasiala in October 2019. The BCF API is part of buildingSMARTs OpenCDE APIs. 253

4.18 Generic CDE Framework 265

4.19 Graphical discovery patterns of other vaults in the project. 266

4.20 Left-hand side: geometric clash related to 'Openings and Recesses' present in the Schependomlaan dataset. Right-hand side: geometry after the issue is resolved. 271

4.21 Sequence Diagram of the requests a client application (Application Layer) makes to a BCF API implementation (Middleware Layer), which aggregates data from the vaults in the project (Storage Layer) and restructures ('adapts') this information to match the requirements of the BCF specification. The sequence corresponds with the scenario described in 4.5.5. 274

5.1 My article related to information security aspects of data management related to confidentiality at a glance 294

6.1 My articles related to information security aspects of data management related to integrity at a glance 323

6.2 Exporting native models into a common conceptual model and representation 330

6.3 Listing: The placement checksum calculation 365

6.4 Listing: The division of the graph 366

6.5 Listing: A function to find which IRIs a blank node is dependent to . 367

6.7 Listing: Short Paths algorithm 367

6.6 Listing: Short Paths blank node naming algorithm 376

7.1 My article related to information security aspects of data management related to availability at a glance 382

7.2 The summarized comparison of decentralized storage systems . . . 400

7.3 The publication process for a new IFC file. 401

7.4 The proportional time usage in publishing an IFC model. 401

7.5 The Filtered triple count versus the total IFC publication time 402

7.6 The model entity count versus the IPFS nodes creating time 402

7.7 The model entity count versus the total IFC publication time 403

7.8 The four RDF triple snippets have the same meaning but a different hash. 422

7.9 On the left is a minimalistic IFC model and on the right the equivalent ifcOWL graph. 423

7.10 The composed directed acyclic subgraphs. 424

7.11 The Turtle - Terse RDF Triple representation of the subgraphs. . . . 424

7.12 The average partition coverage and the overlap percentage in the partitioned models based on the converted 102 error-free and unfiltered IFC2X3 and IFC4 models in Open IFC Model Repository . 426

7.13 The standard form Turtle representation of the subgraphs. 427

7.14 Linked Data access to the model. 428

7.15 Linked Data access using indirect links to share data. 429

7.16 The publication process for a new IFC file. Modified from the original work in [Oraskari et al., 2018a]. 430

7.17 Test setup to measure the download times of the complete models. 431

7.18 The download time of the sequential element-wise of the whole test model. Results are based on 100 runs. 432

7.19 The outliers can be seen in the plot of the consecutive download runs. 433

7.20 The size of the geometry filtered model versus the created IPFS File count. The results are based on the 102 error-free and unfiltered IFC2X3 and IFC4 models in Open IFC Model Repository in Auckland. 434

7.21 The download time of the concurrent element-wise of the whole test model. The results are based on one hundred runs. 435

7.22 The file publication times measures using the IPFS add command. The results are based on the 102 error-free and unfiltered IFC2X3 and IFC4 models in Open IFC Model Repository in Auckland. 435

7.23 The performance comparison of the download times of the larger IFC files. The results are based on one hundred runs. 436

List of Tables

2.1 The work of Bonduel et al. [Bonduel et al., 2018] in the Hevner et al. [Hevner et al., 2004] design science research framework 29

2.2 Corresponding ifcOWL and LBD classes 39

2.3 ifcOWL property paths and corresponding LBD relation or property path 40

2.4 IFCtoLBD converter output results per module 42

2.5 The work of Bourreau & Oraskari [Bourreau and Oraskari, 2021a] in the Hevner et al. [Hevner et al., 2004] design science research framework 48

2.6 Summary of expected relations - generated with IFC2BOT 60

2.7 Performance test results (times in ms) 61

2.8 The work of Oraskari et al. [2021b] in the Hevner et al. [2004] design science research framework 72

2.9 The work of Kuo and Oraskari [2016] in the Hevner et al. [2004] design science research framework 94

3.1 The work of Schulz et al. [2021] in the Hevner et al. [2004] design science research framework 120

3.2 The work of Oraskari et al. [2022a] in the Hevner et al. [2004] design science research framework 139

3.3 Comparison of the bcfOWL version description approaches. 153

3.4 The work of Oraskari [2021] in the Hevner et al. [2004] design science research framework 159

4.1 The work of Senthilvel et al. [2020] in the Hevner et al. [2004] design science research framework 188

4.2 The work of Senthilvel et al. [2021] in the Hevner et al. [2004] design science research framework 209

4.3 1 Consolidated Functionalities as specified by ICDD, OpenCDE-API and LDP 217

4.4 Applicable container operations (as specified by W3C) for the ICDD-LDP implementation 1/2 220

4.5 Applicable container operations (as specified by W3C) for the ICDD-LDP implementation 2/2 221

4.6 The work of Oraskari et al. [2022b] in the Hevner et al. [2004] design science research framework 226

4.7 The work of Werbrouck et al. in the Hevner et al. [2004] design science research framework 244

4.8 Comparison between different collaborative AECO platforms in research 263

4.9 Results of the query in Listing 4.6, based on the data in Listings 4.4 and 4.5. Rows in the table represent found Topic States. 273

5.1 The work of Oraskari and Törmä [2016a] in the Hevner et al. [2004] design science research framework 295

5.2 Table 1 - Model publication use case 303

5.3 Table 2 – Cross-model linking use case 305

5.4 Table 3 – Product data use case 306

5.5 Table 4 – Status sharing use case 308

6.1 The work of Oraskari and Törmä [2015a] in the Hevner et al. [2004] design science research framework 326

6.2 The uniqueness of the signatures 341

6.3 The used data sets 347

6.4 The estimates of the minimum change sets 348

6.5 Results of the first test case 348

6.6 Results of Solibri Model Checker Test Data 348

6.7 Results of the test where 1 in 500 of the triples were removed randomly 349

6.8 The CPU time and Java heap size usage estimates for the algorithms 350

6.9 The work of Oraskari [2015] in the Hevner et al. [2004] design science research framework 357

7.1 The work of Oraskari et al. [2018a] in the Hevner et al. [2004] design science research framework 384

7.2 The models that were used in the IPFS publication test. 394

7.3 The IfcOpenHouse data set 396

7.4 IPFS: Sofware process case measurement 396

7.5 IPFS: Local case measurement 397

7.6 IPFS: Network case measurement 398

7.7 The work of Oraskari et al. [2018a] in the Hevner et al. [2004] design science research framework 409

7.8 The models used in the IPFS publication test. 429

7.9 The element level download times of the IfcOpenHouse test model. The results are based on 7600 entity downloads per a case. 433

8.1 Summary of the Design Science Method in the work 1/3. 451

8.2 Summary of the Design Science Method in the work 2/3. 452

8.3 Summary of the Design Science Method in the work 3/3. 453

Abstract

The construction industry has embraced digital technologies more slowly than other industries, and fragmentedly. The coordinated exchange and synchronization of information across company boundaries inherently involve communication across heterogeneous software environments, which causes interoperability challenges. Open standards, mainly Linked Building Data, are a potential means of overcoming these challenges. To tackle the sheer volume of data, I have studied various aspects of Linked Building Data management, including supervision of the data, provision of information, managing versioning, data availability, confidentiality, and integrity. Furthermore, considerations of how this can fulfil the ISO 19650 vision of the most mature building information framework have been regarded.

Überblick

Das Baugewerbe hat die digitalen Technologien langsamer als andere Branchen und nur bruchstückhaft angenommen. Der koordinierte Austausch und die Synchronisierung von Informationen über Unternehmensgrenzen hinweg beinhaltet zwangsläufig die Kommunikation über heterogene Softwareumgebungen hinweg, was zu Problemen bei der Interoperabilität führt. Offene Standards, insbesondere Linked Building Data, sind ein mögliches Mittel zur Überwindung dieser Probleme. Um die schiere Menge an Daten zu bewältigen, habe ich verschiedene Aspekte des Managements von Linked Building Data untersucht: Überwachung der Daten, Bereitstellung von Informationen, Verwaltung der Versionierung, Datenverfügbarkeit, Vertraulichkeit und Integrität. Darüber hinaus wurden Überlegungen angestellt, wie dies die Vision der ISO 19650 von einem ausgereiften Rahmenwerk für Gebäudedaten erfüllen kann.

Acknowledgements

My PhD work has been a long process. I thank Heikki Saikkonen, who has supported my studies, D.Sc. Seppo Törmä for long-time guidance, D.Sc. Vincent Kuo, D.Sc. Mehmet Yalcinkaya, D.Sc. Vishal Singh, D.Sc. Mathias Bonduel, Pieter Pauwels, Oliver Schulz, Jeroen Werbrouck, Madhumitha Senthilvel, G. F. Schneider, D.Sc. Mads H. Rasmussen, D.Sc. Pierre Bourreau, D.Sc. Michel Böhms for good and fruitful cooperation, and Prof. Dr Sigrid Brell-Cokcan. I also would like to say special thanks to Professor D.Sc. Jakob Beetz who has supervised my thesis. Also thanks to my friends and family for encouraging me when it was needed.

This research was carried out at Aalto University in DRUMBEAT Web-Enabled Construction Lifecycle (2014-2017) funded by Tekes, Aalto University, and the participating companies and at RWTH Aachen (2020-2023). It was also funded by and the Aalto-yliopiston tekniikan tukisäätiö (2017), and the EU through the H2020 project BIM4REN grant (grant no. 820773).

Chapter 1

Introduction

The construction industry has been permeated by digital technologies, although it is still fragmented and not fully integrated [WEF, 2016]. The designing of buildings is increasingly carried out using Building Information Modeling (BIM)[Eastman et al., 2011] tools in different BIM models that specify different aspects of a facility: architecture, structures, and technical systems. Building Information Modelling (BIM) involves creating and using a digital model of a building throughout its entire life cycle, from the initial design phase to construction, operation and eventual demolition. [Borrmann et al., 2018b].

Building Information Modelling (BIM) is the foundation for digital transformation in the Architecture, Engineering, Construction and Operations (AECO) sector. On a large scale, the vision of Big BIM (see Section 1.1) is to streamline processes throughout the construction supply chain. By its very nature, cross-enterprise collaboration involves communication across heterogeneous software environments, resulting in interoperability gaps between information systems. These gaps can lead to inefficiencies, inconsistencies in knowledge bases and errors when data has to be re-entered into systems. [Pauwels and Petrova, 2020, Borrmann et al., 2018b]

The heterogeneity arises from the different data needs and functionality requirements of companies in different

sectors, sizes, software and skills. As a result, no one-size-fits-all solution can provide the best value for money for all collaborating parties. In addition, to avoid vendor lock-in, where companies have no choice but to buy solutions and updates from a single software vendor, the overall system should have room for heterogeneity.

In addition, although digital tools are becoming more common in all phases of the building's lifecycle, the continuous use of digital information throughout the entire process is not as widespread as in other industries. Often, valuable information is lost because it is still primarily shared as physical drawings on paper or in a digital format use scenario. These breaks in the flow of information occur at every stage of a building's life, including design, construction, operation and, most importantly, when information is transferred from one phase to another. [Borrmann et al., 2018b]

In addition, in larger projects, the group of allied parties varies for each project, causing cross-organisation cooperation to occur in different data environments. This can lead to the use of a variety of tools. Gaining expertise across the range of tools is challenging and can lead to inefficiencies in training users to excel with the new software sets. Even here, gaps between the systems limit the reuse of knowledge. When data is copied from one project to another, duplicate information entries may be re-entered, making it challenging to keep the data up-to-date and error-free.

Standardization is needed to bridge the gaps and allow different parties to communicate and use the software of their choice without being tied to a single vendor. [Pauwels, 2014a, Pauwels and Petrova, 2020]

On the other hand, poorly used information or expensive mistakes can be avoided with proper data handling. One possible approach is to separate the data from the application logic to make it equally accessible to all tools, regardless of their vendor. This approach also requires common standards but has the advantage of avoiding duplication of information. It is in line with the core idea

behind the Linked Building Data (see Section 1.4). I also discuss the topic in Section 4.5.

Open standards, available to all communication parties, are one way of overcoming interoperability challenges and facilitating communication between parties using different systems. buildingSMART is an international not-for-profit association dedicated to improving the exchange of information between software applications used in the AEC industry. It actively promotes open standards through its OpenBIM initiative.

The vendor-neutral buildingSMART Industry Foundation Classes (IFC) standard [ISO, 2013] and BIM Collaboration Formats (BCF) [Dangl et al., 2017, Paasiala et al., 2019] are good examples. I will describe Open BIM and OpenBIM in more detail in Section 1.2.

In addition to the data generated and used during the life of a building, there is a growing amount of external data that is of value to the asset owner. This includes urban plans, city models, infrastructure models, product data, utility markets, and weather conditions. Although this data is not directly related to the building itself, it can still help make decisions and optimise the performance of the asset.

Given the sheer volume of data, managing the knowledge it contains is essential. Data management connects my work to the general concept of digital twin, which encompasses three primary functions: model, simulate, and manage. In my work, I have chosen to study how data management can be done using open standards, specifically Linked Building Data.

Linked Building Data (LBD) is a semantic knowledge base for sharing consistent, structured building-related information. I have identified some aspects that reflect the definition of knowledge base management: provision of information, supervision of the work and data content, data governance, versioning, provenance and availability, confidentiality and integrity. For more about that, see Section 1.5.

In addition, I chose to study how the management of linked building data could be realised in the envisioned future frameworks of BIM collaboration, where systems are data-driven, data is accessed in a granular manner, database is used and allow data to be queried. I have used the ISO 19650 [International Organization for Standardization, 2018a] standard maturity model as a reference framework. I am particularly interested in how Linked Data can provide a consistent source of information for the users of the organisation while allowing for decentralised systems. The focus is on the information and technology layers of ISO 19650. I concentrate on this topic in Sections 4.4 and 4.5.

In most of my published research, I have followed Hevner's guidelines for design science research: (1) design an innovative and purposeful artefact, (2) have utility for a specific problem, (3) evaluate thoroughly, (4) solve an unsolved problem or a known problem more effectively or efficiently, (5) the artefact has to be "rigorously defined, formally represented, coherent, and internally consistent", (6) the process incorporates a search of a solution over a problem space, and (7) the results have to be appropriately communicated [Hevner et al., 2004].

The Design Science Research (DSR) method is well suited as a research method for this study, as it aims to reduce the gap between theory and practice (see [Aken, 2004]). It also aims to enable the development to solve problems to improve existing systems and enhance the performance of tasks in organizations. They are essential when we aspire to find ways to materialize the Management of Linked Building Data. DSR is a research method that uses generic problem-solving methods to design, construct, and evaluate artefacts to solve a particular problem [March and Smith, 1995, March and Storey, 2008].

To address the overarching research problems introduced above and related to Linked Data Management, I follow the method inspired by Bernstein's and Noy's work [Bernstein and Noy, 2014]. Initially, I introduce research questions that encompass the identified issues. Subsequently,

within each chapter, I present hypotheses that cover these questions, present my related work, and provide supporting and opposing perspectives for the presented assertions.

I have selected two main research questions to cover the data management aspects.

1. How to publish linked building information to help find and access the right information, including versioning?
   Of the knowledge management factors, I here deal with how to provide building information as LBD and how it can be supervised by filtering, checking, and querying.
   I have divided this into three parts
   (a) How to publish, filter, and facilitate finding information in Linked Building Data? (addressed in Section 2)
   (b) How to publish a Linked Building Data ontology? (addressed in Section 3)
   (c) How to publish federated building data as a Common Data Environment? (addressed in Section 4)
2. How information security aspects of data management can be handled?
   The information security factors of data management, which are confidentiality, integrity, and availability, are covered as follows:
   (a) Confidentiality is addressed in Chapter 5.
   (b) Integrity is addressed in Chapter 6.
   (c) Availability is addressed in Chapter 7.

Figure 1.2 [Commons, 2008] shows traditional paper-based handover of the construction documentation. I use a series of use cases to illustrate how the chaos can be handled using Linked Building Data.

I start with scenarios related to publishing LBD, then use scenarios related to problem handling in facilities management, and finally, information security aspects are dealt with in scenarios. They are summarized in Figure 1.1 and listed below.

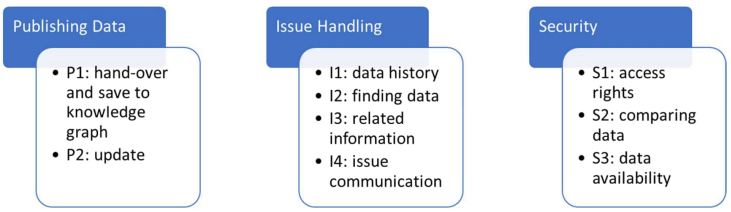


Fig. 1.1. Summary of use cases in the work.

The use cases related to the publication of the Linked Building Data

P1 A handover is made.
   (a) An Architectural office hands a Facility manager the architectural model of the building.
   (b) He adds that to the facility manager knowledge graph.

   This is addressed in Sections 2.1, 4.2, 4.3, 4.4 and 4.5.

P2 A renovation work is made. The new updated as-built model is given to the facility manager and added to the knowledge base. This is addressed in Section 2.2.

Fig. 1.2. Traditional construction handover documentation (Photo by Cornhenge in Wikimedia Commons).

The use cases related to issue handling

- I1 The facility manager uses the issue history of the air terminal to check past actions. This is addressed in Section 3.2.
- I2 He also searches the knowledge-base issue comments about any air terminals. This is addressed in Section 2.4.
- I3 The facility manager browsers information about an air terminal. The system provides all relevant information (Sections 2.3, 4.4 and 4.5.).
- I4 A fix is made. In the work, BCF is used for communication in facility management. This is addressed in Section 3.1.

The use cases related to information security

- S1 A new air terminal has been installed. The facility manager wants to add the associated database element a link to the external product data. Access rights are needed. This is addressed in Chapter 5.
- S2 The facility manager expects that the problem in the air terminal is related to the last renovation made. He wants to compare the original and the new as-built models. This is addressed in Chapter 6.
- S3 Facility Manager office wins management services deal of a building, but the architectural office that designed it is out of business. Luckily, a governmental office has automatically archived the models and can provide an address to the peer-to-peer network for download. This is addressed in Section 7.1.
- S4 A facility manager visiting a building to check an office's roof window. There is very bad network coverage. The intelligent system fetches the needed piece of information from the office users' computers nearby and shows the needed knowledge to her. This is addressed in Section 7.2.

Figure 1.3 shows how my published works connect and how they map to the overall structure of the work.

To answer the question of what Linked Building Data is, I have divided it into three sub-questions:

- What is building data? In Section 1.1, I describe the concept of a building information model and how it has been understood. I extend this in Section 1.2 by describing the concept of Open BIM.
- What is the definition of Linked Data? This is introduced in Section 1.3.
- How can building data be represented as Linked Data? Linked Building Data is presented in Section 1.4

1.1 The Building Information model (BIM)

The concept of Building Information Modeling (BIM) is considered to have its roots in the articles published in the 1970s [Charles et al., 1974, Eastman, 1975]. He proposed the use of virtual building models instead of drawings, databases data management capabilities, ways of visualizing data, and quantitative analysis. At that time, it all was called Building Description System (BDS). The term Building Information Modelling (BIM) was first mentioned around the turn of the century. Subsequently, the term was established. [Borrmann et al., 2018b, Pauwels and Petrova, 2020]

In the AEC sector, the development of digital solutions has traditionally focused on individual tasks or functions, with each task typically performed by a dedicated person or group. These task-specific tools are attractive because the requirements are well-defined, and the value to those performing the task is apparent. However, this focus on individual tasks has resulted in data fragmentation and disconnected systems, each with its way of modelling and managing data [WEF, 2016]. This can cause long waiting times, data reentry, human interpretation, and conversions between different data formats. Making systems interoperable so that they can communicate

automatically could improve productivity and make the flow of information more efficient.

Since the end of the 2010s, there has been a renewed focus on the roots of the concept in information management, data exchange methods, and BIM-based workflows. [Pauwels and Petrova, 2020]

Initially, it was assumed that BIM would contain all possible information centrally throughout the lifecycle of a building, from design through construction to facility management. This assumption is incorrect for both technical and legal reasons. A model is always an abstraction of reality created for a specific purpose. Models of physical reality vary according to perspective and use. [Granholm, 2022]

It is also envisaged that there will be a shift from the production of data to its consumption. Instead of document-based systems, where the information producer decides how to use the information, data models allow users to decide how to use the data. Furthermore, whereas in document-based systems, people have to interpret documents, knowledge bases can be used in programmes as is and enable new innovative ways of working. [Granholm, 2022]

The Little BIM and Big BIM are connected to the maturity of the BIM use in the companies. [Pauwels and Petrova, 2020, Borrmann et al., 2018b]

Little BIM refers to BIM processes where BIM is used internally within an organisation. This is the first step for companies when adopting BIM. It provides a platform for specialists to work together on the same project and associated models. Little BIM provides the tools needed to bring a project to completion. It also provides means of sharing information with external parties, albeit with file exports, such as schedules, plans and PDF documents.

Big BIM has a broader scope: the entire production chain of a project, not just a company or organisation. Transparency and collaboration are key values. In a

heterogeneous environment, Big BIM is often openBIM. Big BIM often involves questions such as how to enable data exchange, and manage the accessibility and user rights of the platform. [Pauwels and Petrova, 2020, Borrmann et al., 2018b]

1.2 The Open BIM

Copyrights cover programs and standards, even if they are in the public domain or free. In this respect, open items are like any other work subject to copyright, within the limits of originality, and are no different from commercial products. Permission to use them is always required from the author, but it is sufficient to comply with the conditions set by the owner in a licence.

While *open BIM* is a generic term referring to the use of BIM in the spirit of open source or open data, openBIM [Petrie, 2020] is a trademark registered by the buildingSMART standards community in 2011. The core idea behind openBIM is the use of open and vendor-neutral standards.

As Petrie [2020] and Pauwels and Petrova [2020] describe, buildingSMART defines openBIM as: something that enables the participation of construction project partners regardless of the software tool chosen, provides a common interlingua for data models, enables the use of the data model throughout the asset lifecycle, and eliminates the need to enter the same data into different systems repeatedly. In contrast, the *closed BIM* model uses proprietary data formats, and non-transparent data flows.

Early as the late 1980s, researchers began investigating ways to improve data exchange in the building and construction sector. When the International Alliance for Interoperability (IAI) was formed around the Industry Foundation Classes (IFC) in 1995, its main objective was to improve interoperability and, thus, productivity in the construction industry. IAI published the first release of

IFC in 1997. The IFC, now defined by ISO/PAS 16739 [The International Organization for Standardization, 2018], is the leading non-proprietary format for exchanging building information in the AECO sector. It is designed to capture building information throughout the building lifecycle and to enable data transfer between BIM software. The open denition of the data structure has become a standard in public sector initiatives. However, there are still limitations. Data exchanges between applications are mostly one-way, and the interoperability of BIM across different lifecycle stages and functionalities is limited due to incomplete, inconsistent or ambiguous use of IFC attributes, names or content. Recent developments have focused on Specifiers' Properties Information Exchange, and Linked Data. Most of the created tools are still used for academic use only. [Borrmann et al., 2018a, Volk et al., 2014]

The first known mentions of "OpenBIM" date back to the turn of the millennium (1999) and, in practice, refer to the IFC standard.

The International Alliance for Interoperability, now rebranded as buidingSMART, has published vendor-neutral formats that include, in particular, the Industry Foundation Classes (IFC), BIM Collaboration Format (BCF), buildingSMART Data Dictionary (bSDD), COBie, CityGML, and gbXML, with the aim of interoperability, flexibility, and sustainability. [Petrie, 2020]

openBIM also covers Semantic Web standards that span the construction industry. For example, ifcOWL is one of the standard formats [Grutters, 2022] for IFC published by buildingSMART. It refers to a set of W3C Web Ontology Language (OWL) specifications that describe IFC entities, their associated attributes, and relationships so that the IFC data model can be represented in the Resource Description Framework (RDF), the W3C standardized language for the semantic web. ifcOWL is a relatively direct mapping from the EXPRESS data description of the IFC/STEP format.

Since 2013, an increasing number of studies have been

published in the academic community on open standards for building information.

1.3 Linked Data

Linked Data is a method of publishing and sharing structured data that allows it to be linked to other data on the web. It is based on principles and technologies, such as the Resource Description Framework (RDF) and Uniform Resource Identifiers (URIs), that allow information to be presented in an interoperable way, linked to different sources, and accessed consistently.

The main goal of Linked Data is to create a decentralised and distributed network of data that can be easily shared, linked and searched across applications and domains. This can help organisations and individuals communicate, publish and reuse data productively.

The concept of Semantic Web is based on the definition given by Tim Berners-Lee in 2006: "*The Semantic Web isnt just about putting data on the web. It is about making links, so that a person or machine can explore the web of data. With linked data, when you have some of it, you can find other, related, data.*"

The Web of Data refers to the vision of a global network of interconnected datasets where a piece of data is linked meaningfully. Tim Berners-Lee has listed a 5-star definition of good open Web of data.

- * Data is available on the web in any format under an open license.
- ** The data is provided in a machine-readable, fixed and organised format.
- *** The data is in a non-proprietary format.
- **** Use W3C open standards, in particular, RDF and SPARQL, for data identification.
- ***** All of the above, but the data is also linked to data produced by others to provide context for the data.

Although Web of Data and Linked Data are closely related concepts and often used interchangeably, they have distinct meanings. Linked Data is a specific technical methodology for achieving the Web of Data vision.

He listed four design principles for Linked Data [Berners-Lee, 2006]. These are:

1. Uniform Resource Identifier (URI) links are used to uniquely name things.
2. Like web pages, the HTTP protocol is used to retrieve facts about these named things.
3. When facts about a named thing is retrieved, the associated information is provided.
4. In addition to the returned information, follow-up links are provided to retrieve more information.

Then, Hyvönen et al. point out in [Hyvönen et al., 2014] that it would be beneficiary to extend the 5-star model with one star granted when the schemas used in the dataset are publicly available on the Web, and one, if the quality of the dataset is marked.

Jorulv Rangnes also pointed out in his vision of interoperability in [Rangnes, 2020] the general assumption when interlinking data in the LBD that the Linked Data is meant to cross domains.

1.4 Linked Building Data

Linked Building Data (LBD) is a term used to describe a model for representing and storing information about buildings and the relationships between their various elements in a way that enables easy access and analysis. This is done using the Resource Description Framework (RDF). Linked Building Data aims to provide a linked and comprehensive view of the data associated with buildings, such as information about their design, construction, operation and performance. The knowledge base can be used for various purposes, such as supporting research

and innovation, improving building design, optimising energy consumption and planning renovation.

The proliferation of fragmented, heterogeneous and distributed data has created a need for information flow solutions that enable efficient data-sharing between stakeholders. Traditional file-sharing or centralised repositories only partially address the challenges of information sharing in decentralised organisations typical of the construction industry, especially given many different stakeholders over a long period of the construction lifecycle.

Linked Building Data [Pauwels et al., 2015, Pauwels, 2014b] is an alternative that addresses this need in a standards-based way without a central intermediary. It is a decentralised approach to information sharing: Building data is published on the web using Web of Data technologies [Bizer et al., 2008, 2009]. At a concrete level, the data is transformed into a graph based on the Resource Description Format (RDF) [Lassila et al., 1998, Beckett and McBride, 2004], which is then published on the web. Each building object in the graph has a URI [Berners-Lee et al., 1994] as its globally unique identifier; using a URI, the object's description can be retrieved over the web as structured data (serialized, for example, in JSON [Bray, 2013] or XML [Sperberg-McQueen et al., 2008]). The names of object types and properties used in structural descriptions are defined in web ontologies, controlled vocabularies published on the web that specify what objects exist in the domain and how they relate to each other. A foundational ontology in the building information domain is ifcOWL [Beetz et al., 2009, Pauwels and Terkaj, 2016, Pauwels and van Berlo, 2019], the ontology version of the IFC specification.

A retrieved description of an object can include links to related objects that provide more information about these, continued in a chain of actions. Like on the human-readable web, these links can lead to other objects within the same graph or objects on other websites. Even if a user is only interested in one object at first, he can follow the links to gradually retrieve all the accessible information

associated with the element and various linked external data sources. However, the idea behind Linked Building Data is that most of them are seldom deeded, especially during the later stages of a construction project. Accessing just a tiny portion of the data is enough in many situations.

Linked Building Data has several characteristics useful in the construction domain. Building objects are published online, which makes them easily accessible (if there is a proper network connection), all parties will access the same latest version of data, and objects can be accessed in a granular manner so that users and their tools are not overwhelmed with large models and objects can be linked to external data using their addresses to enrich and interconnect them. Data can be managed in a decentralized manner, respecting the digital sovereignty of each party.

Linked Building Data (LBD) presents building data as linked data. As the name suggests, LBD includes providing building data according to the four principles set out by Tim Berners-Lee in [Berners-Lee, 2006].

Since the models describe different aspects of the same buildings, the datasets produced by different parties are strongly interrelated. Unfortunately, the relations are seldom represented explicitly in the data. Instead, the borders between organizations and disciplines, divergent conventions and cultures, and a general lack of collaboration lead to unnecessarily disconnected and fragmented information, a problem that siloed systems and formats have traditionally exacerbated.

Linked Building Data [Pauwels, 2014b, Törmä et al., 2012, Torma, 2013] is a set of technologies designed to solve these problems. It is also called Web of Building Data [Törmä, 2014]. It allows the building data to be published in a distributed manner, with a unique URI for each building object. The URIs allow for the creation of links between objects, possibly across data sets that reside on different hosts. The data can be accessed according to Linked Data principles [Bizer et al., 2008] in a granular way, one object at a time and the description returned can

include links to other objects. The advantages are that the data is online and up to date and can be shared in flexible fragments rather than as huge files of full BIM models as is currently the case.

The ultimate goal of Linked Building Data is to create a global, decentralized network of connected building objects to make all relevant data easily accessible. Linked Building Data was created to increase openness to improve the quality and use of data, increase efficiency and coordination of activities, and encourage innovation and collaboration. However, many of the technologies used in this system would be even more widely used if there was a way to control access to the data. The importance of user management and access control is emphasized by the focus that other solutions, such as centralized project repositories, place on these features.

In addition, the World Wide Web Consortium (W3C) Linked Building Data Community Group[1] has created and maintained the following ontologies: the Building Topology Ontology (BOT), the Product Ontology (PRODUCT), the Properties Ontology (PROPS), and the Geometry Ontology (GEOM). When referring to Linked Building Data, reference is usually made to these ontologies. These ontologies are open specifications, the idea of which is to describe some essential information about IFC datasets.

1.5 Linked Data Management

Data management is the process of organizing, storing, and maintaining the integrity, security, and accessibility of data. There are several vital points to consider when it comes to data management.

Linked Building Data can be seen as a knowledge graph. Therefore, Linked Building Data Management can be seen as information management which is

[1] https://www.w3.org/community/lbd/

defined to mean "the process of collecting, organizing, storing, and providing information within a company or organization" by Cambridge Dictionary [2022], and management in general, the act of managing, and in this context, supervising (Merriam-Webster [2022]). On the other hand, Gartner defines in[2] Data Management as methods to provide consistent access to data and meet the consumption requirements of the business processes.

Data management has many aspects. From the data usage standpoint, finding the required information when needed is essential. It consists of the ability to find the data source and filter the facts of existing data collection to offer focused information for the application.

To find the relevant information efficient query mechanisms are needed as well as filtered views of the data models to support a specific application purpose. For the queries, the accessibility of the data is crucial. That is, the data needs to be in a format/syntax that is easy to query and the used ontologies should support the use cases. This is handled in Chapter 2.

The ISO 19650 concept of Common Data Environment (CDE) [International Organization for Standardization, 2018a] covers the aspect of discovery as it provides a single source of structured information for the cooperating parties. It also defines how to manage information consistently. I cover this in Chapter 4.1.1 and present our work on how Linked Building Data(LBD) can be published as CDE. Also, it is presented how the requirements of ISO 19650 Stage 3 can be fulfilled in the proposed framework.

Moreover, mechanisms to create partial models that are specified for a certain application purpose are beneficial. In this aspect, the standard Model View Definition (MVD) of buildingSMART and the data need specifications for a specific collaboration purpose are helpful. In this work, I concentrate on the technical aspects and study to find answers to the question of how using Linked Building Data can facilitate data management and how Linked Building

[2] https://www.gartner.com/en/information-technology/glossary/dmi-data-management-and-integration, Accessed Jun 30,2022

Data users can integrate this into the existing Linked Data framework.

I have selected some aspects that reflect the definition of t the management of Linked Building Data. First, to be able to supervise the knowledge base, the data should be easily queries, it needs to be easily deduced, what has been changed between versions of a model, and it should be easy to validate that the data fulfils information requirements that are set to the project, and the last part from the Gartner definition: the availability, confidentiality, and integrity of the data.

In the Linked Building Data standpoint, the aspects concerning data management can be organized as follows:

1. Data quality: Ensuring the data used in an LBD data model is accurate, complete for the application purpose, and up-to-date is critical to the model's success.
2. Data organisation: Data should be constructed to make it easy to find and access. This might involve using labels and tags, creating a hierarchical structure, or implementing a search functionality.
3. Data security: A linked Building Data model may contain sensitive information about the building and its assets, so appropriate security measures to protect this data are needed. Data should be protected from tampering or unauthorised access. The measures include encryption, authentication, and access control. Data availability: Data should be accessible to legitimate users who need it and when they need it.
4. Data storage: AECO tools can generate large amounts of data. It is vital to have a robust and scalable storage solution to preserve the data. Data should be stored in a secure and persistent location. Essential factors to consider are backup and scalability, recovery, and data protection.
5. Data integration: LBD data may interlink data from multiple sources, such as design and construction models, documents, as-built data, and facility management systems. Managing consistency is

needed. Data should be up-to-date, accurate, and consistent, achieved through regular checks and updates, and by implementing validation processes for verifying correctness.

6. Data governance: Establishing clear policies and procedures for managing BIM data can ensure that all stakeholders have access to the information they need while protecting the data's security and integrity.

Confidentiality, Integrity, and Availability have been considered the fundamental terms of security controls in information systems. The triad has been used as a definition of information security and, later, information network security since the early days of cybersecurity. It is based on the theoretical work by Ware, Anderson, Saltzer and Schroeder in the 1970s. As a well-known classification, I use them here to address the information security management themes. Each of them is handled in a separate section. [Ham, 2021, Samonas and Coss, 2014]

Confidentiality is handled in Section 5, integrity in Section 6, and availability in Section 7.

Figure 1.4 shows an overview of how my dissertation research work connects to the Linked Building Data Management concepts listed above.

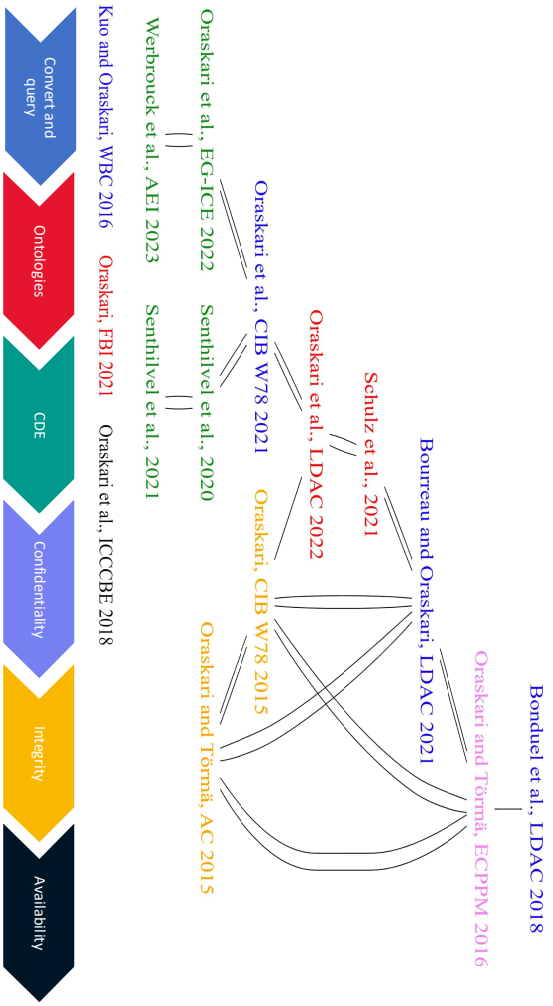


Fig. 1.3. Semantic connection between published articles and the overall structure of my dissertation work.

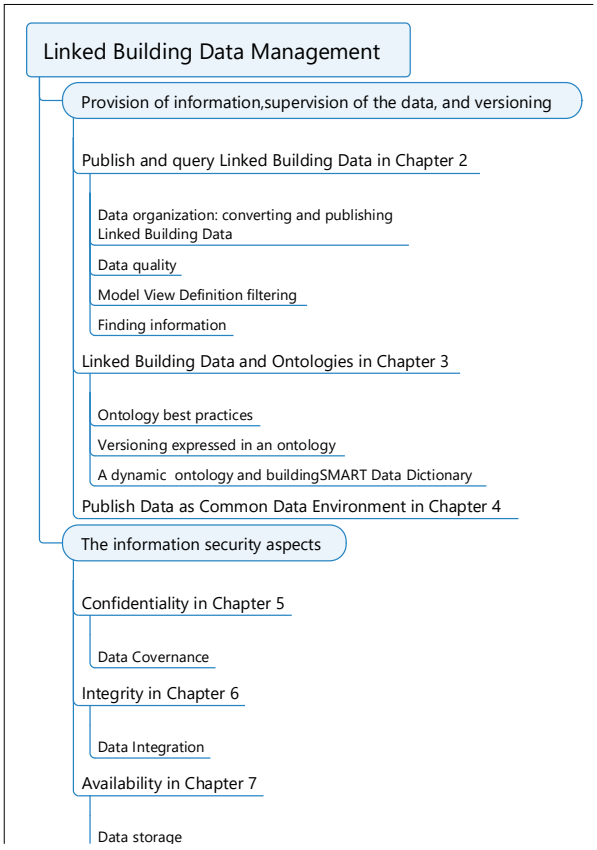


Fig. 1.4. An overview of my published dissertation research work.

Chapter 2

Conversion and findability of Linked Building Data

In this chapter, I address the following research question: How can linked building information be published, and what factors improve the queryability and accessibility of the information? By publication, I mean how to prepare data to be presented as Linked Building Data.

I have chosen the following hypotheses, which express the focal points that I have explored in more detail.

Hypothesis 2.1 *When building data is transformed into a semantic web using ontologies from the World Wide Web Consortium (W3C) Linked Building Data Community Group, the generated graph assertions make the information more accessible to developers and data analysts, and are easier to use compared to the Industry Foundation Classes (IFC) standards expressed as Linked Data (ifcOWL [Beetz et al., 2009, Pauwels and Terkaj, 2016, Grutters, 2022]), making data management easier.*

Hypothesis 2.1 is covered in Section 2.1.

Developers and data analysts have been selected as a target group, since the semantic web is a technical concept that

can be compared with relational databases and so, are hidden to the end users of the applications. Therefore, the main aim when making data structures easier is to facilitate creating new applications, and enable new innovations.

Hypothesis 2.2 *Rule-based conversion is possible and is better than algorithm-based conversion at keeping models up to date after changes in the model. However, algorithm-based conversion does a better job of separating the resulting models from the input and can take non-RDF data as input. This is handled in Section 2.2.*

Hypothesis 2.3 *Ways to implement quality management for the RDF conversion process in the LBD context are possible and beneficial. Unit tests can be written to validate that the RDF converter in the LBD context is doing what the developer of the software or people behind the alignment specification of the ontologies think it should do.*

The paper presented in Section 2.3 addresses Hypothesis 2.3.

Hypothesis 2.4 *It is possible to implement the buildingSMART Model View Definitions (MVD) filtering for LBD models. The paper presented in Section 2.3 addresses this.*

Hypothesis 2.5 *The open BIM issue data presented as buildingSMART BIM Collaboration Format (BCF) can be made more accessible compared to the standard BCF XML or BCF API approach and thus promotes data management.*

Hypothesis 2.5 is covered in Section 2.4.

In this Chapter, I will introduce what support I have for the claims and discuss arguments that are against them. In Section 2.5, I will summarize findings presented in the chapter.

General discovery of the data is partly handled in the Linked Data Principles (see Section 1.3), but is also part of

the work about implementing a search engine presented in Section 2.4. Naming conventions are out of the scope of my work, as they are handled in ISO 19650 for files, in [International Organization for Standardization, 2018a] and [Chen et al., 2017] for BIM objects.

Figure 2.1 summarises how the research questions, hypotheses and their handling are connected. Then, Figure 2.2 shows my articles of the chapter at a glance.

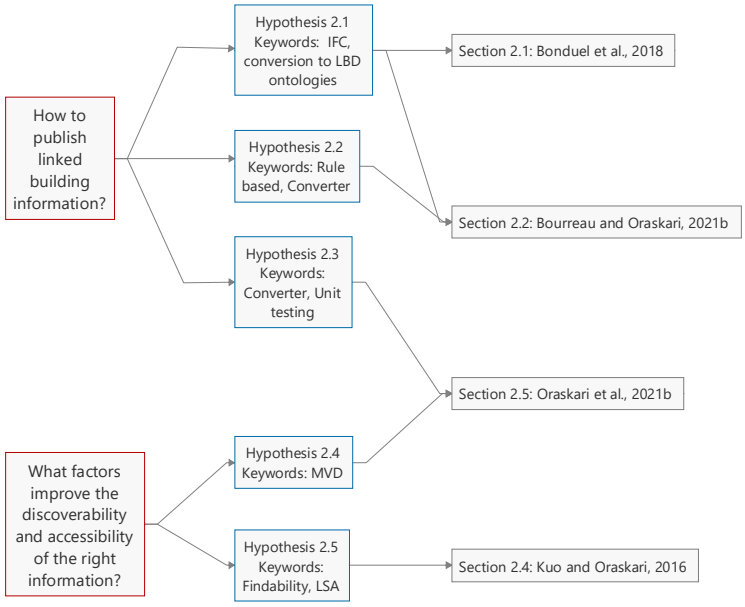


Fig. 2.1. How the research questions, hypotheses, and their handling in articles and the sub-sections are connected.

Publication	Objectives	Method	Key findings
Bonduel et al., The IFC to linked building data converter-current status., 2018	• method to convert IFC models into Linked Building Data	The Design Science Research Method: Analytical Static Analysis	•The proposed IFCtoLBD converter facilitates the conversion of IFC building models to modular LBD Abox graphs. •The resulting Abox graphs are more user-friendly than ifcOWL Abox graphs
Bourreau & Oraskari, Dynamic BIM model conversion as inference-based ontology alignment., 2021	• a rule-based method to convert IFC to W3C Linked Building Data ontologies	The Design Science Research Method: Functional (Black Box) Testing and Experimental Controlled Experiment	• The rule-based approach is feasible • An algorithmic approach can cover Linked Building Data more comprehensively. • The presented method offers a data-driven solution where rules are separated from the programming code.
Oraskari et al. , SHACL is for LBD what mvdXML is for IFC., 2021	• How to define a sound and valid conversion output? • What are the characteristics of a good set of unit test for a translator tool? • How to implement an MVD filtering for LBD?	The Design Science Research Method: A descriptive Scenario	• We presented several unit test methods to improve Linked Building Data conversion quality. Characteristics were described. • Model views can be created for LBD data using SHACL. • The presented method limits in Ontology for Property Managements level 1.

Fig. 2.2. My articles related to publishing and facilitating queries at a glance (For the term Abox, see site https://www.lesliesikos.com/knowledge-base/)

2.1 The IFC to Linked Building Data converter-current status (LDAC 2018)

Consider a scenario where digital engineering documents, such as architectural plans, electrical and mechanical

schematics, and structural designs, are handed to a facilities manager. The data are converted and provided as Linked Data so that all the information can be aggregated into a coherent knowledge base that can be kept up to date as changes are made.

This paper describes the basic design principles and benefits of the IFCtoLBD converter presented by Oraskari et al. in [Oraskari et al., 2021a]. It addresses the hypothesis 2.1 presented at the beginning of this chapter.

Hypothesis 2.1 claimed that the conversion of BIM data from an ifcOWL presentation into LBD ontologies facilitates the use of the information and is therefore a data management enabler. This relates to the monitoring aspects of data management (see Section 1.5). It is shown that the queries required to retrieve the same information from the new resulting LBD data representation are simpler and less complex than the queries of the corresponding ifcOWL assertion models. The graph structure is simplified and the SPARQL queries is significantly shorter. I argue that this helps to find the needed information, and since monitoring is one of the data management aspects, the queries also help to manage the data. However, this comes at the cost of reducing the information content of the model.

The importance of this work to the LBD community was to provide a simple means of converting IFC-SPF formatted (IFC STEP Physical File)[Pratt, 2001] BIM data into Linked Building Data. This paper demonstrates the conversion principles of how an RDF assertion data model can be converted into a new data model using a selected set of aligned ontologies and a simple graph step method. We also showed how highly abstract graph path descriptions could be used to create a mapping to corresponding triples in the new data model. Since then, the method has been further developed by introducing direct element mappings alongside the step.

The IFCtoLBD converter writes the output of the conversion to a file. Only when the file is published online, and the base URI [Berners-Lee et al., 2005] is

chosen to match the internet address of the server, does the result fully comply with the Linked Data principles listed in Section 1.3. It uses URIs(URLs) so that you can use HTTP to retrieve information about the data elements, and the links can be followed to get more information.

In the usage scenario, the proposed IFCtoLBD tool can be used to convert IFC/STEP formatted files into Linked Data. As shown in the BIM4Ren project[1] (the source code is in IFCtoB4R-DM_OpenAPI[2]) repository, the generated model can also be automatically uploaded into an RDF repository[3] that can serve as a knowledge base.

Table 2.1 summarises the use of design science research methods in this study.

My contribution to the paper: In the research work, my contribution was twofold. First, I designed the graph traversal algorithm and the core principle of how the conversion is made. The idea comes from the programming pattern I have been using in my earlier work (e.g., in the checksum algorithms presented in Section 6.1). The new idea was to combine this with the RDF path patterns and extract a new graph on the fly. In addition, I wrote the Java code implementation. Mathias Bonduel analyzed the ifcOWL ontology for suitable IFC graph patterns and wrote the SPARQL queries. He had a

major work in the analysis of the ontologies.

Table 2.1. The work of Bonduel et al. [Bonduel et al., 2018] in the Hevner et al. [Hevner et al., 2004] design science research framework

Guideline 1: Design as an artifact
This Java Apache Jena implementation was a proof of construction for an IFC to Linked Building Data converter that converts standard IFC/STEP formatted BIM models into assertions of the W3C LBD community[4]-defined Linked Building Data ontologies [5]. The concept was presented at an abstract level so that the work could be reimplemented in any programming language. The first known IFCtoLBD converter and the associated SPARQL queries were presented.

Guideline 2: Problem relevance
Reading and understanding IFC files requires expertise. ifcOWL existed but using it involves the same challenges of complexity as IFC models have. Therefore, the research question was: how to facilitate access to BIM data expressed as IFC and allow people to benefit from the good sides of Linked Data? Linked Data allows data integration from multiple sources, ensuring stakeholders have access to up-to-date information about the building assets. The integration enables effective collaboration among different collaborators. Therefore, the business problem was how to provide users with the information already available as IFC. The constituent user community of the artefact is the researchers of the Linked Building Data Community, but also developers in the AEC sector that want to use BIM data in their software.

[1] https://bim4ren.eu/
[2] https://github.com/jyrkioraskari/IFCtoB4R-DM_OpenAPI
[3] in that case Stardog https://www.stardog.com/
[4] https://www.w3.org/community/lbd/ (Accessed June 13, 2023)
[5] https://github.com/w3c-lbd-cg/ontologies

The solution belongs to the converter and more specifically, the LBD converters problem class.

This work is an enabler for the following BIM use cases: Quantity take-off of, Author Cost Estimate [The Pennsylvania State University, 2013], and Building operation and maintenance [Borrmann et al., 2018b]. Also, Decision-making for residential building renovation when comparing renovation alternatives (buildingSMART use-case GUID:A82E2F30-3752-48EA-A2D7-8C6B7F780733 [buildingSMART International, 2022]). Simple quantity estimates are possible, but specific support for the IFC Quantity Sets was left for future work.

Guideline 3: Design evaluation

Hevner et al. Design Research Evaluation Method: Analytical Static Analysis was used when the complexity and size of the queries to get information from the models were evaluated.

The model sizes were compared in an Experimental Controlled Experiment where the Duplex model from the NIBS collection was converted (see Section 2.1.2).

Guideline 4: Research contributions

While IFC enables limited interoperability between systems, BIM data represented as Linked Data allows the exchange and integration of data with tools in the construction domain and outside of the AEC sector.
There was no such method to convert from IFC data to Linked Building Data ontologies existing yet.
Generalization of the results: The conversion method that follows the depth-first kind of search over the ifcOWL Abox graph and uses RDF path patterns can be generalized to implement other graph searches and conversions.
Also, the notions of the three levels of complexity of the PROPS ontology structure are valid in general (For example, for Ontology for Property Management (OPM)[6] and Ontology for Managing Geometry (OMG) [Wagner et al., 2019]).

Guideline 5: Research rigor

The work was peer-reviewed. Also, the conversion rules, source code, and queries were made available.

Guideline 6: Design as a search process

There are many ways to implement the functionality of the converter. Here we selected the approach to extend our earlier work of IFCtoRDF[7] with the assumption that it would save effort.

Guideline 7: Communication of research

In Proceedings of the 6th Linked Data in Architecture and Construction Workshop (LDAC), University College London (UCL) and ARUP, in London, United Kingdom, 19 - 21 June 2018.

[6] https://w3c-lbd-cg.github.io/opm/ (Accessed June 13, 2023)
[7] https://github.com/pipauwel/IFCtoRDF (Accessed June 13, 2023)

In agreement with the organizers of the Linked Data in Architecture and Construction workshop, the authors of the material preserve all their copyrights and publication rights. The work is originally published in [Bonduel et al., 2018]. The content presented here is based on our final draft of the article.

Mathias Bonduel KU Leuven, Dept. of Civil Engineering, Technology Cluster Construction, Ghent, Belgium, mathias.bonduel@kuleuven.be
Jyrki Oraskari Department of Computer Science, Aalto University, Konemiehentie 2, 02150 Espoo, Finland, Jyrki.Oraskari@aalto.fi
Pieter Pauwels Ghent University, Dept. of Architecture and Urban Planning, Ghent, Belgium
Maarten Vergauwen KU Leuven, Dept. of Civil Engineering, Technology Cluster Construction, Ghent, Belgium
Ralf Klein KU Leuven, Dept. of Civil Engineering, Technology Cluster Construction, Ghent, Belgium

Abstract: Several methods for creating building-related Linked Data graphs exist. This paper focuses on the conversion of IFC Building Information Models (BIM) to RDF Abox graphs using the emerging W3C Linked Building Data (LBD) modular ontologies: BOT (building topology), PRODUCT (classification of building elements) and PROPS (building-related properties). The existing IFC-to-RDF converter tool, converts IFC into ifcOWL-based Abox graphs which are rather complex and difficult to implement in software applications. The IFC-to-LBD converter presented, is necessary to transform IFC building models into RDF Abox graphs structured according the new LBD ontologies. An output graph contains the relevant information of the IFC building model related to building topology, building elements classification and building-related properties. Additionally, the graph structure becomes more concise (minimum 83 % less triples) and significantly easier to query compared to the output of the ifcOWL-based IFC-to-RDF converter.

keywords: Linked Data, IFC, converter, AECO, BIM, Linked Building Data, BOT, PROPS, PRODUCT

2.1.1 Introduction

Linked Data for the Architecture, Engineering, Construction and Operation (AECO) domain is a research topic gaining momentum during the last decade. Besides numerous scientific publications and dedicated conferences, a Linked Building Data Community Group within the W3C (W3C LBD CG) and a Linked Data Working Group (LDWG) as part of the Technical Room of buildingSMART

were founded. Linked Data graphs can be created from scratch, or by converting existing structured data available in other formats than RDF. The presented research focuses on the conversion of existing Building Information Models (BIM) in IFC format to modular Linked Data Abox graphs as proposed by the W3C LBD CG [Terkaj et al., 2017].

BIM and IFC

Building Information Modeling or BIM is an established concept within the AECO domain, regarding the representation and exchange of structured building information. BIM is implemented in object-oriented database systems with a strong focus on 3D modelling. Different software environments apply the BIM concept and a wide variety of both open-source and commercial software applications exist such as Autodesk Revit, Graphisoft ArchiCAD, FreeCAD (with plugins) and Tekla Structures. Each of these software packages uses its own proprietary data schema to represent the structure of a building, the elements it consists of and their properties. If a BIM workflow is used to exchange building information between project stakeholders, a neutral format to exchange the models is necessary. The Industry Foundation Classes (IFC), proposed by buildingSMART International, is a neutral and open ISO standard. The IFC standard consists of an IFC schema (in both EXPRESS and XSD) and an IFC file format (in IFC-SPFF and ifcXML) for BIM data [Pauwels and Terkaj, 2016]. BIM software developers can implement an exporter and/or importer to convert respectively their native BIM format to the neutral IFC format, or the other way around. Many national BIM standards strongly recommend or even impose the use of IFC to improve the exchange of building information between project stakeholders.

Besides the clear advantages regarding BIM software interoperability, IFC also has some limitations as indicated in [Pauwels and Terkaj, 2016]. First of all, the EXPRESS and XSD languages lacks methods for defining formal semantics, making it difficult to apply generic reasoning and querying methods on IFC building models. Secondly, developers can propose extensions for the IFC schema to buildingSMART, but the technology does not allow to extend the IFC schema on-the-fly in an user-friendly way. Thirdly, fine grained linking of building information stored in an IFC file to related data on the web (e.g. regulations of local authorities, geographic information, general knowledge, building product information, etc.) is not possible. Finally, the IFC schema is large - the most recent IFC schema, IFC4 Add2, contains about 1200 classes - and rather complex, making it a challenge to implement correctly in software.

The ifcOWL Ontology and IFC-to-RDF Converter

To overcome the first three limitations imposed by the usage of the IFC standard, researchers investigated the potential of Semantic Web Technologies such as web ontologies, the SPARQL query language and reasoning engines. Before developing tools to convert individual BIM models to RDF Abox graphs, a shared data structure or ontology (Tbox) had to be decided upon by the research community [Beetz et al., 2009]. Over the years, several ontologies for the IFC schema were presented, resulting in a recommended ifcOWL ontology endorsed by buildingSMART [Pauwels and Terkaj, 2016]. The ifcOWL schema was designed to be as equivalent as possible to the original IFC EXPRESS schema, with the idea of backwards compatibility in mind. Following the development of the ifcOWL ontology, an IFC-to-RDF converter tool was developed [8]. The application converts an IFC building model to an RDF Abox graph-structured according to ifcOWL, i.e. sets of RDF triples that contain assertions between individual RDF resources.

BOT-PROPS-PRODUCT Ontologies

The ifcOWL ontology and the IFC-to-RDF converter made it possible to apply Semantic Web technologies such as standardized querying (SPARQL) and reasoning methods, on IFC building models. Having IFC files available as RDF graphs also facilitates fine-grained linking to other RDF datasets. The last limitation of IFC mentioned in Section 2.1.1 however, remains unsolved. The ifcOWL schema was developed as a Linked Data counterpart of the original IFC EXPRESS schema, resulting in one monolithic and complex ifcOWL ontology (Tbox). Converted IFC building models (Abox) are structured according to ifcOWL, so they too are large and complex. As a result, query writing is negatively influenced, making it hard to use the building data in Linked Data applications.

Alternatives, such as IfcWoD (an extension of ifcOWL) [Mendes de Farias et al., 2015] and simpleBIM (a tool to simplify ifcOWL Abox graphs) [Pauwels and Roxin, 2016], were proposed to overcome the above limitation of ifcOWL. Although these alternatives made query writing easier and even proved to reduce query execution times, a more generic approach using modular ontologies and RDF graphs is seen as a necessity for real industrial applications [Terkaj and Pauwels, 2017]. The Building Typology Ontology (BOT) [Rasmussen et al., 2017b] was developed within the W3C LBD CG as a central and modular ontology for the AEC industry, followed by the emergence of subgroups developing a whole range of other modular ontologies for building products, building-related properties, geometry, etc.

[8] https://github.com/jyrkioraskari/IFCtoRDF-Desktop

BOT is developed to allow defining a building's topology, by using dedicated BOT classes (bot:Site, bot:Building, bot:Storey, bot:Space and bot:Element) and specific BOT relations between them (e.g. bot:containsElement). The PRODUCT ontology is designed to classify individual building objects, e.g. as a p4bldg:Wall, while the PROPS ontology is used to assign properties to building-related elements. As the PROPS ontology is still in the conceptual design phase, no ontology (Tbox) is available yet. The current early proposal for the PROPS ontology structure includes three levels of complexity [9], as demonstrated in Listing 1. Grouping (L2 or L3) and versioning of properties (L3) is only possible when using the more complex structures.

Listing 1. Currently proposed levels of complexity in the PROPS ontology, demonstrated here for the case when the actual property value is a literal

```
#PROPS level of complexity 1 (L1)
inst:slab_A props:phaseCreated_simple "New Construction" .

#PROPS level of complexity 2 (L2)
inst:slab_B props:phaseCreated inst:property_B .
inst:property_B schema:value "New Construction" .

#PROPS level of complexity 3 (L3)
inst:slab_C        props:phaseCreated inst:property_C .
inst:property_C seas:evaluation inst:state_C .
inst:state_C a opm:CurrentState ;
        schema:value "New Construction" ;
        prov:generatedAtTime "2018-01-03T13:35:23Z" .
```

IFCtoLBD Converter

Ontologies (Tbox) in the Semantic Web domain are mainly used to allow instantiating them into RDF data instances (Abox). As stated before, such RDF graphs can be made from scratch or by converting other data, such as conventional BIMs, into RDF. It is for example possible to develop a dedicated plugin for existing BIM software environments to convert proprietary BIM data in RDF graphs, as demonstrated in [Rasmussen et al., 2017a] for the BOT ontology and Revit BIM authoring tool. On the other hand, correctly exported IFC files from different proprietary BIM systems can also be converted to RDF graphs. The IFC-to-RDF converter translates IFC to RDF Abox graphs according to the ifcOWL ontology. In this paper, a similar approach is used to convert IFC to modular Linked Building Data (LBD) graphs, aiming initially at the BOT, PROPS and PRODUCT ontologies.

[9] https://github.com/w3c-lbd-cg/lbd/blob/gh-pages/presentations/props/presentation_LBDcall_20180312_final.pdf

Relevant information from IFC building models is extracted and transformed into effective and simple Abox RDF graphs suited for usage in Linked Data applications.

All URI prefixes mentioned in this paper are assembled in Listing 2.

Listing 2. Used URI prefixes in this paper

```
@prefix inst: <https://example.com/> . # demo namespace for node instances
@prefix bot: <https://w3id.org/bot#> .
@prefix props: <https://w3id.org/props#> .
@prefix p4bldg: <https://w3id.org/product/BuildingElements#> .
@prefix schema: <http://schema.org/> .
@prefix seas: <https://w3id.org/seas/#> .
@prefix opm: <https://w3id.org/opm#> .
@prefix prov: <https://www.w3.org/TR/prov-o/#> .
@prefix express: <https://w3id.org/express#> .
# similar for other ifcOWL versions:
@prefix ifcowl: <http://www.buildingsmart-tech.org/ifcOWL/IFC2X3_TC1#>.
```

2.1.2 The Current State of the IFCtoLBD converter

Implementation Details

The IFCtoLBD converter is an open-source project written in Java using the well known Jena framework, and is available on Github [10]. The proposed converter can handle IFC files from all actively used IFC schemas, from IFC2x3 TC1 till IFC4 Add2. An IFC building model can be converted to one RDF file combining all selected LBD Abox modules (BOT, PROPS and/or PRODUCT) or to multiple separate RDF files per selected LBD module (Abox). Using such Abox modules, which can be used independently from each other, it becomes easier to understand the structure of the combined graph.

If the user selects the PROPS module, he or she can choose one of the three PROPS levels of complexity (see Listing 1). If PROPS L2 or L3 is selected, the user can choose to define the in between property instance nodes (L2 and L3) and/or the state instance nodes (L3) as blank nodes or instances with a stable URI. If the above nodes do not have a stable URI, they are not unique outside the database or file they reside in, making it impossible to connect them to other resources outside their original environment. On the other hand, using blank nodes can decrease the file size of the exported RDF and increase the human readability of the Turtle file. As the PROPS ontology is still in a conceptual phase, specific datatype and

[10] https://github.com/jyrkioraskari/IFCtoLBD

object properties are not defined there. For the moment, the converter creates these properties on-the-fly based on the IFC property name. If the property value is a literal and the user selects PROPS L1, the suffix `_simple` is added at the end of the URI of the generated predicate. In this way, there is a distinction between e.g. props:phaseCreated in PROPS L2 and L3 (an owl:ObjectProperty) and props:phaseCreated_simple in PROPS L1 (an owl:DatatypeProperty).

The Conversion Process

The IFCtoLBD converter first uses the existing IFC-to-RDF converter internally to transform the original IFC file to a temporary ifcOWL Abox graph (Figure 2.3). That graph is converted incrementally to one or multiple LBD Abox graphs, according to the settings of the user. By first converting the IFC file to an ifcOWL-based Abox graph, the overall conversion process becomes slightly slower and there is an extra dependency on the ifcOWL ontology. However, by implementing this intermediate step, using the well-tested IFC-to-RDF converter and the reliable ifcOWL ontology, it becomes possible for the converter to query RDF patterns directly. When converting very large IFC building models, a connection with a RDF database system is necessary, but implementing this requires only minor changes to the current application.

When the intermediate ifcOWL Abox graph is available, the converter iteratively creates new LBD nodes starting from the relevant ifcOWL nodes, followed by corresponding LBD edges between these new nodes (Figure 2.3).The converter follows the controlled graph traversal using path templates: it starts from the IfcSite instance node and traverses the ifcOWL-based graph towards its IfcBuilding instance nodes, while simultaneously looking for the IFC properties of the IfcSite instance. The search is not started from the IfcProject instance, since there is no corresponding class in the BOT ontology. In a successive phase, it queries all instances of IfcBuildingStorey of the previously found IfcBuilding instances, together with the properties of these same IfcBuilding instances. This approach is executed until all IFC building elements and their properties are found.

Graph nodes

By using the corresponding classes between the ifcOWL and the LBD modular ontologies (BOT, PRODUCT, PROPS) displayed in Table 2.2, the newly created instance resources get the right LBD OWL classes assigned. If PROPS L2 or L3 is selected, new instance nodes for properties are created. Unique property set

(pset) instances do exist in the ifcOWL-based Abox graphs and as such, they can be converted directly into new instances of props:PsetGroup. In the case of PROPS L3, also new instance nodes for property states are made for each property. The URIs of the new building element instance nodes are constructed as `<base URL><human readable label>_<GUID>`. The human readable label is derived from the PRODUCT class or - if it is not a bot:Element - from the corresponding BOT class. The GUID is always the IFC GUID of the corresponding IFC object in hexadecimal notation.

Instantiated property nodes (PROPS L2 and L3) have URIs in the form of `<base URL><propertyName>_<GUID>`, where the GUID is identical to the one of the connected BOT resource. In the case of property set instance nodes (PROPS L2 and L3), the URIs are constructed as `<base URL>psetGroup_<psetName>`. Similarly, the URIs of attribute group instances follow the `<base URL>attributeGroup` structure. Finally, the state instances (PROPS L3) have URIs in the form of `<base URL>state_<propertyName>_<GUID>_<timestamp>`. An extra filter was implemented in the code to exclude redundant properties, i.e. properties that contain no value, an empty string value or a string value that is identical to the property name.

Graph edges

At the moment of writing, the converter simplifies seven different ifcOWL Abox triple patterns into seven corresponding direct BOT relation (see Table 2.3), depending on the type of start node. The bot:adjacentZone relations are not yet implemented as no clear property path between such IFC elements was found. The pattern P1 for finding the IfcBuilding from the respective IfcSite is given in Listing 3 using the SPARQL language.

Regarding the building-related properties, a distinction between IFC properties stored in property sets and IFC attribute properties is made. Attribute properties do not reside in property sets but are directly attached to its building element in IFC. In the first case, three triple patterns from the ifcOWL Abox graph are needed (Listing 4): one for finding the properties in property sets of a building element (P8a), one for the property name (P8b) and one for the property value of such a property (P8c). In the second case, the general rule (path P9) is to list all attributes that have a String, Integer, Double, or Boolean data type. This approach only retrieves tag_IfcElement (the BIM Autoring Tool ID or batid), globalId_IfcRoot (the IFC GUID), name_IfcRoot, objectType_IfcObject, refElevation_Ifcsite and longName_IfcSpatialStructureElement, while for example IfcMaterial, IfcPresentationLayer, country_IfcPostalAddress, etc. are not covered by this

Table 2.2. Corresponding ifcOWL and LBD classes

ifcOWL classes	**BOT classes**
IfcSite	Site
IfcBuilding	Building
IfcBuildingStorey	Storey
IfcSpace	Space
IfcElement / IfcFurnitureType	Element
ifcOWL classes categorized as bot:Element	**PRODUCT classes**
IfcWall / IfcWallStandardCase	Wall
IfcDoor	Door
...	...
ifcOWL classes	**PROPS classes (L2 and L3)**
IfcPropertySet	PsetGroup

path. Specific property paths will be necessary to convert these other attribute properties.

Duplex House IFC case study

The current IFCtoLBD converter was tested on the IFC file of the Duplex apartment benchmark model (`Duplex_A_20110907.ifc`), published by NIBS [11]. The initial IFC building model in SPF format (IFC2x3 TC1) of 2.3 MB is converted into an RDF graph using the existing IFC-to-RDF converter (ifcOWL) and compared with the results of the new IFCtoLBD converter (BOT-PROPS-PRODUCT). The conversion with the IFCtoLBD converter was executed three times, with different settings for the PROPS level of complexity. In the case of PROPS L2 and L3, all property and state instance nodes are converted into a stable URI instead of using blank nodes.

[11] https://www.nibs.org/?page=bsa_commonbimfiles

Table 2.3. ifcOWL property paths and corresponding LBD relation or property path

ifcOWL property paths	**ifcOWL class of start node**	**Corresponding LBD property or property path**
P1	IfcSite	bot:hasBuilding
P1	IfcBuilding	bot:hasStorey
P1 / P2	IfcBuildingStorey	bot:hasSpace
P3 / P4	IfcBuildingStorey	bot:containsElement
P3	IfcSpace	bot:containsElement
P5	IfcSpace	bot:adjacentElement
P6 / P7	IfcWall / IfcRoof / ...	bot:hostsElement
P1	IfcStair / ...	bot:aggregates
P8a, P8b and P8c (properties part of psets)	(any building element found above)	PROPS L1 / L2 / L3 (see Listing 1)
P9 + specific property paths (IFC attribute properties)	(any building element found above)	PROPS L1 / L2 / L3 (see Listing 1)

Reduced graph size

The results of the conversion to LBD Abox graphs are presented in Table 2.4. The ifcOWL Abox Turtle file counts 227,143 triples (17428.0 KB file size). First of all, there is a clear decrease in number of triples, because all geometry-related triples from the ifcOWL-RDF are excluded: [Pauwels and Roxin, 2016] noted a reduction in triples of about 38 % only by removing the IFC geometry-related triples. The number of property steps between building-related elements (e.g. spaces and storeys) decreased from two steps to one by applying the BOT ontology properties. Building-related properties are handled now as is typically done in a Semantic Web

Listing 3. Property path P1 query to find the IfcBuilding instances of a known IfcSite

```
SELECT ?building WHERE {
  inst:IfcSite_38274 ^ifcowl:relatingObject_IfcRelDecomposes ?relJoin .
  ?relJoin ifcowl:relatedObjects_IfcRelDecomposes ?building . }
```

Listing 4. Property paths P8a, P8b and P8c for finding property sets of a certain building element and the properties (values and property names) they contain

```
SELECT ?psets ?propNames ?propValues WHERE {
  # path P8a: properties of property sets of a know element
  inst:IfcSlab_37864 ^ifcowl:relatedObjects_IfcRelDefines ?relJoin .
  ?relJoin ifcowl:relatingPropertyDefinition_IfcRelDefinesByProperties
    ?psets .
  ?psets ifcowl:hasProperties_IfcPropertySet ?propSingleVal .

  # path P8b: property names
  ?propSingleVal ifcowl:name_IfcProperty ?propNameJoin .
  ?propNameJoin express:hasString ?propNames .

  # path P8c: property values
  ?propSingleVal ifcowl:nominalValue_IfcPropertySingleValue ?propValJoin .
  ?propValJoin express:hasString | express:hasBoolean | express:hasDouble
         | express:hasInteger | express:hasLogical ?propValues . }
```

environment, leaving out redundant intermediate nodes. In this way, the number of steps/relations between a building element and a property value is reduced from five to one, two or three steps depending on the selected PROPS level of complexity. The element's property name is now modeled as a datatype/object property (L1) or object property (L2 and L3), instead of a string literal.

In total, a decrease of 83% in the overall number of triples (80% in file size) is demonstrated when all three modules are exported to one file with PROPS L3 selected, compared to the ifcOWL Abox graph of the same IFC building model. When other PROPS levels are selected, or other Abox modules are left out, the file size and number of triples can even be further reduced. The use of Abox modules make it easier to share and study specific parts of the converted LBD graph. Regarding the PROPS module, users can select the most appropriate level of complexity they need for their project.

Simplified queries

As the original ifcOWL Abox graph reduced significantly in number of triples by applying the LBD ontologies, the resulting Abox graph structure becomes simpler while the information in the graph is equivalent to the one in the ifcOWL-based Abox graph. As a logical consequence, query writing also becomes more clear and shorter, as shown in Listing 5 for BOT (find the bot:Building instances of a specific bot:Site) and Listing 6 for PROPS (find the loadBearing property value of a specific element). These queries can be compared with the earlier presented queries for

Table 2.4. IFCtoLBD converter output results per module

BOT	PRODUCT		PROPS	TOTAL
45.5 KB (729 triples)	18.8 KB (263 triples)	PROPS L1	397.7 KB (6099 triples)	441.2 KB (7091 triples)
		PROPS L2	1665.3 KB (19831 triples)	1708.4 KB (20823 triples)
		PROPS L3	3399.0 KB (38128 triples)	3442.4 KB (39120 triples)

Listing 5. SPARQL query on the output of the IFCtoLBD converter (all instances of bot:Building of a known site)

```
SELECT ?site ?buildings WHERE {
  BIND(inst:site_7b7032cc-b822-417b-9aea-642906a29bd7 AS ?site)
  ?site bot:hasBuilding ?buildings . }
```

the ifcOWL Abox graph, respectively in Listing 3 and Listing 4. In the first case, the number of intermediate steps between both objects drops from two to one. In the second case, the amount of intermediate steps between a property value and the object is reduced from five to one (PROPS L1).

2.1.3 Conclusion

The IFCtoLBD converter facilitates the conversion of IFC building models to modular LBD Abox graphs structured according the BOT, PROPS and PRODUCT ontologies designed within the W3C LBD CG. The resulting Abox graphs are more user-friendly than previous ifcOWL Abox graphs, because the graph structure is simplified and closer to what would be expected in a Semantic Web environment.

Listing 6. SPARQL query on the output of the IFCtoLBD converter (the loadBearing property value of a certain slab; the property is of PROPS L1)

```
SELECT ?propValue WHERE {
  BIND(inst:slab_982f59b0-f2e1-485f-8ce1-c9f6117b70a9 AS ?slab)
  ?slab props:SlabCommon_loadBearing_simple ?propValue . }
```

As a consequence, the output graphs are smaller in size and SPARQL queries become significantly shorter and easier to understand, as demonstrated with the Duplex house use case. Users can test the three different PROPS levels as currently proposed in the W3C LBD CG, and provide crucial feedback to the PROPS ontology subgroup. By using PROPS L3, it becomes possible to do - with little extra implementation effort - versioning of properties in IFC building models.

Future research includes more elaborate use cases for the converter and a detailed analysis of the improved query execution times compared to ifcOWL Abox graphs. The conversion of IFC unit information using the Custom Datatype properties (CDT) ontology and a feasibility study regarding ifcOWL property paths corresponding to bot:adjacentZone relations and bot:adjacentElement relations between bot:Storey and bot:Element instances will be addressed. The conversion of all relevant IFC attribute properties will also be documented and implemented in future versions of the converter.

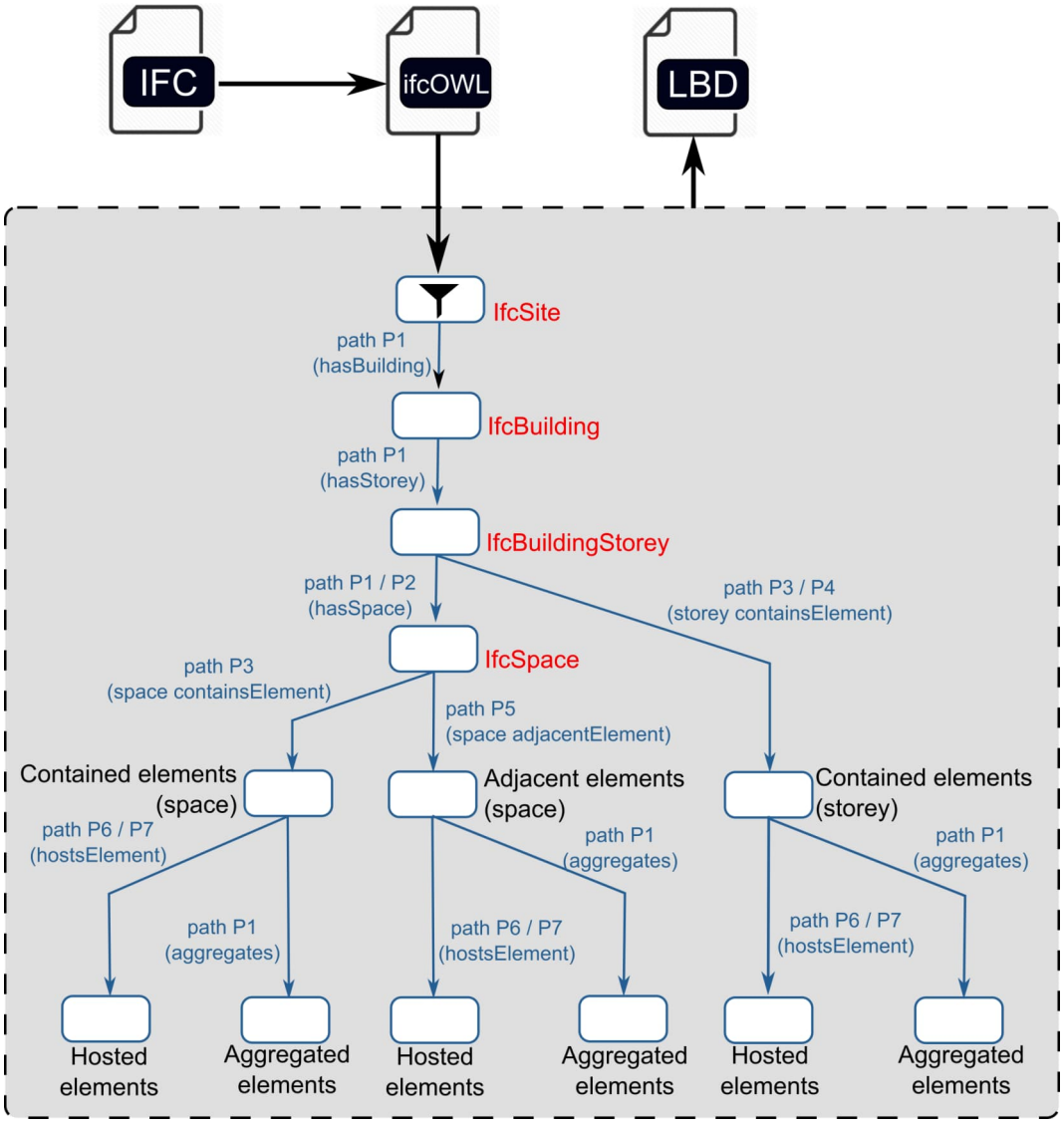


Fig. 2.3. Implemented IFCtoLBD conversion process regarding BOT

References

Jakob Beetz, Jos Van Leeuwen, and Bauke De Vries. IfcOWL: A case of transforming EXPRESS schemas into ontologies. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing: AIEDAM*, 23(1):89–101, 2009. ISSN 08900604. doi: 10.1017/S0890060409000122. ISBN: 0890060409000.

Tarcisio Mendes de Farias, Ana-Maria Roxin, and Christophe Nicolle. IfcWoD, Semantically Adapting IFC Model Relations into OWL Properties. In *Proc. of the 32nd CIB W78 Conference 2015, 27th-29th October 2015, Eindhoven, The Netherlands*, pages 175–185, 2015.

Pieter Pauwels and Anna Roxin. SimpleBIM : From full ifcOWL graphs to simplified building graphs. In Symeon Christodoulou and Raimar Scherer, editors, *eWork and eBusiness in Architecture, Engineering and Construction (ECPPM)*, pages 11–18, Limassol, Cyprus, 2016. CRC Press. ISBN 1138032808.

Pieter Pauwels and Walter Terkaj. EXPRESS to OWL for construction industry: Towards a recommendable and usable ifcOWL ontology. *Automation in Construction*, 63:100–133, 2016. ISSN 09265805. doi: 10.1016/j.autcon.2015.12.003.

M. Rasmussen, P. Pauwels, M. Lefrançois, G. F. Schneider, C. Hviid, and J. Karlshøj. Recent changes in the building topology ontology, in: Ldac2017-5th linked data in architecture and construction workshop. https://backend.orbit.dtu.dk/ws/portalfiles/portal/140019287/Untitled.pdf, 2017a. Paper presented at LDAC2017 5th Linked Data in Architecture and Construction Workshop, Dijon, France.

Mads Holten Rasmussen, Pieter Pauwels, Christian Anker Hvidd, and Jan Karlshøj. Proposing a Central AEC Ontology That Allows for Domain Specific Extensions. In *LC3 2017: Proceedings of the Joint Conference on Computing in Construction*, pages 237–244, Heraklion, Greece, 2017b.

Walter Terkaj and Pieter Pauwels. A Method to generate a Modular ifcOWL Ontology. In *Proceedings of the 8th Workshop Formal Ontologies Meet Industry,*

Joint Ontology Workshops 2017, CEUR Workshop Proceedings, volume 2050, Bolzano, Italy, 2017.

Walter Terkaj, Georg Ferdinand Schneider, and Pieter Pauwels. Reusing Domain Ontologies in Linked Building Data : the Case of Building Automation and Control. In *Proceedings of the 8th Workshop Formal Ontologies Meet Industry, Joint Ontology Workshops 2017, CEUR Workshop Proceedings*, volume 2050, Bolzano, Italy, 2017.

2.2 Dynamic BIM format conversion as inference-based ontology alignment (LDAC 2021)

Consider a scenario where a facility manager stores all building-related data in a knowledge base. She wants to keep all the provided data, but have easy access to the information, i.e. shortcuts and high-level data abstraction. Then a renovation is carried out, and a new as-built model is provided. The inference model automatically updates the model to keep the abstraction extension up to date.

Hypothesis 2.2 suggests that the implementation of a rule-based conversion approach is not only possible, but also better than an algorithm-based converter in terms of keeping models up-to-date after modifications. The claim includes an assumption that inferencing models are automatically or semi-automatically reacting the updates to the model. This seems to be true for Apache Jena inferencing engine, Stardog, and GraphDB. According to the Apache Jena documentation [The Apache Software Foundation, 2022], when the standard API is used to modify an inference model by adding or removing statements, additional rule firings are triggered. Stardog incorporates reasoning capabilities through query re-editing[12], where reasoning is invoked dynamically upon the execution of a relevant query. Then, GraphDB automatically[13] updates inferred facts in response to changes made to the underlying data model. These ensure that the offered data remain up to date. In contrast, the algorithm-based solution is only executed and the data refreshed when explicitly invoked.

Early binding and late binding are two different approaches to handling inference results. In the early binding approach, the inference results are immediately incorporated into the inference model. If only the conversion result is needed, the added triples must be

[12] https://docs.stardog.com/inference-engine/ (Accessed June 13, 2023)

[13] https://graphdb.ontotext.com/documentation/10.2/inference.html(Accessed June 13, 2023)

filtered out of the inference model. On the other hand, in the late binding approach, only partial results are generated, which gives a performance advantage, but to export the full conversion requires querying the entire dataset. In the given usage scenario, both approaches are well suited to achieve the intended purpose.

Table 2.5 summarises the use of the design science research method in this research.

My contribution to the paper: In the research work, my contribution concentrated on two aspects. First, I modified the IFCtoLBD converter [Oraskari et al., 2021a] source code so that the performance could be compared with the Semantic Web Rule Language (SWRL) rules rule-based conversion. Then, I created a test setup for the rule-based conversion and the IFCtoBOT[14] converter. Pierre Bourreau created the SWRL rules separately. We analysed the results together and discussed the limitations.

Table 2.5. The work of Bourreau & Oraskari [Bourreau and Oraskari, 2021a] in the Hevner et al. [Hevner et al., 2004] design science research framework

Guideline 1: Design as an artifact
The Semantic Web Rule Language (SWRL) inference rules were created to translate ifcOWL models into the BOT [Rasmussen et al., 2021]. This research works was a proof of construction for a rule-based IFC to LBD converter. The presented Semantic Web Rule Language (SWRL) rules can be applied in any RDF inference environment that supports the rule language. In addition, the general abstract ideas of the rules are described. So, the rules can be rewritten in other RDF rule languages also.

[14] https://github.com/NIRAS-MHRA/IFC2BOT(Accessed June 13, 2023)

Guideline 2: Problem relevance

As this asset creates LBD models, it shares the business problems of the IFCtoLBD converter presented in Section 2.1. However, the capability to keep the converted model updated while keeping the original model suits well in the situation where the model is updated frequently, and the conversion is wanted to be consistent all the time. This could be the case when applications that use different conceptual models would communicate via the dataset, and so the rule-based conversion would offer simultaneous translation for the applications.
There had been converters that convert IFC models into Linked Building Data ABox models. What had been missing was rule separation from the converter software.

The constituent user community of the artefact are the researchers of the Linked Building Data Community.
The solution belongs to the converter, and more specifically, the LBD converter problem class.
This work is an enabler for the following BIM use cases: Quantity take-off of, Author Cost Estimate [The Pennsylvania State University, 2013], and Building operation and maintenance [Borrmann et al., 2018b].
Simple quantity estimates are possible, the rule-based conversion has the limitation that it does not cover property or quantity sets, data units, or geometry.

Guideline 3: Design evaluation

Hevner et al. Design Research Evaluation Method: In a Functional (Black Box) Testing and Experimental Controlled Experiment, the Apache Jena Inference model was used to test the conversion of the Dunant and Duplex BIM models to discover failures and identify defects.
The conversion performance was compared with the IfcOpenShell[15] based Python converter IFC2BOT[16] and the Apache Jena based IFCtoLBD [Oraskari et al., 2021a] Java solution.
Since the rule-based model uses ifcOWL ABox RDF models as input instead of IFC/STEP files, adaptations were made to make the tools comparable.

Guideline 4: Research contributions

The new method showed that translation from ifcOWL ABox models to LBD ontologies is possible. Also, the limitations of such a solution were discussed.
Generalization of the results: In the Linked Building Data domain, the vision has been to connect all relevant information to provide a single source of information for the communication parties in the AEC sector. This article presents the conversion from ifcOWL to BOT. The same approach can be used to translate other RDF datasets also.

Guideline 5: Research rigor

The work was peer-reviewed. Also, the axioms and rules were included in the appendix part.
I implemented the presented conversion rules using Apache Jena GenericRuleReasoner[The Apache Software Foundation, 2022]. The implementation can be found on my GitHub page [17].

[15] https://ifcopenshell.org/ (Accessed June 13, 2023)
[16] https://github.com/NIRAS-MHRA/IFC2BOT(Accessed June 13, 2023)
[17] https://github.com/jyrkioraskari/IFCtoB4R-DM_OpenAPI (Accessed June 13, 2023)

Guideline 6: Design as a search process

This work presents an alternative way to implement the conversion from IFC data (the RDF serialization of IFC, ifcOWL[18]) to LBD: As such, the presented research suits as a design science research search step for a conversion architecture.

Guideline 7: Communication of research

LDAC2021 - 9th Linked Data in Architecture and Construction Workshop (11 - 13 October 2021) in Luxembourg.

[18] https://technical.buildingsmart.org/standards/ifc/ifc-formats/ifcowl/ (Accessed June 13, 2023)

The published work was made available as an open-access publication under the Creative Commons License Attribution 4.0 International (CC BY 4.0 [19]). The Linked Data Architecture and Construction Workshop 2020 editors, RWTH Aachen, CEUR-WS and archiving partners were given irrevocable rights to make the work available. The work was originally published in [Bourreau and Oraskari, 2021a]. The content presented here is based on our final draft of the article.

Pierre Bourreau Nobatek/INEF4, 9 rue Jean-Paul Alaux, 33000 Bordeaux, France, pbourreau@nobatek.inef4.com
Jyrki Oraskari CAAD RWTH Aachen, Templergraben 55, 52062 Aachen, Germany, Jyrki.Oraskari@dc.rwth-aachen.de

Abstract: The AEC sector is known to be highly fragmented, different experts requiring different information. Current BIM collaborative practices can be described as static as they are based on file exchange, mainly using IFC files. Instead, in an ideal level 3 BIM platform, different stakeholders could work together on a single centralized building model, using different domain-specific vocabularies, and updating the model synchronously. We propose an approach based on semantic rule reasoning to dynamically convert information in different vocabularies. While existing alignments only cover class conversions, we develop inference rules that handle conversion of relations in a dynamic and transparent way, being executed by a reasoner. The approach is implemented and tested on converting models from ifcOWL to BIM4Ren, a BOT-based ontology. The scalability of the implementation is discussed as well as its limitation.

keywords: BIM, alignment, logic programming, interoperability

2.2.1 Introduction

The fragmentation of the AEC industry is a well-known issue in construction projects: different AEC (Architecture and Engineering in Construction) actors use different vocabularies and information to describe the same building. BIM as a collaborative process supported by digital tool and models intends to ease data exchange between project stakeholders. To do so, the industry delivered IFC or the ISO 10303-21 IFC-SPF file format, which aims at being a standard exchange model to which all software vendors can comply, at the cost of a translation mechanism. We call such a data exchange 'static' as each transferred model is a picture of the model at some instant; indeed, if multiple actors are allowed to update it, some file merging functionalities become necessary. In an ideal BIM level 3 environment, as mentioned in the literature, all actors work on a single and centralized model, and

[19] https://creativecommons.org/licenses/by/4.0/legalcode

updates are handled automatically so that each actor works on the latest model version. In the present article we investigate a linked data-based approach that reduces model conversion to an ontology mapping problem, for a model stored in a single triple store. With such approach, different users may use different vocabularies to work on the very same centralized model. It could be used for 'horizontal' (i.e. from one model to another, like IFC2x3 to gbXML) or 'vertical' conversion (i.e. from one model version to another one, like IFC2x3 to IFC4)

Ontology mapping can be seen as a task where concepts (i.e. entities and relations) of different vocabularies are mapped based on their semantic correspondence and organisation, keeping the intended interpretations [Kalfoglou and Schorlemmer, 2005, Mao, 2007]. In the Linked Building Data domain, a few ontology alignments exist [Schneider, 2017]. While many ontology mapping techniques exist, the article focuses on logic-based mapping by elaborating inference rules that are interpreted by a reasoner that can dynamically generate matching relations.

The article is structured as follows: section 2 introduces the problem of collaboration and model conversion in the BIM community, and ontology mapping. We also introduce ifcOWL and the BIM4Ren ontology as the two ontologies used to test our approach. Section 3 presents different ontology mapping strategies used to align the two ontologies above. In section 4, we introduce the implementation, test it, discuss issues and solutions proposed, and compare the approach with a programmatic one.

2.2.2 BIM collaboration and ontology mapping

2.2.3 BIM collaboration

BIM is the main asset towards a digitization of the AEC community. The sector being highly fragmented, data exchange between project stakeholders is an issue that BIM collaborative process supported by BIM models intends to solve. In this direction, BIM levels are often used to describe the collaboration maturity of a project: while almost no collaboration are required in a BIM level 0 or 1, BIM level 2 requires project stakeholders to exchange information through files. In an ideal BIM level 3, all actors work on the very same model, without any need of exchanging files.

At the moment, the community seems to focus its efforts on improving BIM level 2, through the delivering of different BIM standards: IFC (Industry Foundation Classes) is the main one; it is rich, extensible and covers most AEC domains. But its complexity seems to be a barrier to software vendors, and reaching an

agreement on the vocabulary and the required information that needs to be in such digital model is highly challenging, considering different countries or regions may have different requirements, and even projects can have their own specificities [van Berlo, 2020]. COBie is another lightweight building model, that focuses on non-geometrical information; in energy modelling gbXML is a *de facto* standard for building energy models. More recently, knowledge graph-based modelling gained some interests as a new paradigm to deliver BIM models: [Beetz et al., 2009, Pauwels and Terkaj, 2016] proposed ifcOWL as an OWL translation of the IFC EXPRESS schema format; [Rasmussen et al., 2019b] proposed BOT (Building Ontology Topology) as a minimal knowledge representation for any building project.

It therefore seems that while the community is aware of the need for stakeholders to share information, more and more data models are created. This could be contradictory with BIM level 3 principle where a single model, and therefore a single vocabulary is potentially used, unless some automatic conversions are implemented. While many such converters exist, they are all written in some specific programming language (Java, Python ...) and their implementation into a single BIM repository does not seem trivial.

Ontology mapping

In linked data, a knowledge graph is a set of statements in which nodes and edges have well-defined meanings. Linking knowledge graphs, known in the literature as *ontology mapping* or ontology matching, is done to benefit from the work of a specialized community; the resulting ontology linkage is a set of relations between concepts of different ontologies, known as an *ontology alignment* [Euzenat et al., 2007]. Ontology mappings can be done in different ways: for instance word distance, translation from one language to another, or graph pattern similarities, can be used to discover similarities between graphs.

The present work focuses on semantic matching techniques [Shvaiko, 2006, Jean-Mary et al., 2009], which relies on logical inference to deduce alignments between models. This is done defining:

1. (**Axioms**): direct ontology alignments between classes/relations; those take the form $\vdash$ Ax where Ax is a list of triples (i.e. usually called literals in formal logic) <s; p; o> where s belongs to an ontology O_1, o to an ontology O_2 and p is an alignment predicate.
2. (**Inference rules**) create logical rules that can be triggered thanks to the creation of new alignments; those are of the form (A $\vdash$ C, where A and C are

list of triples respectively called antecedents and consequents)

Triples in inference rules are usually *open* triples, i.e. they contain variables. Nevertheless, only *safe rules* can be triggered correctly; a rule is said to be safe when every variable in its consequent is already declared in its antecedent.

As an example, the transitive closure of the `rdfs:subClassOf` relation can be achieved with the two following rules: (i) states that an instance of `owl:Class` is a subclass of itself (i.e. reflexivitiy); and (ii) that a `rdfs:subClassOf` of a class y, is a subclass of all super classes of y (i.e. transitivity).

$$(i)\ \texttt{(?x rdfs:type owl:Class)} \vdash \texttt{(?x ex:transSubClass ?x)}$$

$$(ii)\ \texttt{(?x rdfs:subClassOf ?y) (?y ex:transSubClass ?z)} \vdash \texttt{(?x ex:transSubClass ?z)}$$

Finally, reasoners can evaluate rules in different ways: (i) a *forward-chaining* reasoning considers every triple in the ABox that instantiates the antecedent of a rule to generate new triples instantiating the consequent; the operation is repeated until a goal statement is reached, or no more triples are added to the ABox. (ii) a *backward-chaining* approach triggers rules on request to satisfy the instantiation of the consequent, through an instantiation of its antecedents. It is known that a forward-chaining avoids iterating in infinite loop, but is memory-consuming. On the other hand, backward-chaining can run in infinite loops if no memoization is turned on, but does not store unnecessary information. Combining both forward and backward-chaining evaluable rules is called *hybrid* reasoning.

ifcOWL and BIM4Ren

The present work focuses on converting ifcOWL models into the BIM4Ren vocabulary. IfcOWL [Beetz et al., 2009, Pauwels and Terkaj, 2016] is the direct translation of the EXPRESS-based IFC schema into the OWL language; its coverage is therefore as wide as the IFC one. The BIM4Ren data model [Bourreau et al., 2020] embodies a set of ontologies with the aim of modelling energy-efficient renovations. It resides on three independent layers (core, product and domain) and a transversal one (utility) as represented in Figure 2.4, which makes it a product-centric model in the sense that: (i) the core layer is made of ontologies that help relate products between them or inside a specific building context, (ii) the product layer is a taxonomy used to assign a specific class to a product, and (iii) the domain layer is used to associate properties to a product; according to specific domains.

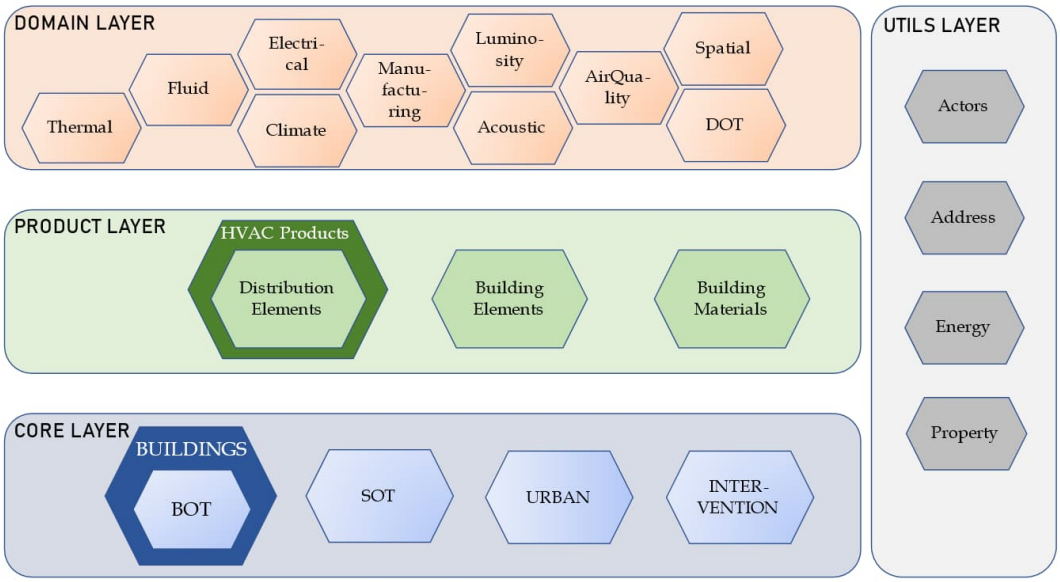


Fig. 2.4. The BIM4Ren data model architecture

Full modularity (i.e. sub-ontologies are pairwise independent: none needs importing another BIM4Ren ontology but from the Utils layer) is achieved through a multiple instantiation mechanism, that allow importing modules on demand. This enables one to use only small models of interest for a specific task instead of a complex multi-domain model.

As an example, a boiler `_:boiler1` may be modelled as an instance of `bot:BuildingElement` to locate it in the building; of `sot:SystemElement` to describe its role in the MEP networks; of `ifc4:Boiler-WATER` in the distribution elements model; and of `elec:Consumer` in the electrical ontology, where `elec:Consumer` comes with a `elec:nominalPower` relation.

At the technical level, the BIM4Ren ontology uses (i) an extension of BOT [Rasmussen et al., 2019b] to describe buildings, with reference to Omniclass classification to assign functional properties to buildings (i.e. apartment, allotment,...) and spaces (kitchen, bedroom,...); (ii) the product layer is made of taxonomies extracted from ifcOWL to classify distribution[20] and building elements[21], where enumerated product types are transformed into classes; (iii) and the property layer uses SML, the property modelling on-going standard provided by the CEN[CEN, 2020], which is based on QUDT [Hodgson et al., 2014] and Ontology for Property Management (OPM) level 2 [Rasmussen et al., 2018].

[20] https://pi.pauwel.be/voc/distributionelement/index-en.html
[21] https://pi.pauwel.be/voc/buildingelement/index-en.html

2.2.4 Model conversion as inference

In the present section, a rule-based ifcOWL to BIM4Ren conversion is presented. As mentioned in section 2.2.3, we first identify the relations used to align the two models (i.e. the axioms) and then detail the inference rules that are used.

Axioms

Different vocabularies can be used to directly map classes or relations of different models. The `rdfs:subClassOf` or `owl:equivalentClass` relations are used in the current implementation to map classes, in the same way as it is done in the existing alignment between BOT and ifcOWL [22]. Extending this approach to the SOT ontology leads to the following set of rules:

```
ifc:IfcDistributionSystem rdfs:subClassOf sot:System.
ifc:IfcDistributionElement rdfs:subClassOf sot:SystemComponent.
ifc:IfcDistributionPort rdfs:subClassOf sot:TransportElement.
ifc:IfcFlowTerminal rdfs:subClassOf sot:TerminalElement.
```

These rules form the axioms of our conversion process from ifcOWL to BIM4Ren. Nevertheless, it is worth mentioning that other vocabularies are used to map the BIM4Ren models with other vocabularies, in particular Omniclass and Property Set Definition (PSD).

As mentioned in 2.2.3, Omniclass is used to extend BOT with concepts on building and space types. Because to our knowledge no web referenceable version of Omniclass exists, every created subclass is bound to its corresponding associated Omniclass code through the `rdfs:isDefinedBy` relation (e.g. `buildings:Bathroom rdfs:isDefinedBy "Omniclass 13-65 13 15"`.

In addition, properties of the domain layer are mapped with their PSD counterpart, which are web referenceable through the buildingSmart Data Dictionary (bsDD). A BIM4Ren property may be associated to multiple bsDD properties (see Figure 2.5), using the `skos` vocabulary.

[22] https://raw.githubusercontent.com/w3c-lbd-cg/bot/master/IFCOWL4_ADD2Alignment.ttl

```
:returnTemperature
  rdf:type owl:ObjectProperty ;
  rdfs:comment "The temperature of the input medium in a thermal loop" ;
  rdfs:domain :Producer ;
  rdfs:label "return temperature"@en ;
  rdfs:subPropertyOf b4r-prop:temperature ;
  skos:narrowMatch <http://bsdd.buildingsmart.org/api/4.0/IfdConcept/1PxGW0qSuHuO00025QrE$V> ;
  skos:narrowMatch <http://bsdd.buildingsmart.org/api/4.0/IfdConcept/1WUt_0qSuHuO00025QrE$V> ;
  skos:narrowMatch <http://bsdd.buildingsmart.org/api/4.0/IfdConcept/2ZqF20qSGHuO00025QrE$V> ;
.
```

Fig. 2.5. Example of alignment with IFD through bsDD

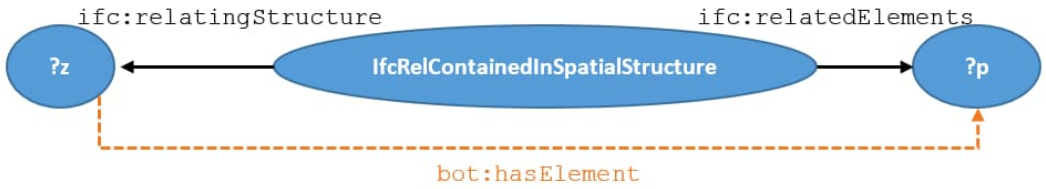


Fig. 2.6. Example of deobjectification of relations

Inference rules

The presented axioms are limited to mapping ifcOWL classes to BIM4Ren, as a direct mapping (using `rdfs:subPropertyOf` for instance) is impossible. Indeed, while relations in ifcOWL are often objectified (i.e. a class is used to model them, with the `IfcRel` prefix), linked building data ontologies simplify modelling using a single relation. This is illustrated in Figure 2.6.

Deobjectifying a relation can easily be translated as an inference rule:
`(?z rdf:type ?cl) (?cl rdfs:subClassOf bot:Zone)`
`(?c ifc:relatingStructure_IfcRelContainedInSpatialStructure ?z)`
`(?c ifc:relatedElements_IfcRelContainedInSpatialStructure ?p)`
$\vdash$ (?z `bot : hasElement` ?p)

Some complementary internal rules need to be elaborated to deal with the list structure of ifcOWL. For instance the range of the `IfcRelNests_relatedObjects` relation is a list, which type in ifcOWL is `https://w3id.org/list`; it is a LISP-like list structure made of `list:hasContents` to get the head, and `list:hasNext` to return the list from which the top element was removed [Pauwels et al., 2015]. Handling lists in the converter requires to have a way to directly check whether an element is in a list, which can be done with the following recursive rules:

`(?l list:hasContents ?elt)` $\vdash$ `(?elt list:isIn ?l)`

`(?l list:hasNext ?q) (?elt list:isIn ?q)` $\vdash$ `(?elt list:isIn ?l)`

Finally, classes in the Product layer of the BIM4Ren data model also need to be aligned with the corresponding enumerated types in IFC. For each enumerated type, a conversion rule can be elaborated as below:

```
(?z rdf:type ifc:IfcBoiler) (?z ifc:predefinedType_Boiler ifc:WATER) ⊢
(?z rdf:type distribution:Boiler-WATER).
```

In order to deal with all enumerated types in the Distribution and Building element ontologies, a Python script was developed. This leads to generating 693 rules amongst the 708 total rules.

2.2.5 Implementation and tests

In order to evaluate our approach, we compare it with a program-based converter considering (i) correctness of the conversion, (ii) performance, (iii) maintenance, (iv) the full conversion workflow.

Description

An implementation of our approach was done on an Apache Jena instance. All rules were implemented according to the Generic Rule Reasoner file format. The program-based converter to which results were compared is a Java-based converter[23], an extension of the IFCtoLBD [Oraskari et al., 2020] converter. Regarding the inference engine approach, we tested two different reasonings: a simple forward-chaining and a hybrid one, which differs from the former in that:

(i) It uses an ad hoc `subClass` relation: the inference rules need to check that some individuals are instances of specific classes, or of descendant of such classes. The `b4r:subClassOf`) is the transitive closure of `rdfs:subClassOf` but it is restricted to ancestor classes used in the inference rules (i.e. `bot:Zone`, `sot:System` and `sot:SystemComponent`).

(ii) It uses backward-chaining on complex rules: all rules used to align the core layer are converted to top-down rules. This can be simply done in Jena's reasoner[24].

(iii) It memorizes recurrent inferred triples: Some of the rules do not directly solve alignment issues but can be seen as utility rules. This is the case of the

[23] https://github.com/jyrkioraskari/IFCtoB4R-DM_OpenAPI/

[24] https://jena.apache.org/documentation/inference/index.html#RULEbackward

Table 2.6. Summary of expected relations - generated with IFC2BOT

Relations	Dunant	Duplex_MEP
bot:hasStorey	2	3
bot:hasSpace	0	42
bot:hasElement	59	926
bot:containsElement	0	167

b4r:subClassOf and list:isIn rules. Because these rules can be used extensively inside other rules and because they potentially generate a high number of triples, a memorization mechanism is enable to allow keeping results in memory and to avoid infinite loops.

Two different IFC models were considered for testing. The first one (called Dunant.ifc) results from the scanning of a multi-dwelling building in Paris as part of the BIM4Ren project. The corresponding ifcOWL ttl file is made of 10033 individuals, from which only 138 are not related to geometry; those are instances of IfcSite, IfcBuilding, IfcStorey, IfcSlab, IfcCovering, IfcWall, IfcDoor and IfcWindow. To test how scalable the approach is, the Duplex_MEP model [Johnson and Fallon, 2011] was considered. Duplex is two-storeys appartement model, well-known in the IFC community[25]; it is made of 687363 individuals (apart from geometric information) and more importantly for our experiments, contains instances of IfcSpace of the DistributionElement hierarchy, so most of the inference rules can be tested. Those rules were created to infer the hasBuilding, hasStorey, hasSpace and hasElement relations in BOT, and the hasElements (i.e. when an element is a component of a system) and connectedWith (i.e. when two MEP products are connected) relations in SOT.

Regarding the methodology, (i) we ensure the correctness of the rules through a comparison of the BOT classes and relations output by our converters (obtained through SPARQL query) with the ones output by the IFC2BOT [26]; (ii) we compare the execution performance of the inference engine approach with the one of the programmatic approach; to do so, we perform queries on a triple store for which an inference engine is running, and on another one on which the model output by the IFCtoLBD tool was uploaded.

[25] available at https://github.com/buildingSMART/Sample-Test-Files/tree/master/IFC%202x3/Duplex%20Apartment

[26] https://github.com/NIRAS-MHRA/IFC2BOT

Table 2.7. Performance test results (times in ms)

Operation	Forward-chaining		Hybrid		IFCtoLBD	
	Dunant	Duplex	Dunant	Duplex	Dunant	Duplex
1. Get all `bot:Building`	4138	Err	159	225	725	3902
2. Get all `bot:Space`	766	Err	157	150	645	4338
3. Get all `bot:Zone`	745	Err	138	10	915	4406
4. Get all `bot:containsZone`	3	Err	20	39	938	3607
5. Get all `bot:hasElement`	4	Err	5	92	977	4494
6. Get all `sot:SystemComponent`	732	Err	137	150	-	-

Results

First, the tests prove some scalability issues for the forward-chaining approach: Jena stopped with a "java.lang.OutOfMemoryError: Java heap space" error, after the execution hang for more than 30 minutes on the first query performed on the Duple_MEP model. As already mentioned, forward-chaining reasoning may require keeping in memory useless triples, and by default the GenericRuleReasoner infers all new triples on the first query performed. The hybrid reasoner runs successfully on all models, and all queries, proving the efficiency of the different modifications performed.

Regarding the correctness of the approach, the hybrid reasoning's BOT outputs are sound and complete with respect to the IFC2BOT outputs, on both the Dunant and Duplex_MEP files, ensuring its correctness. The same happens for the forward-chaining reasoning on the Dunant file.

Regarding execution performance, the results are described in Table 2.7. Operations were run in the same order given as in the table.

First of all, for the forward-chaining reasoning, getting all buildings takes longer than getting all spaces or zones, but this is due to the testing procedure: getting buildings was the first query performed, and it therefore triggers the reasoner that infers all new triples, a time-consuming operation. In fact when repeating the

query, answers are provided within approximately 700ms.

The hybrid evaluation performs better than the forward-chaining one on queries 1, 2 and 3 on Dunant. The performed queries are of the form: `SELECT ?z WHERE {?z a ?cl. ?cl REL ENT.}`, where `ENT` is to be replaced with the goal class (`bot:Building` on 1., `bot:Space` on 2. and `bot:Zone` on 3.) and `REL` with `rdfs:subClassOf*` (i.e. the transitive closure of `rdfs:subClassOf`) for the forward-chaining evaluation, and with `b4r:subClassOf` for the hybrid one. This gives an idea of the efficiency provided by adding the former over the latter. To confirm it, getting all buildings on Duplex in the hybrid approach with the `rdfs:subClassOf*` relation takes 22 533ms.

Generally speaking, the hybrid reasoning outperforms the forward-chaining one for almost all queries on the Dunant file; this tends to show there is no clear advantages on computing all inferred triples at initialization. This is surprisingly clear with queries 4 and 5, where the relations are inferred with the help on non-axiomatic rules and may therefore require high runtime computation in the hybrid implementation. Moreover, the more different queries are run on the hybrid model, the better it performs. Indeed, memorization of the results (either intermediate or final) of multiple queries ensures those can be reused without any computational cost for future queries.

In addition, queries on a logic-based approach are competitive with the Java program, if not outperforming it, in the case of the hybrid implementation. The high difference between the hybrid approach and IFCtoLBD seems again to reside on the use of the `b4r:subClassOf` relation.

Regarding maintenance, the inference rule-based approach may not be as easy to develop or make evolve as a program-based approach. Indeed, debugging inference rules in Jena using a configuration file for the rules is not trivial, and, to our knowledge, no tool exists to assist developers in this task as opposed to debugging environments for long-time consolidated programming languages. Moreover, the syntax for the inference rules may differ from a triple store to another one.

Finally, regarding the workflow, the inference rule-based approach resides in using a triple store as a central model storage, and its native engine as a converter; no deployment effort is therefore needed as all required functionalities are available within Jena or any other triple store. The program-based conversion requires (i) to download the ifcOWL model, (ii) convert it to the BIM4Ren model, (iii) upload it again to the triple store. This process can probably be automatized, but the advantage in using an inference-rule based approach is that every changes made, using IFC (resp. BIM4Ren) vocabulary, is automatically reflected in the BIM4Ren

(resp. IFC) one.

2.2.6 Conclusion

Advantages and limitations

The experiments show that the linked-data approach is efficiently dealing with the conversion of models from IFC to BIM4Ren. Nevertheless, the approach is scalable at the cost of some tuning on the rules evaluation. The hybrid approach differs from the forward-chaining one as the latter converts parts of the model on-demand, while the former maintains the conversion at run time. Nevertheless, both dynamically convert only the necessary part of the model, which is the main difference with a programmatic approach, that converts the the full model at once. A programmatic approach may have a wider coverage, being able to deal with geometry conversion as well as properties, but our approach is language-independent, can be directly implemented on any triple store and is dynamic.

The current work also led to some diagnosis on ifcOWL. As mentioned, we created specific rules for the `isIn` relation, which checks the presence of an element inside an `owl:OWLList`. Using `rdf:List` instead, the built-in `swrlb:member` could be used[27], which may improve the overall performance of the converter.

Finally, some experiments were done on backward conversions, i.e. from BIM4Ren to ifcOWL. The main issue is that backward inference rules are unsafe, because ifcOWL uses objectified relations, introducing therefore more concepts than BIM4Ren. In practice, this can be bypassed using the `makeTemp` Jena built-in function. This is illustrated in the rule above where the variable `?c` is the objectified relation to create (i.e. the main element in the consequent), and as such, it needs to be declared in the antecedent.

```
(?z rdf:type ?clz) (?clz b4r:subClassOf ifc:IfcSpatialElement)
(?z bot:hasElement ?p)
(?p rdf:type ?clsp) (?clsp b4r:subClassOf ifc:IfcProduct)
makeTemp(?c)
-> (?c rdf:type ifc:IfcRelCointainedInSpatialStructure)
(?c ifc:relatingStructure_IfcRelContainedInSpatialStructure ?z)
(?c ifc:relatedElements_IfcRelContainedInSpatialStructure ?p)
```

[27] see 8.7 of https://www.w3.org/Submission/SWRL/

Future work

In the future, the approach should be extended to cover PSD properties and geometry. Regarding properties, a mapping will be done in the BIM4Ren project between the model and a POBIM (ISO 12006-3) implementation, that is used to describe product properties. If a formal ontology for PSD properties arises, it will also be used to convert such properties into relations in the Domain layer of the BIM4Ren model.

Extending the approach to geometry would require a deeper work, because the amount of geometry information in BIM is high, and the scalability of such approach cannot be easily ensured. In addition, dealing with geometry in a linked data environment is still a reasearch topic in itself to the authors knowledge. Ideally, such conversion should also imply geometry representation conversion, i.e. bounding boxes, B-REP…

Extending the approach to models used in the industry, in particular gbXML or COBie, is another interesting topic as it would deliver a more impacting use case to the overall AEC industry.

Other alignment methodologies may also be explored to solve BIM issues in the AEC community; for instance the rule-based alignment is manually done through the elaboration of the rules, but other approaches can be tested to discover alignments automatically (patterns discovery, word distance...).

Overall, in a BIM level 3 implementation, our approach could be used to allow stakeholders to use their own vocabularies, which we consider crucial to federate the sector. Such an approach strongly differs from bringing a single unified standard. Other tests could be performed to ensure the scalability of the solution on bigger models, concurrency issues on triple stores when updating the model,…

References

Jakob Beetz, Jos Van Leeuwen, and Bauke De Vries. Ifcowl: A case of transforming express schemas into ontologies. *Ai Edam*, 23(1):89–101, 2009. Cambridge University Press.

Pierre Bourreau, Nathalie Charbel, Jeroen Werbrouck, Madhumitha Senthilvel, Pieter Pauwels, and Jakob Beetz. Multiple inheritance for a modular bim. In Régine Teulier and Sandra Marquès, editors, *Le BIM et l'évolution des pratiques: Ingénierie et architecture, enseignement et recherche*, pages 63–82. Eyrolles, 2020.

CEN. prEN 17632:2021, Semantic Modelling and Linking (SML), 2020. (out for first ballot in March 2021).

Jérôme Euzenat, Pavel Shvaiko, et al. *Ontology matching*, volume 18. Springer, 2007.

Ralph Hodgson, Paul J. Keller, J.ack Hodges, and J.ack Spivak. Qudt-quantities, units, dimensions and data types ontologies. *USA Available http://qudt.org March*, 156, 2014. Online; accessed Oct. 15. 2023.

Yves R. Jean-Mary, E. Patrick Shironoshita, and Mansur R. Kabuka. Ontology matching with semantic verification. *Journal of Web Semantics*, 7(3):235–251, 2009. Elsevier.

Mark Johnson and K.ristine K. Fallon. Experimental building information models. Technical report, Construction Engineering Research Laboratory (US), 2011.

Yannis Kalfoglou and Marco Schorlemmer. Ontology mapping: the state of the art. In *Dagstuhl Seminar Proceedings*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2005.

Ming Mao. Ontology mapping: An information retrieval and interactive activation network based approach. In *The Semantic Web*, pages 931–935. Springer, 2007.

Jyrki Oraskari, Mathias Bonduel, Kris McGlinn, Pieter Pauwels, Freddy Priyatna, Anna Wagner, Ville Kukkonen, Simon Steyskaland, and Joel

Lehtonen. jyrkioraskari/ifctolbd: Ifctolbd 2.5, September 2020. URL https://doi.org/10.5281/zenodo.4056940.

Pieter Pauwels and Walter Terkaj. Express to owl for construction industry: Towards a recommendable and usable ifcowl ontology. *Automation in Construction*, 63:100–133, 2016. Elsevier.

Pieter Pauwels, Walter Terkaj, Thomas Krijnen, and Jakob Beetz. Coping with lists in the ifcowl ontology. In *22nd EG-ICE International Workshop*, pages 113–122, 2015.

Mads Rasmussen, Maxime Lefrançois, Mathias Bonduel, Christian Hviid, and Jan Karlshøj. Opm: An ontology for describing properties that evolve over time. In *6th Linked Data in Architecture and Construction Workshop*, pages 24–33, 2018.

Mads Holten Rasmussen, Maxime Lefrançois, Georg Ferdinand Schneider, and Pieter Pauwels. Bot: the building topology ontology of the w3c linked building data group. *Semantic Web*, 12(1):1–19, 2019. IOS Press.

Georg Ferdinand Schneider. Towards aligning domain ontologies with the building topology ontology. In *Proceedings of the 5th Linked Data in Architecture and Construction Workshop (LDAC 2017)*, 2017.

Pavel Shvaiko. *Iterative schema-based semantic matching*. PhD thesis, University of Trento, 2006.

Léon van Berlo. BuildingSMART tehnical roadmap, 2020. Published on buildingsmart.org, version 30 april 2020.

2.2.7 Axioms

```
@prefix : <https://models.bim4ren.eu/alignment/buildings-IFC#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix dcterms: <http://purl.org/dc/terms/> .
@prefix bot: <https://w3id.org/bot#> .
@prefix sot: <https://models.bim4ren.eu/sot#> .
@prefix ifc4: <https://standards.buildingsmart.org/IFC/DEV/IFC4_1/OWL#> .
@prefix ifc2x3: <https://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#> .

bot:Site owl:equivalentClass ifc4:IfcSite ;
            owl:equivalentClass ifc2x3:IfcSite.
bot:Building owl:equivalentClass ifc4:IfcBuilding ;
                  owl:equivalentClass ifc2x3:IfcBuilding .
bot:Storey owl:equivalentClass ifc4:IfcBuildingStorey  ;
                  owl:equivalentClass ifc2x3:IfcBuildingStorey .
bot:Space owl:equivalentClass ifc4:IfcSpace ;
             owl:equivalentClass ifc2x3:IfcSpace .
bot:Element owl:equivalentClass ifc4:IfcElement ;
             owl:equivalentClass ifc2x3:IfcElement .

ifc2x3:IfcDistributionSystem rdfs:subClassOf sot:System.
ifc4:IfcDistributionSystem rdfs:subClassOf sot:System.

ifc2x3:IfcDistributionElement rdfs:subClassOf sot:SystemComponent.
ifc4:IfcDistributionElement rdfs:subClassOf sot:SystemComponent.

ifc2x3:IfcDistributionPort rdfs:subClassOf sot:TransportElement.
ifc4:IfcDistributionPort rdfs:subClassOf sot:TransportElement.

ifc2x3:IfcFlowTerminal rdfs:subClassOf sot:TerminalElement.
ifc4:IfcFlowTerminal rdfs:subClassOf sot:TerminalElement.
```

2.2.8 Extract of the hybrid rules used in the experiments

```
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix list: <https://w3id.org/list#> .
@prefix bot: <https://w3id.org/bot#> .
@prefix sot: <https://models.bim4ren.eu/sot#> .
@prefix b4r-building: <https://models.bim4ren.eu/buildings#> .
@prefix distribution: <https://pi.pauwel.be/voc/distributionelement#>.
@prefix building: <https://pi.pauwel.be/voc/buildingelement#>.
@prefix b4r: <https://models.bim4ren.eu#> .
@prefix ifc: <https://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#> .

[equivalent1: (?a owl:equivalentClass ?b) -> (?a rdfs:subClassOf ?b) (?b
rdfs:subClassOf ?a)]

-> table(b4r:subClassOf).
[transSubClassZone:  -> (bot:Zone b4r:subClassOf bot:Zone)]
[transSubClassSystemComponent:  -> (sot:SystemComponent b4r:subClassOf
sot:SystemComponent)]
[transSubClassSystem:  -> (sot:System b4r:subClassOf sot:System)]
[transSubClass1: (?cls1 b4r:subClassOf ?cls2) <- (?cls1 rdfs:subClassOf ?cls2)]
[transSubClass2: (?cls1 b4r:subClassOf ?cls3) <- (?cls1 rdfs:subClassOf ?cls2)
(?cls2 b4r:subClassOf ?cls3)]

-> table(list:isIn).
[isIn1: (?elt list:isIn ?l) <- (?l list:hasContents ?elt)]
[isIn2: (?elt list:isIn ?l) <- (?l list:hasNext ?queue) (?elt list:isIn ?queue)]

[botHasElement-IFC: (?z bot:hasElement ?p) <-
 (?c ifc:relatingStructure_IfcRelContainedInSpatialStructure ?z)
 (?c ifc:relatedElements_IfcRelContainedInSpatialStructure ?p)
 (?z rdf:type ?cl) (?cl b4r:subClassOf bot:Zone)]

[relHasBuilding-IFC2x3: (?z1 bot:hasBuilding ?z2) <-
 (?rel ifc2x3:relatingObject_IfcRelDecomposes ?z1)
 (?rel ifc2x3:relatedObjects_IfcRelDecomposes ?z2)
 (?z1 rdf:type ?cl1) (?cl1 b4r:subClassOf bot:Zone)
 (?z2 rdf:type ?cl2) (?cl2 b4r:subClassOf bot:Building)]
[relHasStorey-IFC2x3: (?z1 bot:hasStorey ?z2) <-
 (?rel ifc2x3:relatingObject_IfcRelDecomposes ?z1)
 (?rel ifc2x3:relatedObjects_IfcRelDecomposes ?z2)
 (?z1 rdf:type ?cl1) (?cl1 b4r:subClassOf bot:Zone)
 (?z2 rdf:type ?cl2) (?cl2 b4r:subClassOf bot:Storey)]
[relHasSpace-IFC2x3: (?z1 bot:hasSpace ?z2) <-
 (?rel ifc2x3:relatingObject_IfcRelDecomposes ?z1)
 (?rel ifc2x3:relatedObjects_IfcRelDecomposes ?z2)
 (?z1 rdf:type ?cl1) (?cl1 b4r:subClassOf bot:Zone)
 (?z2 rdf:type ?cl2) (?cl2 b4r:subClassOf bot:Space)]

[relContainsZone-IFC4: (?z1 bot:containsZone ?z2) <-
 (?rel ifc4:relatingObject_IfcRelAggregates ?z1)
 (?rel ifc4:relatedObjects_IfcRelAggregates ?z2)
```

```
 (?z1 rdf:type ?cl1) (?cl1 b4r:subClassOf bot:Zone)
 (?z2 rdf:type ?cl2) (?cl2 b4r:subClassOf bot:Zone)]


[connectedWith-IFC: (?elt1 sot:connectedWith ?elt2) <-
 (?c ifc:relatingPort_IfcRelConnectsPorts ?p1) (?c
ifc:relatedPort_IfcRelConnectsPorts ?p2)
 (?elt1 rdf:type ?cls1) (?cls1 b4r:subClassOf sot:SystemComponent)
 (?elt2 rdf:type ?cls2) (?cls2 b4r:subClassOf sot:SystemComponent)
 (?n1 ifc:relatingObject_IfcRelNests ?elt1) (?n1 ifc:relatedObjects_IfcRelNests
?list1) (?p1 list:isIn ?list1)
 (?n2 ifc:relatingObject_IfcRelNests ?elt2) (?n2 ifc:relatedObjects_IfcRelNests
?list2) (?p2 list:isIn ?list2)]

[sotHasElements-IFC: (?s sot:hasElements ?elt) <-
 (?c ifc:relatingGroup_IfcRelAssignsToGroup ?s)
 (?c ifc:relatedObjects_IfcRelAssigns ?elt)
 (?s rdf:type ?cls) (?cls b4r:subClassOf sot:System)
 (?elt rdf:type ?clse) (?clse b4r:subClassOf sot:SystemComponent)]


[ifcToLbdAirTerminalBox-USERDEFINED:(?z rdf:type ifc:IfcAirTerminalBox) (?z
ifc:predefinedType_AirTerminalBox ifc:USERDEFINED) -> (?z rdf:type
distribution:AirTerminalBox-USERDEFINED) .
[ifcToLbdAirTerminalBox-VARIABLEFLOWPRESSUREDEPENDANT:(?z rdf:type
ifc:IfcAirTerminalBox) (?z ifc:predefinedType_AirTerminalBox
ifc:VARIABLEFLOWPRESSUREDEPENDANT) -> (?z rdf:type distribution:AirTerminalBox-
VARIABLEFLOWPRESSUREDEPENDANT) .
[ifcToLbdAirTerminalBox-VARIABLEFLOWPRESSUREINDEPENDANT:(?z rdf:type
ifc:IfcAirTerminalBox) (?z ifc:predefinedType_AirTerminalBox
ifc:VARIABLEFLOWPRESSUREINDEPENDANT) -> (?z rdf:type distribution:AirTerminalBox-
VARIABLEFLOWPRESSUREINDEPENDANT) .
[ifcToLbdAirTerminalBox-CONSTANTFLOW:(?z rdf:type ifc:IfcAirTerminalBox) (?z
ifc:predefinedType_AirTerminalBox ifc:CONSTANTFLOW) -> (?z rdf:type
distribution:AirTerminalBox-CONSTANTFLOW) .
[ifcToLbdAirTerminalBox-NOTDEFINED:(?z rdf:type ifc:IfcAirTerminalBox) (?z
ifc:predefinedType_AirTerminalBox ifc:NOTDEFINED) -> (?z rdf:type
distribution:AirTerminalBox-NOTDEFINED) .

[ifcToLbdValve-ANTIVACUUM:(?z rdf:type ifc:IfcValve) (?z ifc:predefinedType_Valve
ifc:ANTIVACUUM) -> (?z rdf:type distribution:Valve-ANTIVACUUM) .


….
```

2.3 SHACL is for LBD what mvdXML is for IFC (CIB W78 2021)

Image a scenario in which digital engineering documents would be handed over to a facility manager. The data are converted to Linked Data, but the amount of data is vast and contains partly unnecessary material. Thus, a facility management model view of the BIM data, i.e., only the required part of the content, is extracted from the data to be aggregated into the knowledge base of the facility manager.

Hypothesis 2.3: The claim was that unit tests [Hunt and Thomas, 2003] can be written to validate that the RDF converter in the LBD context is doing what the writer of the software thinks it should do. It also should test that the implementation follows the ideas of the persons that have defined alignments between the input and the output presentations of the data.

If the unit testing has good expressiveness and coverage in the problem domain, we can assume that we have some certainty that the tests express the idea that the programmer has. Three different approaches were addressed. Firstly, a change management of the source code scenario was used where the output was compared to a pre-established correct output. The proposed method was shown to be better than the earlier bitwise file comparison approach. The alignment relation was then verified to hold throughout the conversion process. Finally, the output was compared to an example that represented the ground truth of the correct output. Furthermore, I covered the theoretical aspects of the conversion process and identified crucial points that needed to be checked. This reflects the coverage of the tests in the problem domain, considering the correctness aspect. Consequently, by using the tests presented in the article, we can determine whether the implementation has met the formal requirements for the converter presented as alignments.

The general assumption is that testing improves code

quality, and unit testing is a good practice. Relation to the code quality can be more complex, as Athanasiou et al. [2014] and Chioteli et al. [2021] have argued. The proof is left for future work.

Hypothesis 2.4 claimed that it is possible to implement Model View Definitions (MVD)[28] filtering for LBD models. Linked building data can be filtered using Shapes Constraint Language (SHACL)[29] rules for specific BIM use cases. It was left for future work to prove that the listed buildingSMART MVD database definitions[30], or generic mvdXML[31] definitions can be converted to SHACL rules. The filtering was possible for the level of complexity 1 (the levels are presented in Section 2.1). As the filtered content is rewritten using SPARQL CONSTRUCT, the knowledge gap is in implementing more complex SPARQL CONSTRUCT queries. There are no known limitations in filtering itself.

Table 2.8 summarizes the use of the design science research method in this research.

My contribution to the paper: I wrote most of the manuscript and research. My contribution focus points were in the related work section in the Linked Building Data and Linked Building Data Converter sections. In addition, I wrote about the theoretical framework of unit texting of the conversion, the conversion rules, the MVD part,s and the discussion. Madhumitha Senthilvel connected the work to the BIM4Ren[32] project context in the introduction section and contributed as a Shapes Constraint Language (SHACL) expert. The conclusions were made together.

[28] https://technical.buildingsmart.org/standards/ifc/mvd/ (Accessed June 13, 2023)
[29] https://www.w3.org/TR/shacl/ (Accessed June 13, 2023)
[30] https://technical.buildingsmart.org/standards/ifc/mvd/mvd-database/ (Accessed June 13, 2023)
[31] https://technical.buildingsmart.org/standards/ifc/mvd/mvdxml/ (Accessed June 13, 2023)
[32] https://bim4ren.eu/

Table 2.8. The work of Oraskari et al. [2021b] in the Hevner et al. [2004] design science research framework

Guideline 1: Design as an artifact
The article presents W3C Shapes Constraint Language (SHACL) rules that can be used to test the correctness of the output of an IFCtoLBD converter. This work was a proof of the construction of an MVD filter for LBD ABox content. They represent both sides of the buildingSMART Model View Definitions concept: checking and filtering were covered. The ideas are presented so they can be implemented using other tools and languages.
Guideline 2: Problem relevance
Here there are two business problems: (1) When converting data from an ontology or a concept model to another, the information is often interpreted. For the user of the output data, how to manage that the information is not distorted and the selected information is not lost. So, the business problem is, in short, how to ensure the quality of the conversion. (2) The LBD models can be vast and extra information may make it harder to query the data. The question is, can the user be offered a split containing only the information needed for the application? The unstudied area was how to guarantee that a converter produces output as RDF that represent the native content impeccably. This work focuses on that, but also, in general, how to write unit tests for the converter. Furthermore, the other aspect of MVD, filtering, had not been studied in the LBD context earlier.

The constituent user community of the artefact is the researchers of the Linked Building Data Community, and particularly the developers of converters or MVD filters.
The solution belongs to the Model View Definition, unit testing, and validation problem classes.
This work is an enabler for the following BIM use case: Model-based quantity take-off and Cost Calculation (Building Design and Construction) (buildingSMART use-case GUID: A25D8756-90B4-4322-97AD-DCE04AC695BF [buildingSMART International, 2022]) in the LBD domain.
The focus of the article is mainly on the converter quality. Therefore these shared use cases of the converter are presented in Section 2.1.

Guideline 3: Design evaluation

Hevner et al. Design Research Evaluation Method:
A descriptive Scenario was used to evaluate the utility of the MVD filter.
The work is a feasibility analysis, but the presented method was compared with the earlier file comparison approach. The methods were discussed in the article. In Controlled Experiments, the rules were tested using TopQuadrant TopBraid SHACL API[33].
Generalization of the results: The presented unit test conversion rules and methods can easily be adapted to other aligned ontology pairs. Also, test by example can use examples of another domain.
The presented MVD filter is an example of the filtering functionality. It represents a group of possible MVD filters. It was left for future work to show that more complex filters can be implemented.

[33] https://github.com/TopQuadrant/shacl (Accessed June 13, 2023)

Guideline 4: Research contributions

SHACL is a language to test the validity of RDF graph content. Earlier work was shown that SHACL could be used to test LBD graphs.
However, LBD has been used as static exports of native models (for example, Revit models) in which the model is managed.
The research showed methods that can be used to test an IFCtoLBD converter. It applies to any implementation as it focuses on the output created. These are improvements when compared with the file content matching approach.
Also, a method to filter LBD content in the sense of MVD was shown. Future work is needed to study more complex filters. To my knowledge, there have not been MVD filters for LBD data earlier.

Guideline 5: Research rigor

The work was peer-reviewed. The article contains the presented SHACL rules, and the complete source code was published at a public GitHub repository[34].

Guideline 6: Design as a search process

Various ways to implement unit tests were presented. The coverage of the tests was also discussed.
The known SHACL validator implementations were TopBraid SHACL API [35], Apache SHACL, and pySHACL[36]. The first two were written in Java which made it possible to integrate them into the existing IFCtoLBD [Oraskari et al., 2021a] Maven toolchain. TopBraid SHACL API was used as it was well known.
SHACL was selected instead of ShEx since SHACL was well-known in the LBD community.

[34] https://github.com/jyrkioraskari/SHACL_UNITTESTS4LBD
[35] https://github.com/TopQuadrant/shacl
[36] https://github.com/RDFLib/pySHACL (Accessed June 13, 2023)

Guideline 7: Communication of research

Proceedings of the 38th International Conference of CIB W78, Luxembourg, 13-15 October

The authors have kept right for the original work, so it can be re-published here. The work was initially published in [Oraskari et al., 2021b]. The content presented here is based on our final draft of the article.

Jyrki Oraskari Department of Design Computation, RWTH Aachen University, Aachen, Germany, Jyrki.Oraskari@dc.rwth-aachen.de
Madhumitha Senthilvel Department of Design Computation, RWTH Aachen University, Aachen, Germany, senthilvel@dc.rwth-aachen.de
Jakob Beetz Department of Design Computation, RWTH Aachen University, Aachen, Germany, j.beetz@caad.arch.rwth-aachen.de

Abstract: The motivation for this work stems from an EU funded project, which focuses on leveraging digital tools for improving the renovation processes. In particular, certain tools require LinkedIn Building Data (LBD) that need to fulfil the application-specific exchange requirements. In this research, we focus on two different use cases to investigate how to validate a Linked Building Data model. First, we study how to minimize data loss and errors when data is converted and brought into an LBD data store. The usage of unit tests to improve conversion quality is introduced. The second use case focuses on how Model View Definition (MVD)in LBD for evaluating the energy performance of the renovation designs in energy simulation can be formed.

keywords: IFC, MVD, LBD, SHACL, OWL, RDF, unit test

2.3.1 Introduction

Information management of the stakeholders' data is one of the critical challenges in the construction industry. Linked-data is an approach that has been proposed to lower the barriers of data silos by using web technologies, Resource Description Framework (RDF), and recommended best practices for expressing and sharing information.

The Linked Data for architecture, engineering and construction (AEC) applies the relevant vocabularies and structured interlinked data to publish content in a machine-interpretable form that can be used for semantic queries. The new model presentation can be used for model checking, machine inferencing, for lossless data exchange, and as an extendable interlingua for the AEC programs, i.e., support interoperability between software applications. [Pauwels et al., 2017]

The motivation for this work stems from an EU funded project which focuses on leveraging digital tools for improving the renovation processes. As a part of

this project, various workflows for chaining of tools used for common renovation specific use cases were developed, out of which it was identified that the tools in such a work flow need to be supplied with the appropriate data necessary for them to function seamlessly. In particular, certain tools required the use of Linked Building Data (LBD) inputs, which were arrived at from IFC files through usage of BOT-based converters.

In this research, we focus on two different use cases to investigate how to validate a Linked Building Data model. First, we study how to minimize data loss and errors when data is converted into LBD. How unit tests can improve the quality of the Linked Data when it is bought into the system. The second use case focuses on how Model View Definition in the context of the LBD data for evaluating the energy performance of the renovation designs in energy simulation can be formed. What can be tested using SHACL, and what are the benefits of SHACL compared with mvdXML for IFC models? There have been studies on validating LBD data earlier. Here we focus on how to create a validated MVD view of LBD data using SHACL.

In the next chapters, we first review the Linked Data in AEC domain, how the data has been converted, and validated. Shapes Constraint Language (SHACL) is presented and and how it has been applied on LBD data. In Chapters 2.3.3 and 2.3.4, we introduce our selected use cases, and define the criteria for a valid LBD Abox model in the context. How the requirements can be tested using SHACL are analysed. Finally, the findings are discussed and conclusions summarised.

2.3.2 Related Work

Linked Building Data

Over the last decade and a half, several linked building data ontologies have been proposed in the Architecture, Engineering, Construction, Owner Operator (AECOO) sector. Web Ontology Language (OWL) [McGuinness et al., 2004] ontology for IFC (ifcOWL) [Schevers and Drogemuller, 2005, Beetz et al., 2009, Pauwels and Terkaj, 2016] ontology is a standard ontology for BIM models designed to be compatible with the IFC standard, developed by buildingSMART and aimed to increase interoperability by sharing common data schema and exchange format. ifcOWL is as faithful as possible and almost a literal conversion of the IFC Express schemas, which makes ABox models large and complex. On the other hand, ifcWoD [Mendes de Farias et al., 2015, Djuedja and Flore, 2019] was developed to express the object-oriented constraints of the IFC schema and the semantics of the model as an OWL ontology. It is as tightly coupled with the IFC

versions as ifcOWL is. Like ifcWoD, SimpleBIM [Pauwels and Roxin, 2016] was an attempt to create a simpler and more developer-friendly view on ifcOWL data.

There were also other ontologies. The Domotic OSGi Gateway ONTology (DOGONT) [Bonino and Corno, 2008] was originally developed to express home automation devices. In the last years, the scope has been widened to cover IoT network components.

COBieOWL [Farias et al., 2015] is an approach to present Construction Operations Building Information Exchange (COBie) standard sheets as OWL classes, and sheet columns as properties. The OWL Ontology has been automatically populated from the template using Java-based COBieOWL tool.

BIM Shared Ontology (BIMSO) and BIM Design Ontology (BIMDO) are modular ontologies that are independent of the IFC standard. BIMSO is build on the UNIFORMAT II classification system, and BIMDO has its own vocabulary [Niknam and Karshenas, 2017, Rasmussen et al., 2019a]

W3C Linked Building Data Community Group (W3C LBD- CG)[37] proposed a new modular approach, where a AEC ontology, the Building Topology Ontology (BOT) [Rasmussen et al., 2017b], would not violate the W3C best practices as the previous ontologies had done. For example, its design principle has been to reuse concepts in other more focused ontologies. Unlike the voluminous ifcOWL, BOT is following the best practice of keeping ontologies simple for easy maintenance.

Notable ontologies that have been used besides the BOT are Ontology for Property Management (OPM)[38], and Building Product Ontology (BPO) [Wagner and Rüppel, 2019] that allows manufacturers to define their products.

There are also domain spesific ontology suites with alignments to other LBD ontologies. Digital Construction Ontologies (DiCon) [Törmä and Zheng, 2021], developed in Diction and BIM4EEB projects, is a modularized ontology group for expressing information related to the execution of construction projects, while the BIM4Ren ontologies[39] are for modelling energy-efficient renovations.

[37] https://www.w3.org/community/lbd/
[38] https://github.com/w3c-lbd-cg/opm
[39] https://models.bim4ren.eu/

LBD Converters

There have been various tools to translate BIM data models into LBD. One of the early ones was IFCtoRDF [P.auwels and Oraskari, 2016] to translate IFC STEP documents onto ifcOWL ABox format. IFCtoLBD converter is a Java-based converter developed in the LinkedIn Building Data community. This tool extracts instances of core BOT ontology classes and their relationships, but also, product data, and property values expressed using the OPM ontology are present. [Rasmussen et al., 2019b, Bonduel et al., 2018, Oraskari et al., 2020]. There are many branches of the code. One of the most notable ones is IFCtoLBD from Pieter Pauwels.

A plug-in that exports Linked Building Data from Autodesk Revit[40] has been initially developed by Jonas Eik Bacher-Jacobsen at NIRAS and continued by Rasmussen et al. [Rasmussen et al., 2017a]. It is written using the .NET developer platform. In addition to the core LBD data, the tool can also export 3D spaces, OBJ encoded mesh geometry of elements and outlines of areas as WKT polygons.

On the other hand, NIRAS IFC2BOT [41] and written by Mads Rasmussen is a lightweight command-line tool written in Python 3.8. It extracts core BOT elements of an IFC document using the IfcOpenShell[42] Python library.

Furthermore, in [Bourreau and Oraskari, 2021b], Bourreau and Oraskari propose a novel dynamic translation method where rule-based reasoning is used for the translation

Shapes Constraint Language (SHACL)

SHACL is a general purpose data validation language and specification from the World Wide Web Consortium (W3C) which focuses on conformance checking of information serialized as Linked Data [Knublauch and Kontokostas, 2017]. It contains two major components: Shapes graph and Data Graph. The former contains a list of user-defined constraints which are syntactically represented in SHACL language, while the latter is the RDF graph which is being validated against the Shapes graph. SHACL constraints can be used to define restrictions on the values that a property can have, the datatype of the property, numerical ranges of values, absolute string matches and also a mixture of the above. More complex

[40] https://github.com/MadsHolten/revit-bot-exporter
[41] https://github.com/NIRAS-MHRA/IFC2BOT
[42] http://ifcopenshell.org/python

constraints such as sub-graph pattern validation, conditional checking can be also expressed using SPARQL queries inside SHACL shapes, or through the use of SHACL JavaScript Functions. SHACL is a data-agnostic constraint modelling language, i.e. regardless of the underlying schema of the data, as long as the data is serialized as RDF, SHACL validation rules can be used for it's conformance checks. Consequently, in the Linked Data and AEC domain, information spans multiple domains and contains interlinks between them, SHACL is emerging as candidate for complex semantic constraint checking.

Hagedorn and König have examined the feasibility of the rule validation approach in the AEC industry. A fictive tunnel construction project was used as a use case [Hagedorn and König, 2020], while in [Stolk and McGlinn, 2020], Stolk and McGlinn studied the possibility of using SHACL for testing ifcOWL models. Furthermore, in [Werbrouck et al., 2019], Werbrouck et al. described how SHACL can be used like mvdXML for LBD data, while SHACL-based dynamic constraint solving has been explored for look-ahead-planning [Soman, 2019].

2.3.3 Case:The Converter Unit Testing

When a converter program is designed and written, the process is prone to human errors caused by misinterpretations of the concepts, and software bugs. Additionally, there may also be concepts that are not fully mappable.

Unit testing is a software development method to avoid unwanted errors in code when introducing new features. A collection of unit tests, where each tests a small behavioural aspect of the software, can be used to improve the code quality. When developing a program that contains complex rules to convert source data into an instance data (ABox) model that comply with the used ontologies, the output-based unit tests can be used. The conversion process is stateless, i.e., the method's prior calls do not change the module's internal state.

To define the requirements for the unit tests, we started with questions such as: How to define a sound and valid conversion output? What are the characteristics of a good set of unit test for a translator tool?

In [Kalfoglou and Schorlemmer, 2003], Kalfoglou et. al. define ontology mapping to be a way to relate two vocabularies of the same domain in such a way that the mathematical structure of ontological signatures and their intended interpretations, as specified by the ontological axioms, are respected.

A data translator instantiates a partial ontology mapping form for individuals

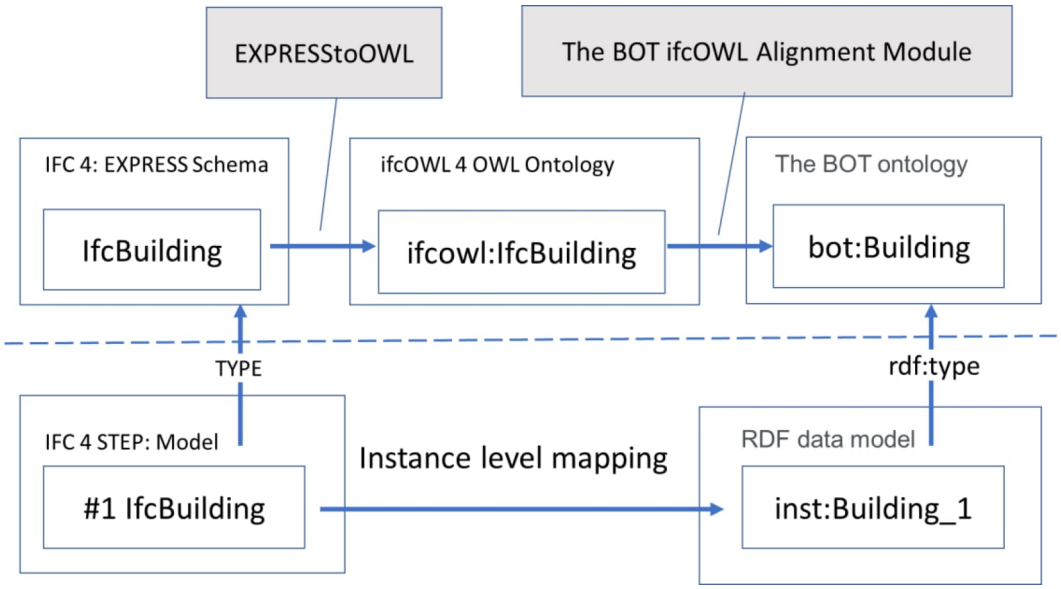


Fig. 2.7. The schema mapping from IFC to BOT.

[Euzenat et al., 2007, Kalfoglou and Schorlemmer, 2003]. The mapping defines the correspondence between entities in respective ontologies and is an essential part of the translator. A valid output consists of individuals with a one-to-one correspondence with an element in the input model. All elements in the input model that have a mapping have a counterpart in the output.Therefore the first requirement R1 is that the unit tests validates the ontology mapping realisation defined in the alignments.

For example, when an IFC document is translated into assertions that use Building Topology Ontology, and there is an IfcBuilding element in the IFC model, IFC EXPRESSS to OWL [Pauwels and Terkaj, 2016] schema mapping defines that there is a ifcowl:IfcBuilding counterpart for that element. There is also an ifcOWL Alignment Module in BOT that defines that all ifcowl:IfcBuilding individuals are also individuals of the bot:Building class. Therefore, a unit test should check that whenever there is a IfcBuilding element in the IFC model, the output should contain one corresponding Building individual. This is illustrated in Figure 2.7.

In addition to class correspondence, in schema mapping, there are relationship path correspondence, i.e., the corresponding classes are similar in relations and so the graphs are isomorphic.

Requirement R1 in SHACL

We need to express the rule for all input graph individuals of an ifcOWL class that is part of an alignment in the used ontologies. There needs to be precisely one individual representing the same concept in the generated Linked Building Data model. In general, testing that there is a bijection between the models can be computation-intensive work, but in particular, in the IFC context, IFC GUIDs can be used. The problem can be expressed: For all instances with an alignment and a GUID attribute, there has to be exactly one instance counterpart with the same GUID in the LBD graph. In SHACL this can be written using SHACL-SPARQL extension as shown in Listing 2.1.

```
inst:shapeBOTClasses a sh:NodeShape ;
  sh:targetClass ifc:IfcSite, ifc:IfcBuilding, ifc:Store;
  sh:property [
     sh:sparql [
       a sh:SPARQLConstraint ;
       sh:message "GUIDs of the aligment instances should match." ;
       sh:prefixes inst:prefixes;
       sh:select """
        SELECT $this (?guid as ?value){
          $this ifc:globalId_IfcRoot/express:hasString ?guid .
          FILTER NOT EXISTS { ?b props:globalIdIfcRoot_attribute_simple ?guid }
          }
       """
       ]
] .
```

Listing 2.1. SHACL rule for checking BOT class alignment

The alignment rule (Listing 2.1).) was tested using the IFCtoLBD [Bonduel et al., 2018] converter and TopBraid SHACL API[43]. In the test, the rule could validate that the aligned instances existed in the output model. The rule is IFC version and OPM level [Bonduel et al., 2018] specific. Similar rules can be written for the other combinations also.

Requirement R2 in SHACL

To validate a converter against R2, an approach where a generic one rule that fits all OWL alignments is not plausible. E.g., owl:equivalentProperty used for property

[43] https://github.com/TopQuadrant/shacl

alignment has rdf:Property as both its domain and range. Therefore it cannot be used for property paths needed for ifcOWL to BOT alignment. Specific test to test the existence of an assertion can be written like shown in Listing 2.2.

```
inst:shapeBOTRelations a sh:NodeShape ;
 sh:targetClass ifc:IfcSite;
 sh:property [
 sh:sparql [
  a sh:SPARQLConstraint ;
   sh:message "bot:hasBuilding missing." ;
   sh:prefixes inst:prefixes;
   sh:select """
     SELECT $this (?guid as ?value){
       $this ifc:globalId_IfcRoot/express:hasString ?guid1 .
       $this a ifc:IfcSite .

       $ifc_building ifc:globalId_IfcRoot/express:hasString ?guid2 .
       $ifc_building a ifc:IfcBuilding .

       ?bot_site props:globalIdIfcRoot_attribute_simple ?guid1 .
       ?bot_building props:globalIdIfcRoot_attribute_simple ?guid2 .
        FILTER NOT EXISTS { ?bot_site bot:hasBuilding ?bot_building }
        }
        """
    ]

 ] .
```

Listing 2.2. SHACL rule for checking BOT relations alignment.

On the other hand, since the input models are part of the unit tests and thus unchanging, and the primary function is to track whether a change in the source code affects the output negatively, it suffices to validate that the known positive characteristics of the output model are kept. In IFCtoRDF this has been tested comparing output with an earlier result bytewise, which causes false-negative warnings, e.g. it did, when Apache Jena library was updated to version 3.16.0 and white spacing was changed.

One approach to avoid oversensitive warnings and be able to give element level error messages is to use checksums of the ordered set of triples connected to the elements or just the properties. This was tested in a Java unit test and the TopQuadran SHACL engine. Listing 2.3 shows the SHACL code. In tests, it was able to warn when there was an unknown change in the connected triple set.

```
inst:shapeBOTChecksum a sh:NodeShape ;
 sh:targetNode inst:stairflight_ca457005-aa0d-4679-92c1-5067d702c9f3;
  sh:property [
   sh:sparql [
     a sh:SPARQLConstraint ;
     sh:message "The checksum of the properties is not valid." ;
     sh:prefixes inst:prefixes;
     sh:select """
     SELECT $this ?ResultSetHash WHERE{
      {
          SELECT $this (MD5(GROUP_CONCAT(CONCAT(STR(?p));
              separator=' ')) as ?hash ) WHERE{
               SELECT * WHERE {
                  $this ?p ?o.
               } ORDER BY ?s ?p ?o
        } GROUP BY $this
      }
      FILTER (?hash != "82c7dc90fcb57319f2bb7ead58ead1de")
     }
     """
    ]
 .
```

Listing 2.3. SHACL rule to check checksum of properties

Also, SHACL validation patterns can be generated programmatically from sample files to get a detailed report for what has changed in the output. The output can be compared with one of an earlier release of the tool and to check that it is consistent with the other available LBD converters to compare the outputs, and so, the interpretation of the ontology usage is in line. To test this, we used BOT Duplex Apartment[44] that had been exported by using Niras Autodesk Revit LBD Plugin Exporter [Rasmussen et al., 2017a]. It is an independent implementation and both converters used the same IFC model.

The challenge was to match the individuals as the base URIs were different, the URI format differs, and the attribute properties are formed differently. The IFC GUID encoding was in the Revit plugin using URL encoding. In contrast, IFCtoLBD uses the uncompressed hexadecimal notation of the IFC GUID by Jan Tulke et al.

A program to write auto-generated SHACL rules to check that the graph of both of the converter matched in topological level was written. For this, IFC GUIDs were used to associate the individuals, and the BOT namespace was used so that

[44] https://github.com/MadsHolten/BOT-Duplex-house

while going through the sample graph using the depth-first search algorithm, corresponding SHACL rules were exposed. Listing 2.4 shows a snippet of the long auto-generated SHACL rules.

```
inst:shape_1
 a sh:NodeShape ;
 sh:targetNode
  inst:building_7b7032cc-b822-417b-9aea-642906a29bd4 .
 sh:property
  [ sh:hasValue
      inst:storey_7b7032cc-b822-417b-9aea-6429f95d6512 ;
    sh:minCount 1 ;
    sh:path
     bot:hasStorey
  ] ;
 <--- cut here --->
 sh:property
  [ sh:hasValue
      inst:storey_7b7032cc-b822-417b-9aea-6429f95f770e ;
    sh:minCount 1 ;
    sh:path
     bot:hasStorey
  ] ;
```

Listing 2.4. Sample auto-generated SHACL rule.

The generated SHACL rule set was used in a unit test to validate the conversion functionality. It was shown that sample models can be used to create SHACL rules for unit tests. They can be used to compare output with an earlier saved and manually tested data model and, as here, to compare the result with one the has been created using an independent tool. To generalize this work is left for future studies.

2.3.4 Case: Energy Simulation MVD use case

buildingSMART International has defined a Model View Definition (MVD) as a subset of IFC schema for a specific use case. Provided standard MVDs are, e.g. Coordination View, Structural Analysis View, Quantity Takeoff View, and Energy Analysis View [Beetz et al., 2018]. mvdXML [Chipman et al., 2012, Weise et al., 2017] is an Extensible Markup Language (XML) format to express MVD definitions that can be used for creating IFC import and export filters for BIM software stating validation rules and validating data.

mvdXML definition consists of two parts. Templates section defines reusable templates to instantiate a set of attribute values in a given IFC entity context. ModelView defines exchange requirements, i.e., the applicability (ApplicableRootEntity and Applicability TemplateRules) of an IFC entity, the used templates, and associated template rules. In SHACL, instances can be selected using rich set of definitions, like sh:targetClass, sh:targetNode, sh:targetSubjectsOf, sh:targetObjectsOf sh:targetObjectsOf, sh:SPARQLTargetType, and
sh:SPARQLTarget. Also, path patterns can be reused, but as SHACL focused on the violation errors [Labra-Gayo et al., 2019], they are checking rules, while in mvdXML the instantiated set of attributes can be shared in rules. Both can be used for creating an MVD of a mode.

In an energy simulation use case, the critical aspects of an MVD view are that there should be geometry associated with the selected elements, and they should have properties that are important for the energy simulation. [Werbrouck et al., 2019] showed that the checking is possible. In this work, we continue that and show that it is also possible to filter a model view from the LBD data. In Listing 2.5, is shown how to filter out windows elements with an associated 3D geometry, and in Listing 2.6, windows of a specific size are listed.

```
inst:HasGeometry
        sh:path bot:hasSimple3DModel ;
      sh:minCount 1 .

inst:RuleFilterWindowsWithGeometryShape
    a sh:NodeShape ;
    sh:targetClass bot:Element ;
    sh:property inst:HasGeometry ;
    sh:rule [
        a sh:SPARQLRule;
        rdfs:label "Construct a Geometry MVD";
        sh:prefixes inst:prefixes ;
        sh:construct """
            CONSTRUCT {
                $this ?p ?o .
            }
            WHERE {
                $this ?p ?o .
                FILTER (regex (STR($this),"window_")).
            }
            """
    ] .
```

Listing 2.5. Sample SHACL Filter for Window elements..

```
inst:RuleFilterShape
   a sh:NodeShape ;
   sh:targetClass beo:Window ;
   sh:rule [
      a sh:SPARQLRule;
      rdfs:label "Constructs an MVD";
      sh:prefixes inst:prefixes ;
      sh:construct """
            CONSTRUCT {
               $this ?p ?o .
            }
            WHERE {
              $this props:objectTypeIfcObject_attribute_simple "2800mm x 2410mm" .
              $this ?p ?o .
            }
            """
] .
```

Listing 2.6. Sample SHACL Filter for Window elements..

2.3.5 Conclusion and Future work

The explorations in this paper is a part of the big picture in the BIM4Ren workflow where conformance checking plays an essential role.Such exploration can also be leveraged in other projects which also require conformance checking of LBD data.

Besides the MVD checking and filtering of the LinkedIn Building Data model, unit tests in a converter have a role in ensuring no data loss or corruption in the translation process. So far, there havent been unit tests that focus on the ontology translation process of an LBD model at the semantic level. In this feasibility study, we have shown that unit test can be written for the key aspects of the conversion. Although specific, the results can easily be generalized.

In the last decade, one of the big problems in LBD domain has been to manage the vast about of building related and the difficult to separate the wheat from the chaff. This work has shown that besides validation, model views can be created for LBD data using SHACL.

2.3.6 Acknowledgments

This research is part of the BIM4Ren project which has received funding from the European Unions Horizon H2020 research and innovation programme under grant agreement No 820773.

References

Jakob Beetz, Jos van Leeuwen, and de Vries. IfcOWL: A case of transforming EXPRESS schemas into ontologies. *AI EDAM*, 23:89–101, February 2009. doi: 10.1017/S0890060409000122.

Jakob Beetz, André Borrmann, and Matthias Weise. Process-based definition of model content. In *Building Information Modeling*, pages 127–138. Springer, 2018.

Mathias Bonduel, Jyrki Oraskari, Pieter Pauwels, Maarten Vergauwen, and Ralf Klein. The ifc to linked building data converter-current status. In *6th Linked Data in Architecture and Construction Workshop*, volume 2159, pages 34–43, 2018.

Dario Bonino and Fulvio Corno. Dogont-ontology modeling for intelligent domotic environments. In *International Semantic Web Conference*, pages 790–803. Springer, 2008.

Pierre Bourreau and Jyrki Oraskari. Bim format conversion as inference-based ontology alignment. In *Proceedings of the 9th Linked Data in Architecture and Construction Workshop - LDAC2021, CEUR-WS Team, Redaktion Sun SITE, Informatik V, RWTH Aachen, Germany.*, 2021. URL https://ceur-ws.org/Vol-3081/11paper.pdf. Online; accessed Oct. 15. 2023.

Tim Chipman, Thomas Liebich, and Matthias Weise. mvdxml, 2012.

Tchouanguem Djuedja and Justine Flore. *Information modelling for the development of sustainable construction (MINDOC)*. PhD thesis, Toulouse, INPT, 2019.

Jérôme Euzenat, Pavel Shvaiko, et al. *Ontology matching*, volume 18. Springer, 2007.

M. T. Farias, Ana Roxin, and Christophe Nicolle. Cobieowl, an owl ontology based on cobie standard. In *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"*, pages 361–377. Springer, 2015.

Philipp Hagedorn and Markus König. Rule-based semantic validation for standardized linked building models. In *International Conference on Computing in Civil and Building Engineering*, pages 772–787. Springer, 2020.

Yannis Kalfoglou and Marco Schorlemmer. Ontology mapping: the state of the art. *The knowledge engineering review*, 18(1):1–31, 2003. Cambridge University Press.

Holger Knublauch and Dimitris Kontokostas. Shapes constraint language (shacl), 2017. URL https://www.w3.org/TR/shacl/. Online; accessed Oct. 15. 2023.

Jose Emilio Labra-Gayo, Herminio García-González, Daniel Fernández-Alvarez, and Eric Prudhommeaux. Challenges in rdf validation. In *Current Trends in Semantic Web Technologies: Theory and Practice*, pages 121–151. Springer, 2019.

Deborah L. McGuinness, Frank Van Harmelen, et al. Owl web ontology language overview. *W3C recommendation*, 10(10):2004, 2004.

Tarcisio Mendes de Farias, Ana-Maria Roxin, and Christophe Nicolle. IfcWoD, Semantically Adapting IFC Model Relations into OWL Properties. In *Proc. of the 32nd CIB W78 Conference 2015, 27th-29th October 2015, Eindhoven, The Netherlands*, pages 175–185, 2015.

Mehrdad Niknam and Saeed Karshenas. A shared ontology approach to semantic representation of bim data. *Automation in Construction*, 80:22–36, 2017. Elsevier.

Jyrki Oraskari, Mathias Bonduel, Kris McGlinn, Pieter Pauwels, Freddy Priyatna, Anna Wagner, Ville Kukkonen, Simon Steyskaland, and Joel Lehtonen. jyrkioraskari/ifctolbd: Ifctolbd 2.5, September 2020. URL https://doi.org/10.5281/zenodo.4056940.

P. P.auwels and J. Oraskari. Ifc-to-rdf converter. *URL: https://github.com/IDLabResearch/IFC-to-RDF-converter (Online; accessed : 24.03. 2016)*, 2016.

Pieter Pauwels and Anna Roxin. Simplebim: From full ifcowl graphs to simplified building graphs. In *Proceedings of the 11th European Conference on Product and Process Modelling (ECPPM)*, pages 11–18, 2016.

Pieter Pauwels and Walter Terkaj. Express to owl for construction industry: Towards a recommendable and usable ifcowl ontology. *Automation in Construction*, 63:100–133, 2016. Elsevier.

Pieter Pauwels, Thomas Krijnen, Walter Terkaj, and Jakob Beetz. Enhancing the ifcowl ontology with an alternative representation for geometric data. *Automation in Construction*, 80:77–94, 2017. Elsevier.

Mads Holten Rasmussen, Christian Anker Hviid, and Jan Karlshøj. Web-based topology queries on a bim model. In *LDAC2017–5th Linked Data in Architecture and Construction Workshop*, 2017a. URL https://backend.orbit.dtu.dk/ws/portalfiles/portal/140019661/Untitled.pdf. Online; accessed Oct. 15. 2023.

Mads Holten Rasmussen, Pieter Pauwels, Christian Anker Hviid, and Jan Karlshøj. Proposing a central aec ontology that allows for domain specific extensions. In *2017 Lean and Computing in Construction Congress*, 2017b.

Mads Holten Rasmussen, Maxime Lefrançois, Pieter Pauwels, Christian Anker Hviid, and Jan Karlshøj. Managing interrelated project information in aec knowledge graphs. *Automation in Construction*, 108:102956, 2019a. Elsevier.

Mads Holten Rasmussen, Maxime Lefrançois, Georg Ferdinand Schneider, and Pieter Pauwels. Bot: the building topology ontology of the w3c linked building data group. *Semantic Web*, 12(1):1–19, 2019b. IOS Press.

Hans Schevers and Robin Drogemuller. Converting the industry foundation classes to the web ontology language. In *2005 First International Conference on Semantics, Knowledge and Grid*, pages 73–73. IEEE, 2005.

Ranjith K. Soman. Linked-data based dynamic constraint solving framework to support look-ahead-planning in construction. In *Proceedings of the 36th International Conference of CIB W78*, pages 871–880, 2019. ISBN 2706-6568. URL https://www.researchgate.net/publication/336311906_Linked-data_based_dynamic_constraint_solving_framework_to_support_look-ahead-planning_in_construction. Online; accessed Oct. 15. 2023.

Sander Stolk and Kris McGlinn. Validation of ifcowl datasets using shacl. In *Proceedings of the 8th Linked Data in Architecture and Construction Workshop*, pages 91–104, 2020.

Seppo Törmä and Yuan Zheng. Digital Construction Ontologies (DiCon), 2021. https://digitalconstruction.github.io/v/0.5/index.html, Accessed: 05-05-2021.

Anna Wagner and Uwe Rüppel. Bpo: the building product ontology for assembled products. In *Proceedings of the 7th Linked Data in Architecture and Construction workshop (LDAC 2019), Lisbon, Portugal*, page 12, 2019. URL https://ceur-ws.org/Vol-2389/08paper.pdf. Online; accessed Oct. 15. 2023.

Matthias Weise, Thomas Liebich, Nicholas Nisbet, and Claudio Benghi. Ifc model checking based on mvdxml 1.1. *eWork and eBusiness in Architecture, Engineering and Construction: ECPPM 2016*, pages 19–26, 2017.

Jeroen Werbrouck, Madhumitha Senthilvel, Jakob Beetz, and Pieter Pauwels. A checking approach for distributed building data. In *31. Forum Bauinformatik:*

11.–13. September 2019 in Berlin. Proceedings, page 173. Universitätsverlag der TU Berlin, 2019.

2.4 A Predictive Semantic Inference System Using BIM Collaboration Format (BCF) Cases and Machine Learning (WBC 2016)

Image a scenario where the ventilation systems air terminal breaks down twice during summer. The facility manager of the building remembered that someone commented something about the air terminals. He has an intelligent search tool to search the data, types just "air terminal", and gets a list of comments containing air terminals and semantically close comments.

The article's Latent Semantic Analysis (LSA) setup implements a search engine for comments extracted from BCF XML files. Finding relevant information is an essential part of data management. As the method can be well suited to BIM issue data expressed as Linked Data, I consider this research work relevant to my dissertations (see Section 3.1).

Hypothesis 2.5 claimed that BIM issue data presented as BCF could be made more accessible compared to the standard approach of BCF XML or BCF API. [Kuo and Oraskari, 2016] presents how LSA can be used to implement a search engine for BCF comments. Because it is based on a free text search to find semantically similar comments in the issue descriptions, it can be seen as more than a text search tool. It improves the searchability of the BCF data with a feature that does not exist in the BCF API.

Table 2.9 summarises the use of the design science research method in this research.

My contribution to the paper: In this research work, I conducted data collection and preparation and designed the method to make the data suitable for the LSA algorithm. I also implemented a program code to get the data from the BCF files. We shared the work of writing the Discussion section. In that section, my work is shown particularly in the Extended BIM possibilities subsection.

Table 2.9. The work of Kuo and Oraskari [2016] in the Hevner et al. [2004] design science research framework

The work in the design science research framework (Hevner et al. [2004]):
Guideline 1: Design as an artifact
This paper introduces the Latent semantic analysis (LSA) machine learning steps to extract and process BCF data and presents the conceptual framework of a queryable knowledge-discovery system. An example search tool to search BCF comments was implemented. The procedure was described in detail but at a high level. It allows the implementation to be made in any suitable programming framework. My BCF comments extractor tool[45] was written for this work. The work was written for the textual comment of the BCF XML format, but can easily be generalized to any text collection, and, in the LBD context, the contained values of the literals.
Guideline 2: Problem relevance
The solution belongs to the queries, and machine learning problem classes. *This work is an enabler for the following BIM use cases:* BCF Communication (buildingSMART use-case GUID:B30EDF79-7A11-4E0B-9B20-104B44AD4A51 [buildingSMART International, 2022]), Monitor Maintenance, and Monitor Assets [The Pennsylvania State University, 2013] use-cases.
Guideline 3: Design evaluation
Hevner et al. Design Research Evaluation Method: Experimental Controlled Experiment.

[45] https://github.com/jyrkioraskari/BCFCommentsExtractor

Guideline 4: Research contributions

The presented method shows how textual information in the BIM context can be made searchable. As the LSA algorithm allows generating a simple measure of semantic text similarity, it can be applied to any BIM-related textual content. The expectation here is that an Indo-European native language is used. In this case, I translated Finnish into English to make the language content applicable.

Guideline 5: Research rigor

The work was peer reviewed. The work is described in the detail that the work is replicable. All the data and source code can be asked from the authors.

Guideline 6: Design as a search process

This work suits a searching step of solutions to find ways to make LBD data easier to find.

Guideline 7: Communication of research

20th Cib World Building Congress (WBC), 30th May to the 3rd June 2016 at the Tampere Hall in Tampere, Finland, and WBC16 Proceedings : Volume III, Subtitle of host publication: Building Up Business Operations and Their Logic Shaping Materials and Technologies.

The work was originally published in [Kuo and Oraskari, 2016]. The content presented here is based on our final draft of the article. The authors were kept the rights of the work (for the proceedings, copyright 2016 TUT Tampere University of Technology with the comment: "permission from the publishers or in the case of individual papers, from the author(s) of that paper.").

Vincent Kuo, Department of Civil Engineering. Aalto University, Jyrki.Oraskari@aalto.fi
Jyrki Oraskari Department of Computer Science, Aalto University, Konemiehentie 2, 02150 Espoo, Finland, Jyrki.Oraskari@aalto.fi

Abstract: Building Information Modelling (BIM) has been hailed as an artefact of collaboration, where all project participants are able to create, modify and implement design/construction configurations within the same virtual environment codified by International Foundation Classes (IFC). To facilitate communication between human participants during project development, the BIM Collaboration Format (BCF) was developed, enabling users of BIM applications to communicate issues and refer those to specific objects. A single BCF entity holds a textual description of the issue, a status, links to a BIM IFC model and objects, a picture of the issue, and a camera orientation. Therefore, BCF poses a rich repository of complex fuzzy semantic knowledge concerning design risks, such as change requests and rework proposals, in a readily accessible format based in the XML schema. This research investigates how data in BCF files can be extracted, and processed using a linear algebra method known as singular value decomposition (SVD). SVD is extensively used, along with dimensionality reduction, for pattern recognition applications and has been shown to infer semantic correlations amidst a variety of different unstructured data (e.g. texts, images, signals etc.), comparable to that of human cognitive associations. Consequently, a dynamic knowledge-base is established of past BCF cases, whereby designers can flexibly query and systematically retrieve most relevant past issues and related objects, given any known current parameter reminiscent of how a human may recall past experiences upon inquiry or design review. This paper introduces the BCF structure, describes the machine learning steps taken to extract and process BCF data, and presents the conceptual framework of a queryable knowledge-discovery system. This allows for relevant past issues to be recalled and the knowledge integrated in future designs as problem- and change-prediction. It is of particular pertinence for users of BIM, in sight of the ever-growing masses of BCF data generated from project to project.

keywords: BIM Collaboration Format, Singular Value Decomposition, Latent Semantic Analysis, Change management, Knowledge Management

2.4.1 Introduction

Design changes are very common in the building industry, where dynamic feedback, due to iterative cycles, are ubiquitous (Lyneis, Cooper, & Els, 2001). Change is a major cause of delay, disruption, and disputes (Motawa, Anumba, Lee, & Peña-Mora, 2007). Furthermore, it is hard to predict changes in construction projects, given the uniqueness of each project and the limited resources (Hanna, Camlic, Peterson, & Lee, 2004). There is thus an obvious need to better manage change, as well as to garner competence to predict and avoid unnecessary changes early during the design stage. The one-off nature of construction projects have often been used to justify unpredictable changes, however, even in such contexts many repetitive issues occur from project to project. Many of such potential change issues could well have been anticipated in advance and thus avoided given the tacit knowledge of experienced designers, and close collaboration between different design parties.

In the advent and increasing adoption of Building Information Modelling (BIM), the design reviews and change issue prediction are often done in the BIM environment. BIM is used to produce building models of different aspects of the building: architecture, structures, plumbing, heating, air-conditioning, and so on (Eastman, Teicholz, Sacks, & Liston, 2011). Therefore during the design, multiple interrelated models are produced in individual design tasks by different disciplines/parties. This creates great a challenge for the management of change across different models, in sight of change prompts and requests for information (RFIs) from both designers and contractors on site.

In recent years, the introduction of the BIM Collaboration Format (BCF) into the BIM ecosystem has enabled an additional layer of semantically rich communication between different project participants. It introduces a workflow communication capability connected to models based in the International Foundation Classes (IFC). The idea is to separate the “communication” aspects from the actual model. In particular, it was envisioned to enable the conveyance of RFIs, proposals, change prompts, or other queries/requests related to the model, collectively referred to as *issues*. It was also to allow such issues to be exchanged in collaborative work without the need to transfer the whole BIM-model as bulk data, as in the past. The BCF is an Open BIM standard format developed by BIM companies Solibri and Tekla in 2010 as “bcfXML v1”, and later “bcfXML v2” became released in October 2014 and adopted by BuildingSMART (BuildingSmart, n.d.).

The BCF presents a repository of semantically rich issues of change-related knowledge. Alluding to knowledge management concepts of Nonaka & Takeuchi (1995), the BCF presents a codified form of, otherwise tacit, knowledge, which

can then be *internalized* by the designers to be applied in value creating work, as part of the knowledge management lifecycle. However, *internalizing* these issues as human practitioners currently requires much time and effort as it occurs organically through the acquisition of experience dealing with such change cases from project to project. In order to aid this process, this study proposes the mining/extraction of textual natural language descriptions in BCF issues and process the dataset using the machine learning textual processing technique known as latent semantic analysis (LSA). LSA is based on a linear algebra operation called singular value decomposition (SVD) and has been shown to infer conceptual links amid natural language texts comparable to human semantic intuition (Landauer, McNamara, Dennis, & Kintsch, 2007) and similar techniques have been used in a variety of other pattern recognition problems (Deerwester, Dumais, Furnas, Landauer, & Harshman, 1990). This allows the matching of texts on semantic/conceptual level rather than explicit string matching used in word search-functions, where semantic relationships between texts are not involved. This ability of LSA is cardinal in managing and retrieving possible future change cases, where different building elements may be related to each other through different types of issues qualified by a variety of parameters. Consequently, this enables change cases to be automatically captured and stored in such a way that future designers are able to effectively query the system for past similar and/or related issues. This helps to enhance the designers' capacity to learn or *internalize* historical change-related knowledge and thus improves future design decisions in avoiding change.

In the Section 2 we cover some preliminaries regarding the background and mathematical basis of the latent semantic analysis method. Section 3 describes the methodology applied in our particular case, the nature of the textual *issue* descriptors extracted from the BCF of a specific project, and the pre-processing steps taken before the analysis. We then discuss the results in Section 4 with reference to other potential developments. Section 5 concludes by drawing up a brief summary and reiterating the value of our proposal in the BIM-based design change management context.

2.4.2 Theoretical preliminaries of LSA and SVD

Latent semantic analysis (LSA) is a techniques for creating vector-based representations of texts, which are claimed to capture their semantic content. The primary function of LSA is to compute the similarity of text pairs by comparing their vector representations, and has been shown to closely match human capabilities on a variety of tasks. It extends the vector-based approach by using Singular Value Decomposition (SVD) to reconfigure the data.

Pre-processing

LSA starts with a text quantification method based on what is known as Vector Space Model (VSM) (Salton, Wong, & Yang, 1975), where a corpus of d documents using a vocabulary of t terms is used to compile a t x d matrix **A**, containing the number of times each term appears in each document (term frequencies). Some trivial terms such as "the", "of", etc. (the stopwords) are excluded, and some others are consolidated, because they share a common stem (Porter, 1980) or some other lexical quality. The frequency counts in **A** typically undergo some transformation (term weighting) known as TF-IDF that penalizes common terms and promotes rare ones. After weighting, the term frequencies are also normalized so that the sum of squared weighted term occurrences within each document is equal to one (Salton, 1989). Subsequently, **A** is subjected to Singular Value Decomposition (SVD).

Singular Value Decomposition

Matrix **A** of dimensions t x d is decomposed into the product of three other component matrices **U** (t x t), **S** (t x d) and $\mathbf{V^T}$ (d x d), that is, $\mathbf{A = USV^T}$. **U** columns consist of the *term* eigenvectors, and **V** columns are the *document* eigenvectors, and **S** is a diagonal matrix of singular values (i.e. square roots of common eigenvalues between terms and documents). In other words, matrix **U** describes the original **A** row entities as vectors of derived orthogonal factor values, matrix **V** describes the original **A** column entities in the same way, and the middle **S** matrix is a singular diagonal matrix containing scaling values in descending order along the diagonal of the matrix. When the three component matrices are multiplied the original **A** matrix can be reconstructed. The **U** and **V** matrices are thus regarded as all the constituent patterns that determine the original matrix **A**. The columns in **U** and **V** are orthonormal vectors that describe these patterns and are arranged in descending order of significance.

Dimensionality reduction

Dimensionality reduction is carried out in order to remove noise (small irrelevant patterns) inherent in natural language. This step is done by removing column vectors from **U** and **V** starting from the least significant, and retaining the most significant vectors. This corresponds also to the same number of diagonal singular values retained in the **S** matrix. The reconstruction of the truncated/reduced component matrices produces another matrix (essentially an approximation

of matrix **A**), which exhibits latent semantic relationship between words and documents not evident in the original matrix **A**. The number of dimensions to keep is a difficult question as it depends on how much of the original variance is desired to be kept without loss of relevant data. In linguistic terms, it is difficult to interpret intuitively. However, many case-specific guidelines are given within different contexts. A very common technique is to plot the square of singular values of the **S** matrix against the number of dimensions (Sidorova, Evangelopoulos, Valacich, & Ramakrishnan, 2008). One may choose the number of dimensions corresponding to where the singular values decrease substantially (elbow of the graph) indicating the point where the patterns become insignificant. Generally, the more noise is removed (dimensions reduced), the clearer will the semantic relationships be revealed in the reconstructed approximate matrix.

Cosine similarity

The reduced **U** and **V** row vectors represent the semantic space of the words and documents. Thus a similarity function can be used to measure semantic correlations between term-term, document-document, and term-document relationships. The cosine similarity function tends to work well empirically (Landauer et al., 2007) and is widely used for vector matching in many applications. Cosine similarity is denoted by the cosine of the angle between two vectors (suppose vectors a and b) and can be calculated by dividing the dot product of a and b by the product of their magnitudes, that is, $a \,.\, b \,/\, (\,\|a\|\,\|b\|\,)$.

2.4.3 Methodology

Broadly, this study involves the automatic extraction of natural language information from BCF issues and processing it using latent semantic analysis. This is used to demonstrate how higher level semantics and linked data may be inferred automatically from existing data or documentation embodying some level of professional knowledge. The semantics inferred can then be fed back to the workflow to augment the knowledge of practitioners for future work. The BIM data acquired for this study is from a large renovation project of a heritage complex in downtown Helsinki, and the BCF issues thus reflect the nature of changes accordingly - predominantly related to mechanical, electrical, and plumbing (MEP) services. The steps of this study are covered in the subsections below.

Extract issues from XML schema of BCF

As described in Paasiala et al. (2015) a BIM Collaboration Format v2.0 formatted bcfzip file contains a folder for each issue. The issues correspond to proposals and change requests in BIM-data-models (BuildingSmart, n.d.). Each folder contains a markup.bcf file that embodies textual information about the issue, a viewpoint.bcfv file that specifies the related elements, the viewpoint, and the camera angle, and snapshot.png pictures describes the issue.

Only markup.bcf files contain human written sentences. In that XML file, although title and description tags can contain clauses, in practice, only comment tags had any content. A small program was written in the Java programming language to extract the comments from the files. The full source code of the program is available in (Oraskari, 2015). When extracting the bcfzip file, the program follows the first steps that was presented in (BIMServer, 2015). However, instead of marshalling the XML documents into Java objects, we parse the files using a JDOM XML parser (Hunter & Lear, 2015). The XML document tree of the markup.bcf is traversed recursively. Whenever there is a comment tag in the graph, the content is copied. A total of 71 passages of text from BCF *issues* were extracted as natural language descriptions for each issue logged by human project participants.

Textual parsing and term-document matrix

The extracted text passages (documents) are then parsed into words. The words are filtered in order to remove stopwords – such as “the”, “a”, “an”, “is”, “in” etc. – that supposedly do not contribute to the semantic value of each document. A standard “stoplist” is used for the English language. The words then undergo morphological stemming, where the words with different morphological conjugations would be reduced to the root of the word. For instance, “ventilate”, “ventilating”, “ventilated” etc, would be reduced to be the same stem or root of the words. As an example, the document “Office area: Electric duct location? Radiators hot water piping penetrates electric duct” are parsed into the following terms: “are”, “duct”, “electr”, “hot”, “locat”, “office”, “penetr”, “pipe”, “radiat”, “water”. The resulting words are now considered the terms in the term-document matrix (t x d), consisting of the occurrences of term t in document d.

The term-document matrix is transformed through TF-IDF (term-frequency, inverse document frequency) weighting to enhance the discrimination between documents, by discounting the weight of very common terms and promoting the weight of less occurring terms (Salton, 1989). The term-document matrix’s raw occurrence values are replaced by the product, $w_{td} = tf_{td} * idf_t$, where tf_{td} is the

normalized term occurrence in each document d; and $\boldsymbol{idf_t} = \mathbf{log(N/}\boldsymbol{df_t}\mathbf{)}$, where **N** is the total number of documents in the collection, and $\boldsymbol{df_t}$ is the number of documents that contains term t. After TF-IDF weighting, the document vectors are normalized to remove the advantage that longer documents have over the short documents. Cosine normalization is an effective normalization technique and is computed, by dividing the TF-IDF weights by the Euclidean length of the respective document vector (Salton, 1989).

SVD, dimensionality reduction, cosine similarity, and query

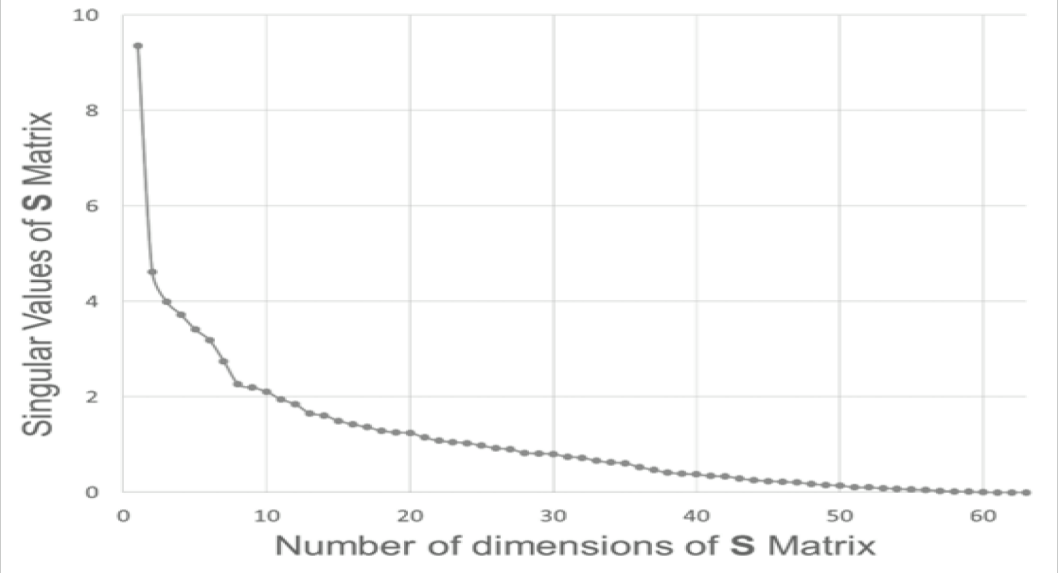


Figure 1: Plot of squares of singular values against the number of dimensions The term-document matrix (t x d) undergoes singular value decomposition to yield the **U**, **S** and **V** matrices. Effectively, the terms and documents are vectorised and transformed or mapped to the same vector-space through dimensionality reduction. To choose how many dimensions to retain, the squares of the diagonal values of the singular **S** matrix are plotted against the number of dimensions (the number of singular values to retain) as shown in Figure 1. The number of dimensions to retain is chosen to be 13, after which the graph starts to flatten out, indicating that the patterns before dimension 13 contributes most to the variance of the original matrix **A**. Singular values after dimension 13 is therefore assumed to constitute "noise".

Cosine similarity and query

The terms and documents can now be represented as vectors of 13 dimensions, as the rows of the reduced **U** and **V** matrices respectively. The similarity between any two vectors can thus be computed (through cosine similarity as earlier explained). If the two vectors are identical, the cosine similarity is 1 since the angle between them is 0ř. Vectors that are totally uncorrelated has a cosine similarity of 0 (angle 90ř), while cosine similarity of -1 implies that the two vectors are total opposites (angle 180ř) or inversely proportional intuitively. The cosine similarity between vectors of the reduced **U** and **V** matrices equates to semantic similarity between them. Since this process is automated, it allows the system to calculate similarities upon query and recommend similar terms or documents from the database. The query is achieved by creating a pseudo-document consisting of a set of terms as query input. This pseudo-document can be represented as an explicit vector (q) of term occurrences, which can be mapped to the semantic space. The semantic query vector (q_s) can be obtained, through a simple algebraic manipulation following $\mathbf{A}=\mathbf{USV}^{\mathbf{T}}$, to be $q_s = q^{\mathrm{T}}\mathbf{US}^{-\mathbf{1}}$. Cosine similarity can then be applied to gauge the similarity between the query vector q_s and all the other semantic term and document vectors represented by the dimensionality reduced rows of **U** and **V** matrices.

2.4.4 Discussions

In principle, the cosine similarity values between term and document vectors depends on the degree of correlations between the row and column spaces of the original **A** matrix, in particular, how co-occurrence of terms within the same documents indicates some level of relationships between these terms. The fundamental concept of semantic inference in human text is that similar/related terms would have much higher co-occurrence within documents than unrelated terms. For instance, considering the hypothetical corpora consisting of all of human literature, it is understandable that the terms “pear” and “apple” would have a much higher chance of occurring together in the same sentence, than “pear” and, for instance, “radiochemistry”, to name an arbitrary example. The semantic meaning of certain terms, and their potentially numerous metaphorical or ambiguous nuances in natural language, can be qualified by other terms that occur together with it. In the same way, a semantic space can be established for domain knowledge, such as design change issues in this case, and co-relationships between the constituents of these issues can be automatically inferred, even to reveal non-apparent associations. To demonstrate the logic, some results of the analysis are shown with more specific detailed discussions covering the interpretation of the output of the proposed data extraction and latent semantic

method on change issue management, integrated in BIM.

Interpretation of query and matching

Queries can be made to the system by inputting any number of terms. Some arbitrary query and results are shown in Figure 2. One- and two-term queries are shown and the top 10 semantically related documents (issues) are tabulated. One can very quickly use simple data visualizations to gain an overview of the nature of the spectrum of issues with relation to specific concepts. For instance, issues related to “corridor” are high in similarity though not as spread out as say “window”, which seems to have one very similar issue, while the rest are spread out. Issues related to “electric cable” and “ventilation ducts” do not have as high semantic similarities as others, implying that even though the issues related to both concepts are widespread, they are not very specific. “Ablution area” has quite a few issues that are very semantically relevant, but not as widespread as the others. Such a table gives a strategic view about the how common certain issues occur with respect to specific concepts queried by the user, as well as how much semantic similarity there is between the issues and the queried concepts. This poses a suggestive function for the user to gauge the likelihood that certain change issues may arise in a new project given certain parameters as concepts to query the historical knowledge.

"Electric cable"		"Ventilation ducts"		"Ablution area"		"Windows"		"Corridor"	
Issue24	0.664	Issue54	0.747	Issue57	0.909	Issue53	0.851	Issue1	0.857
Issue62	0.618	Issue33	0.697	Issue68	0.902	Issue50	0.657	Issue27	0.805
Issue63	0.588	Issue11	0.683	Issue67	0.775	Issue44	0.617	Issue28	0.777
Issue38	0.544	Issue65	0.643	Issue60	0.767	Issue54	0.591	Issue7	0.682
Issue39	0.538	Issue70	0.603	Issue16	0.648	Issue30	0.575	Issue36	0.607
Issue8	0.532	Issue53	0.560	Issue14	0.353	Issue14	0.540	Issue20	0.576
Issue57	0.424	Issue50	0.545	Issue44	0.302	Issue11	0.493	Issue49	0.454
Issue32	0.400	Issue71	0.540	Issue17	0.290	Issue65	0.446	Issue18	0.315
Issue20	0.334	Issue15	0.527	Issue49	0.226	Issue67	0.431	Issue11	0.296
Issue2	0.318	Issue38	0.480	Issue19	0.195	Issue70	0.320	Issue59	0.275

Figure 2: Top 10 semantically related document/issues with respect to specific concept queries

"Electric cable"		
Issue24	0.664	Technical area: Electric cable tray through ventilation duct
Issue62	0.618	Technical area: Electrical box through ventilation pipe and ventilation pipe through cable tray
Issue63	0.588	Technical area: Cable tray through sewage pipe
Issue38	0.544	Business area: Electric cable tray through ventilation duct
Issue39	0.538	Business area: Electric cable tray location
Issue8	0.532	Technical area: Electric cable tray penetrates inside ventilation pipes
Issue57	0.424	Ablution area: Electric cable tray underneath the ceiling
Issue32	0.400	Sewage drain through electric cable tray?
Issue20	0.334	Corridor area: Electric cable tray through the hot water pipe
Issue2	0.318	Office area: Electric duct location? Radiators hot water piping penetrates electric duct

"Windows"		
Issue53	0.851	Window and ceiling height? Architect model space object is missing
Issue50	0.657	Social area: ventilation duct door and water drain are at the same point. Floor is missing.
Issue44	0.617	Lobby area: ventilation piping in the same place as the front surface of the ceiling
Issue54	0.591	Ventilation duct and ventilation terminal clashes with electric duct and cooling pipes. Space object is missing
Issue30	0.575	Display area: Radiators in front of windows and door
Issue14	0.540	Stairwell: modelled staircase elevations dimensions?
Issue11	0.493	Drainage pipe penetrates ventilation ducts.
Issue65	0.446	Utility area: Cooling coil penetrates ventilation duct and electric cable tray
Issue67	0.431	Stairwell: ventilation duct is underneath the ceiling and goes through the building structure
Issue70	0.320	Technical area: Cooling pipe and electric cable tray penetrate the ventilation duct

Figure 3: Full issues semantically similar to queries: “electric cable” and “windows”

Figure 3 tabulates more detail about the “electric cable” and “windows” queries for further interpretation. For “electric cable” the top 10 semantically similar issues all contain some form of the terms “electric” or “cable”, but it doesn’t necessarily need to be the case, as the system uses “concept matching” rather than “string matching”. Nevertheless, it is of course expected that issues explicitly containing the query terms would have high semantic similarity. A few explanations need to be given for how the issues are ranked for “electric cable”. One sees that cases to do with clashes with ventilation related elements seem to be placed higher than other issues related to electric cables. This is due to the terms being weighted differently. “Ventilation” related terms appear many times in the corpus, so much so that the weight of those terms have been reduced as they are deemed non-unique (through TF-IDF and normalization). Therefore the contribution of “ventilation” to the semantic value of the issue is low, thus leaving the semantic weights of the electric cable to be prominent in that specific issue. A bit lower down (Issue57, 32, and 20 for instance) all contain “electric cable”, but because of the contributing semantic weight of the other related elements, the share of the weight of “electric cable” is reduced with respect to the issue. Therefore, as one can see, the method is able to cater for different semantic weights of terms and aggregation of terms to match whole issues to a concept, and not merely wherever a term appears within an issue. This is the property of LSA that allows latent semantic traits of passages of texts to be identified and mapped to other passages of text, as concepts denoted by the

aggregation of semantic vectors.

The results for “windows” (Figure 3) can be interpreted in a similar way. However, one sees that only Issue53 contain “window” and is expectedly semantically significant, while the rest is somehow related even though “window” doesn’t occur explicitly. Issue 50 for instance is matched, because Issue53 (where “window” is evident) also is related to some object being “missing”, while Issue50 has similar (i.e. floor is “missing”). This co-occurrence of concepts has created a link between “window” and some problem related to “missing” objects or information. Issue44 is linked probably due to the relation to “ceiling” which also occurs in Issue53. LSA has the ability to take into account the complex second order relationships of concepts, in fact, all latent relationships identified are based on aggregates of multiple orders of semantic associations across all the issues in the corpus.

"in the conference area, the hot water pipe is below the ceiling"		
Issue20	0.734	Corridor area: Electric cable tray through the hot water pipe
Issue7	0.677	Corridor area: Hot water pipes penetrates through the electric cable tray.
Issue41	0.567	Office area: Lighting rail clashes with ventilation terminal. Electric cable tray clashes with ventilation pipe and hot water pipe.
Issue42	0.474	Office area: Electric duct and radiator clash
Issue59	0.465	Corridor: Level and the stair in the same place? Cable tray and water pipe clash

Of course, it is also possible for the system to return the most semantically related terms rather than issues as shown above. However, in this particular exercise, the terms by themselves (being out of context) are difficult to infer meaning. It is therefore more meaningful for this particular renovation orientated context to retrieve the issues. Other cases, for instance, colloquial discourse within large corpora of literature, term retrievals usually reveal synonymy – multiple terms describing similar concepts.

Figure 4: Example of a natural language query

Due to the limited length of the paper, more detailed discussions and interpretations will be reserved. However, it is worth mentioning that given the nature of how query vectors are established, the queries can be indeed done in natural language. An example in Figure 4 shows the most semantically related issues returned given the arbitrary natural language query. Such queries will be much more accurate and valuable when the training corpus is big. In our case, we only have 71 issues (for one change review cycle) consisting of 78 terms. If, for instance, the BCF knowledge regarding change management is extracted from 10 projects, each having just 100 change issues, 1000 issues could be processed and vectorised semantically in the database, and used as suggestive knowledge augmentation for future engineers’ decisions. The benefit is that the system can handle datasets of change cases that are ever growing from project to project

and the latent semantic structure of the data also updates accordingly. This is comparable to how a human engineer may accumulate experience over time through experience, and apply the lessons learnt to new decisions and predictions to avoid problems.

Currently, with only 71 issues, consisting of 78 terms (4430 characters, thus supposedly 4.3 kB of data), it is remarkable how intuitive semantic relationships between terms and documents can be inferred through the proposed semantic system. This is noteworthy in sight of the association of machine learning techniques with "big data", where large datasets (in the order of Gigabytes) apparently needs to be mined to train algorithms. Our proposed technique demonstrates effective results even given small sets of data, available in all segments of the construction context.

Extended BIM possibilities

It is useful to enrich the textual comments used in this study with some further explicit contextual information. They can be added in the sentences in the form of "keywords" or even include ontological entities from IFC schema. The sample sentences would contain not only more information, but also it would allow the learning algorithm to use the information that is available in the BIM models.

The BCF specification makes it possible to obtain a list of identifiers of entities that are related to a BCF issue. In principle, those identifiers can be used to fetch information from the relevant BIM models. This can be done either manually using a BIM tool or using a program to automate the task. In practice, the relevant identifiers are listed in the viewpoint.bcfv files, the corresponding entities can be marked visible or selected using tags. The viewpoint.bcfv files further specify a camera view inside a BIM model. That makes it possible to use standard 3D graphics processing to calculate which parts of the model are visible for the camera. For example, ray tracing (Glassner, 1989) can be used. The parts could be used to infer the affected BIM entities. They could, in turn, be used to enrich the issue descriptions, so as to allow these BIM entities to be included in the latent semantic analysis.

Additionally, if we had a series of changes in the model that has taken place after the BCF is time stamped, and possibly before the next BCF prompt, the series can automatically convey information of the actual changes that were imposed upon the BIM model, following the requests in the previous BCF issues. In principle, BIM tools could keep track of changes in the model, which is possible if the BIM tool can be changed. Otherwise, a more generic way is to export the model versions

into IFC/RDF representation (Pauwels & van Deursen, 2012) and use algorithms presented in Oraskari and Törmä (2015) to provide change sets of added and removed RDF triples. The triples include identifiers of entities in the model and they have location in the graph. It makes it possible to deduce the affected entities in the BIM model and the triples express the changes made. The change description could be used as a context for a BCF issue comment. In other words, these change events identified can be attached to the change requests in BCF issues, therefore establishing the possibility for the semantic retrieval of not only change requests, but as well as the change actions taken in response. For instance, the natural language query in Figure 4 may be used to retrieve similar past issues, and if the past issues also include the following change events, it poses a suggestive function to respond to future change requests/prompts, be it in BCF or otherwise. This is especially useful for a novice engineer who may query possible change risks, and the system adds an additional layer to suggest the possible actions to be taken in response to the risk, thus catering both for the prediction (change avoidance), as well as mitigation (change cost alleviation) functions.

2.4.5 Conclusions

Design change management poses a huge challenge in any civil engineering project. This study sets out to investigate how to better manage design changes in a BIM integrated environment, in sight of the growing adoption of BIM in typical workflows. In particular, this study demonstrates how data from existing BCF files in the BIM environment can be extracted and semantic inferences made to predict future changes given natural language queries by the user. Natural language is particularly interesting as it is able to capture specifics, as well as fuzzy semantic dimensions, required to describe complex change prompts and issues. The proposed method uses data extraction techniques to obtain textual data from BCF issues, processes the texts via latent semantic analysis and establishes a semantic knowledge repository, which can be flexibly queried by future users. The types of semantic inference is shown and discussions are made on how retrieval results can be interpreted to illustrate the reasoning of the technique. This semantic knowledge repository, in the form of vectorised terms and issues, is to some extent an analogy for how human tacit experience in change prediction and mitigation can be captured and presented in an intuitive manner, that is also both scalable in size and flexible for multiple domains. The computational technique thus simplifies the complexity of knowledge/information modelling, by harnessing the properties of linear algebra decomposition, thus enabling an inductive logic to be applied to complex, unstructured data. It is remarkable that with as little as 4.43kB of training data in this exercise, intuitive semantic associations can be inferred and thus used for matching intuitive concepts, otherwise tacit in

expert reasoning. On a fundamental level, the simplicity and effectiveness of the prototype democratizes artificial intelligence for construction contexts, and shows the possibilities beyond natural language texts to include specific BIM or IFC entities, made possible by innovations in linked-data and ontology-related research. Ultimately, the effectiveness of the system can be enhanced given more data over time, which also caters for the adaptability of the system by considering the shifting trends of the industry or domain with respect to how design changes are managed.

2.4.6 Acknowledgements

This research has partly been carried out at Aalto University in DRUMBEAT "Web-Enabled Construction Lifecycle" (2014-2017) —funded by Tekes, Aalto University, and the participating companies.

2.4.7 References

BIMServer. (2015). BIMServer clash detection. Retrieved November 28, 2015, from https://bimserver.googlecode.com/svn/trunk/ClashDetection/src/org/bimserver/clashdetection/bcf/Bcf.java

buildingSMART. (n.d.). BCF intro. Retrieved November 30, 2015, from http://www.buildingsmart-tech.org/specifications/bcf-releases

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., & Harshman, R. (1990). Indexing by latent semantic analysis. Journal of the American Society for Information Science, 41(6), 391407.

Eastman, C., Teicholz, P., Sacks, R., & Liston, K. (2011). BIM Handbook: A Guide to Building Information Modeling for Owners, Managers, Designers, Engineers and Contractors. Building (Vol. 2).

Glassner, A. S. (1989). An introduction to ray tracing. London: Academic Press.

Hanna, A. S., Camlic, R., Peterson, P. a., & Lee, M.-J. (2004). Cumulative Effect of Project Changes for Electrical and Mechanical Construction. Journal of Construction Engineering and Management, 130(6), 762771.

Hunter, J., & Lear, R. (2015). JDOM. Retrieved November 28, 2015, from

http://www.jdom.org/ Landauer, T. K., McNamara, D. S., Dennis, S., & Kintsch, W. (2007). Handbook of Latent Semantic Analysis. Psychological Review.

Lyneis, J., Cooper, K., & Els, S. (2001). Strategic management of complex projects: a case study using system dynamics. Syst. Dyn. Rev., 17(3), 237260.

Motawa, I. a., Anumba, C. J., Lee, S., & Peña-Mora, F. (2007). An integrated system for change management in construction. Automation in Construction, 16(3), 368377.

Nonaka, I., & Takeuchi, H. (1995). The Knowledge-creating Company: How Japanese Companies Create the Dynamics of Innovation. Oxford University Press.

Oraskari, J. (2015). jyrkio/BCFCommentsExtractor. Retrieved November 28, 2015, from https://github.com/jyrkio/BCFCommentsExtractor

Oraskari, J., & Törmä, S. (2015). RDF-based signature algorithms for computing differences of IFC models. Automation in Construction, (June).

Paasiala, P., Laukanen, J., Liftländer, L., Lindhard, K., Pijnenburg, E., & van Berlo, L. (2015). BIM Collaboration Format v2.0 Technical Documentation. Retrieved November 28, 2015, from https://github.com/BuildingSMART/BCF-XML/tree/master/Documentation

Pauwels, P., & van Deursen, D. (2012). IFC/RDF: adaptation, aggregation and enrichment. In Proceedings of the 1st Workshop Liked Data Architecture and Construction (LDAC2012). Ghent, Belgium.

Porter, M. F. (1980). An algorithm for suffix stripping. Program: Electronic Library and Information Systems, 14(3), 130137.

Salton, G. (1989). Automatic text processing: the transformation, analysis, and retrieval of information by computer.

Salton, G., Wong, A., & Yang, C. S. (1975). A vector space model for automatic indexing. Communications of the ACM, 18(11), 613620.

Sidorova, A., Evangelopoulos, N., Valacich, J. S., & Ramakrishnan, T. (2008). Uncovering the Intellectual Core of the Information Systems Discipline. MIS Quarterly, 32(3), 467A20.

2.5 Discussion and Conclusions

In this chapter, I have discussed four topics: presenting data as linked data, conversion quality, filtering, and findability. They are related to the three data management topics of data organisation, data quality, and data governance (see Section 1.5). Here I will focus on assertional axioms (ABox).

I have consistently used the word *conversion* using the Cambridge English dictionary meaning of file conversion in the context of IT. It refers to the process that changes computer data inputs and outputs them differently, in this case, applying a set of ontology descriptions. Here the weight is that we read the data from one standard computer presentation and write it deterministically in another. An alignment definition between ontology classes is a strong statement of mappability of the concepts. I could also have used *translation* to denote that the process reads one human-defined ontology language and expresses the same information using another. The latter denotes that the process is not only a syntax mapping; the ontology classes are like words in a spoken language. When translating, meanings may not map 100%, and connotations can vary.

The first data management topic is achieved when data are published using an ontology that allows easy access to the information. We hypothesised that this would be achieved by exposing data using the core modular ontologies published by the W3C Linked Building Data community group. When [Bonduel et al., 2018] was published, these were BOT (building topology), PRODUCT (classification of building elements), and PROPS (building-related properties). The article showed that it is possible to convert IFC-STEP [Kiviniemi, 1999] data into assertions using LBD community ontologies. It provides an easier-to-retrieve data format than the previous ifcOWL format.

This is well achieved for the building topology, elements and properties. The article does not cover geometry, bSDD classification data, or data units. Support for these has

been added later in the IFCtoLBD converter, but there are still knowledge gaps.

Currently, the selected list of ontologies could be different (for example, the Ontology for Property Management (OPM) [Holten Rasmussen et al., 2018] would be used), but the results still apply.

The conversion focuses on providing RDF-based data. To cover the second point of the 5-star definition of the Linked Data principles, i.e., the data should be published online, and the addresses made dereferenceable. We have addressed this in the DRUMBEAT project by creating a concept for a Linked Data server [Törmä et al., 2012, Hoang and Törmä, 2017].

The article presented in Section 2.2 then extended the conversion idea by showing that rule-based conversion is also possible, allowing a model in which rules maintain the conversion relationship even when the model is updated.

On the other hand, the first data management theme also includes the idea that data should be fit for purpose. Using IFC as a starting point, the data content is based on the decisions made to export the data from the native model. For example, when exporting data, the appropriate profile must be selected to get the information needed to calculate the life cycle analysis for a model. However, a large amount of data may still be irrelevant to the application context. Section 2.3 presents a solution for Model View Definition filtering of LBD data. The idea is that filtering the data improves its accessibility.

Data quality for conversion is discussed in Section 2.3; quality assurance of application data in SHACL is also mentioned in the article. My work on mvdXML (see GitHub[46]) is relevant in that it applies to ifcOWL-based LBD data. These relate to the selected aspects of data quality. I have addressed the selected aspects to cover data conversion, filtering and querying of Linked Building Data. They have been reflected against the data management

[46] https://github.com/jyrkioraskari/mvdXMLChecker (Accessed June 13, 2023)

themes of data organisation and data quality. The selected hypotheses describe the focus of the research in this area. The associated artefacts and techniques provide insight into the associated research questions and can be generalised to some extent and used to solve new but related problems.

This study aimed to develop a method for converting IFC-STEP standard formatted building models into Linked Building Data (LBD) and to analyse the characteristics of the solution. The proposed IFCtoLBD converter simplifies the IFC building models into modular LBD Abox graphs, which are more user-friendly than ifcOWL Abox graphs because they have a simplified structure that is more suited to a Semantic Web environment. The study also found that while a rule-based approach is feasible, an algorithmic approach can provide more comprehensive coverage of LBD. The method presented in the study is a data centric solution where the rules are separated from the programming code. In addition, several unit test methods are presented to improve the quality of the LBD conversion. Model views can be created using SHACL rules to filter LBD data for specific applications. The study demonstrates also how data from existing BCF files in the BIM environment can be extracted, and semantic results for natural language queries can be inferred.

Section 2.4 presents a way to make the data semantically searchable. Latent Semantic Analysis is a natural language processing technique that can be used for this purpose. It is one of the basic techniques in topic modelling but by no means the only one. Others include LSA, pLSA, LDA and lda2vec. What is essential is the idea of how to transform data, i.e. structural data containing natural language, into a searchable form.

Efficiently querying managed datasets for monitoring is crucial, requiring well-organized and structured data. Adopting common open standards and shared ontologies serves as a universal language in the field. Linked Building Data builds upon Linked Data principles and incorporates open standards from architecture, engineering, construction, and facility management sectors, like

Industry Foundation Classes, BIM Collaboration Format, and buildingSMART Data Dictionary.

Chapter 3

Linked Building Data and Ontologies

In this chapter, the research questions are: what makes an LBD ontology good, and what are the best practices to follow. Also, how to express version data using LBD is a topic here.

I have chosen the following hypotheses that express the chosen focal points that I have researched in more detail.

Hypothesis 3.1 *Issue handling in the building industry context can be implemented using Linked Data. The solution has advantages over the standard BIM Collaboration Format application programming interface (BCF API) or its file-based alternative, BCF XML, in terms of improved data retrieval and management. It enables retrieval of data from the entire issue history, allowing new analyses to be made, e.g. on the effectiveness of the work of teams, the reoccurring issues, and solutions.*

Section 3.1 introduces our work addressing Hypothesis 3.1.

Hypothesis 3.2 *There are effective ways to express version information of BIM Collaboration Format (BCF) when the data is expressed as Linked Data.*

Section 3.2 introduces our work handling Hypothesis 3.2.

Hypothesis 3.3 *The proposed dynamic buildingSMART Data Dictionary (bSDD) ontology improves the consistency of the LBD data sets compared to the earlier methods where either separate exports were used or the models were linked to an online but non-RDF entry of the data.*

I cover Hypothesis 3.3 in Section 3.3.

In this Chapter, I will introduce what support I have for the claims and discuss arguments that are against them. In Section 3.5, I will summarize findings presented in the chapter.

Figure 3.1 summarises how the research questions, hypotheses and their handling are connected. Then, Figure 3.2 shows my articles of the chapter at a glance.

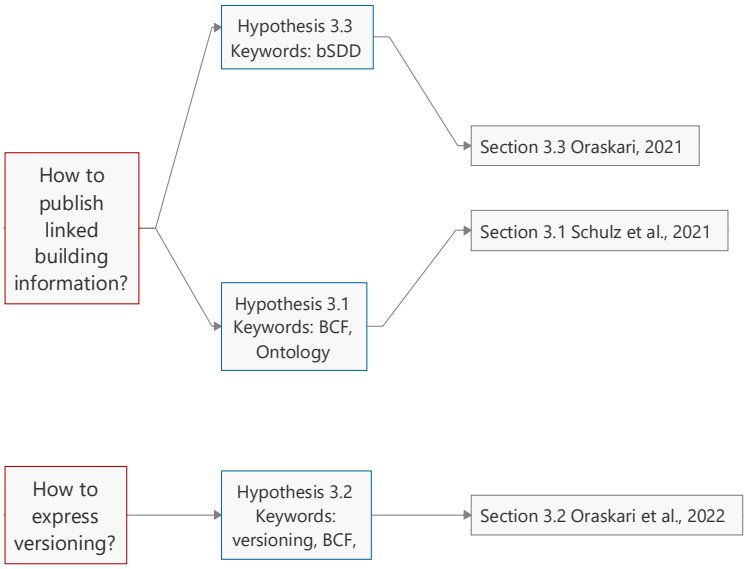


Fig. 3.1. How the research questions, hypotheses, and their handling in articles and the sub-sections are connected.

Publication	Objectives	Method	Key findings
Schulz et al., bcfOWL: A BIM collaboration ontology., 2021	• How to express the core ideas of BCF by following the best practices of ontology design? • How to keep compatibility with the original BCF API and BCF XML	The Design Science Research Method: Descriptive Scenario	• A transformation of BCF into a Linked Data approach is viable. • The new data format makes it easier; - to retrieve relevant issue data - attach and track provenance information
Oraskari et al., Towards describing version history of BCF data in the Semantic Web., 2022	• How to express changes using bcfOWL	The Design Science Research Method: Descriptive Scenario, Analytical Static Analysis	• four different Linked Data approaches to describe the history of changes in BCF Topics • RDF-star approach performed reasonably well • Future work: how RDF-star should be described in a Tbox statement for reasoning
Oraskari, Live Web Ontology for buildingSMART Data Dictionary., 2021	• to publish the buildingSMART Data Dictionary classifications and terms as a web ontology • use them in theLinked Building Data context • keep the data in sync with the buildingSMART Data Dictionary primary data	The Design Science Research Method: Controlled Experiment	• It is feasible to create a dynamic web API solution that dynamically creates ontologies from buildingSMART Data Dictionary database. • compared with the earlier solutions, this has a clear advantage: it is update.

Fig. 3.2. My articles related to publishing ontologies at a glance

3.1 bcfOWL: A BIM collaboration ontology (LDAC 2021)

Image a scenario in which a facility manager creates a reparation task and tracks the fixes made. During the process, he communicates with contractors about the status of the building details or any new issues that arise,

and they report back. All of this can be done using the buildingSMART BIM Collaboration Format. When Linked Building Data is used, a natural selection for BCF serialization is to use the Resource Description Framework (RDF) as a communication format.

Hypothesis 3.1 claims that BIM-related issue handling can be implemented using Linked Data. This has two levels: (1) The data can be expressed and saved in a Linked Data format. In this article, we show that the main concepts of the BIM Collaboration Format (BCF) specification can effortlessly be expressed using the W3C Web Ontology Language (OWL), which also implies that issue data can be expressed using that ontology. We then created the standard BCF API interface and showed that the data could be mapped and served that way. (2) The data can also be published using linked data principles (presented in Section 1.3) and linked to other content on the internet. This follows from assertions written using an OWL ontology in the RDF, and RDF data can be published as linked data.

Furthermore, one claim was that federated Linked Data queries make the whole data history more accessible. This is a bold claim since the internal implementation of the data safe of a BCF server can be anything. Therefore, the reasoning goes that there may be many collaboration organizations and many BCF server providers in a longer life span. In this context, the data would be better available if the software providers had published the data using open standards such as BCF. Furthermore, Linked Data would allow linking to existing data sets, eliminating the need to copy data from one system to another. They would allow querying of the data as a single source of information. It is a consistent framework (that is, Linked Data and SPARQL) for data access. Then, a counterargument against using Linked Data is that it might not be the only solution. For example, also GraphQL could make federated queries possible.

Then, although OpenAPI (i.e., Restful AP) and GraphQL specifications are easy to access, they miss the full integration to Linked Building Data (LBD), as LBD is

RDF-based and the shared ontologies, which can act as a lingua franca between computer systems, or a shared conceptual framework for reasoning systems. So, in principle, semantically rich queries that cross the silo borders of BCF and use BIM data and all data available as Linked Data could be used to make more advanced analyses of historical BCF issue data. Software providers could implement something similar proprietarily, but Linked Data already offers that vision and is based on open standards.

Table 3.1 summarizes the use of the design science research method in this research.

My contribution to the paper: This conference article was written in collaboration with Oliver Schulz. Oliver Schulz and I wrote an equal part of the manuscript.

The work originated from our joint internal hackathon work on the topic, from which I wrote the first OWL versions using the Apache Jena APIs ontology library in Java, with the idea that the library would guide to clean and faultless OWL syntax. Later, we moved on to maintain the OWL description manually. Since the idea was not to convert the existing BCF standard but to maintain interoperability with the standard, we had long discussions about the right approach. The BCF API expertise, implementation, and analysis of its shortcomings are primarily the work of Oliver Schulz. Together, we went through best practices for ontology publishing. The starting point for this was a presentation [Bonduel, 2021] by Mathias Bonduel that I saw on the theme. We extended this by looking at the fields recommendations on the subject. I did the work to

publish the ontology online.

Table 3.1. The work of Schulz et al. [2021] in the Hevner et al. [2004] design science research framework

Guideline 1: Design as an artifact
A published ontology description for the buildingSMART BIM Collaboration Format (BCF): bcfOWL. The ontology is presented using the W3C Web Ontology Language (OWL) as a proof of concept. The documented use of the best practices for creating the ontology can be applied to other standards in the field as well.

Guideline 2: Problem relevance
BCF is typically used in issue communication. The linked data version has the same purpose. The business problem is: when the organization starts using Linked Data for building information, can there also be a Linked Data counterpart for the BCF data so that the data is interoperable? Furthermore, are there ways to present the data so that all issued data can be used as one consistent knowledge base? The second business problem is how to use the collected issue information for further analysis, e.g., to monitor maintenance and assets. In the semantic web domain, there was no ontology to express BCF data in connection with the BIM ontologies (ifcOWL and the BOT ontologies).

The constituent user community of the artefact are the researchers of the Linked Building Data Community, but also developers in the AEC sector that want to use BCF data in their software, particularly when already using other Linked Data in the project.
The solution belongs to the LBD ontologies problem class.
This work is an enabler for the following BIM use cases: BCF Communication (buildingSMART use-case GUID:B30EDF79-7A11-4E0B-9B20-104B44AD4A51 [buildingSMART International, 2022]), Monitor Maintenance, and Monitor Assets [The Pennsylvania State University, 2013].

Guideline 3: Design evaluation

Hevner et al. Design Research Method: Descriptive Scenario was used to evaluate how data presented using the presented ontology can be queried.

Guideline 4: Research contributions

There was no ontology to use in the context of the semantic web and Linked Building Data.
The main contribution was to show how BCF data can be expressed as Linked Data following the best practices of ontology publication and to harness the potential of Linked Data in the context. It enables the Linked Building Data community to access BCF data content as Linked Data. Therefore BCF standards were not mechanically translated but handcrafted to express the same concepts in a compatible way to keep the interoperability with the existing BCF standard. The result is a proposal of a data format, meaning it is not meant to replace the existing standards but to offer research results to show a potential analysed direction of how the standard could be extended.

Although an ontology description implies that the data can be used as Linked Data, the data assertions in Resource Description Framework are interface neutral. So, the work does not exclude using the BCF API interface with it.

Guideline 5: Research rigor

The work was peer-reviewed. Also, the OWL ontology description, the ontology website, test data and source code were made available.

Guideline 6: Design as a search process

We followed the best practices presented in the article to guide creating the ontology. A special focus was on that is not mechanically translated from the BCF API standard but use the expressiveness and advantages of the semantic web (e.g., using URLs instead of global identifiers (GUID) for linking) while keeping the compatibility with the existing BCF standards. The aspects and the presented use case guided our search for a solution in the problem domain.

Guideline 7: Communication of research

Proceedings of the 9th Linked Data in Architecture and Construction Workshop, Luxembourg, Luxembourg, October 11-13, 2021
It was also presented for the buildingSMART BCF working group.

The article was made available as an open-access publication under the Creative Commons License Attribution 4.0 International (CC BY 4.0 [1]). The Linked Data Architecture and Construction Workshop 2020 editors, RWTH Aachen, CEUR-WS and archiving partners were given irrevocable rights to make the work available. The work was originally published in [Schulz et al., 2021]. The work presented here is based on our final draft of the article.

Oliver Schulz Department of Design Computation, RWTH Aachen University, Aachen, Germany, schulz@dc.rwth-aachen.de
Jyrki Oraskari Department of Design Computation, RWTH Aachen University, Aachen, Germany, Jyrki.Oraskari@dc.rwth-aachen.de
Jakob Beetz Department of Design Computation, RWTH Aachen University, Aachen, Germany, j.beetz@caad.arch.rwth-aachen.de

Abstract: The BIM Collaboration Format (BCF) is a buildingSMART standard used to exchange issues in a digital model between heterogeneous software applications and planners. Although the BCF issues are spatially located in a model and provide links to building elements, the format is only loosely connected to the actual BIM model. While the format is well suited for its original use case, it lacks the flexibility to retrieve information from the BCF in conjunction with the BIM model.

In this paper, we introduce the BIM Collaboration Format Ontology (*bcfOWL*), which translates the format to the Semantic Web and harnesses the expressive richness of OWL. We present the structure of the ontology and highlight the differences with existing implementations of BCF. Using example queries, we show how extended relationships between a BIM model and BCF information are enabled by *bcfOWL*. We show that by transferring BCF to Linked Data, the format's applicability is enhanced without losing compatibility with existing implementations and workflows. In doing so, the ontology aims to integrate into the Linked Building Data environment and facilitates access to synergies between heterogeneous building data.

keywords: BCF, SPARQL, OWL, REST API, XML,Linked Data

3.1.1 Introduction

Issue management is an integral part of most BIM processes in the Architecture, Construction, Engineering, Owner and Operator (AECOO) industry. Today this is often implemented by using the BIM Collaboration Format (BCF) of the

[1] https://creativecommons.org/licenses/by/4.0/legalcode

buildingSMART organization. It represents a standardized way of communicating issues between heterogeneous software applications. There are two standard approaches for exchanging data: a file-based workflow, where *Issues* are written into a ZIP container using the BCF XML[2] schema, and a server-based approach, which uses the BCF API[3] specification for the data exchange. It is based on the principles of a RESTful API. It allows users of different disciplines using different software to communicate the reported *Issues* online, eliminating the need to exchange files via email or a data device. [van Berlo and Krijnen, 2014]

Common use cases for such *Issues* include clash detection during partial domain model integration and change or information request in the design phase of a building. They often result in hundreds of *Issues* in a single information handover or integration scenario. For such common scenarios, the management of *Issues* captured in individual files is highly impractical and queries for specific data are challenging to implement via the API.

Over the last years, there have been many approaches to compose Linked Building Data. The ifcOWL ontology [Beetz et al., 2009, Pauwels and Terkaj, 2016] was designed to be as equivalent as possible a Semantic Web representation of the IFC EXPRESS schema. IfcWoD was designed to express the object-oriented constraints of the IFC schema as an OWL ontology [Mendes de Farias et al., 2015]. BimSPARQL [Zhang et al., 2018] proposed an extension for SPARQL Query Language (SPARQL) to query geometrical and BIM specific data of IFC models. Like ifcWoD and BimSPARQL, SimpleBIM was an attempt to create a more accessible view on ifcOWL data. [Pauwels and Roxin, 2016]. Recently, the evolution culminated in the development of the Building Topology Ontology (BOT) [Rasmussen et al., 2020], which reduced the extensiveness compared to ifcOWL and enabled modular ontology design when describing buildings in the Linked Data context.

The Linked Building Data Community Group (LBD-CG)[4] of the World Wide Web Consortium (W3C) is devoted to advance the adoption of the Linked Data and Semantic Web technologies in AECOO. The vision is to enable web-based information exchange and workflows, provide open and versatile standards for different domains, and facilitate distributed data integration. [Rasmussen et al., 2020]

In this paper, we present the BIM Collaboration Format Ontology (*bcfOWL*)[5] that shares the principles of Linked Building Data, and which provides a way to create BCF information that is interlinked in the context of a building. We

[2] https://github.com/buildingSMART/BCF-XML
[3] https://github.com/buildingSMART/BCF-API
[4] https://www.w3.org/community/lbd/
[5] http://lbd.arch.rwth-aachen.de/bcfOWL#

describe the design principles used and demonstrate the benefits of the approach by example queries. We discuss the differences from the BCF API server definition and argue how the representation of *Issues* as semantic graphs are beneficial in continuous information integration scenarios in common BIM-based workflows. The underlying research questions are:

1. How can *Issues* that arise in common workflows be integrated and linked with the BIM workflow?
2. How to express the core ideas of BCF by following the best practices of ontology design and still keeping the result compatible with the original BCF API and BCF XML standards?

The paper is structured as follows: The first section describes the current implementations of BCF and highlights a first approach of converting BCF to an ontology. The following section deals with the newly created ontology, *bcfOWL*, and focuses on which design principles and best practices were considered when creating it, and how it compares to the current implementations of BCF. In section 4, we provide example use-cases and show how SPARQL can be used to query BCF information. We discuss the results of *bcfOWL*, conclude the paper, and give an overview of future work in section 6.

3.1.2 BIM collaboration Formats

BCF XML

BCF was initially developed by the companies Solibri and Tekla, and the Institute of Applied Building Informatics to exchange problems within a digital building model between different software applications.[6] The format is often used in the area of issue management and collision checking. The original version of BCF is based on an XML format, and the *Issues* are organized in sub-folders within a ZIP file. This file can then be distributed and imported into other software applications that support the standard. The BCF XML format also allows round-tripping, where the same file can be extended with additional *Issues*, and existing *Issues* can be modified. Thereby preventing that the *Issues* are stored over several files, and thus a clear structure is lost.

[6] https://technical.buildingsmart.org/standards/bcf/

BCF API

Although this round-tripping of the BCF XML files already reduces ambiguity and helps stricter, machine-readable representations of *Issues*, the exchange of files via e-mail or data storage media is a weak point of related processes and workflows. [van Berlo and Krijnen, 2014] To address these shortcomings, the BCF API was developed to replace the file-based exchange of *Issues* in the digital building. This is achieved through a server infrastructure that enables communication via a RESTful API with client applications. Rather than in XML, the data is captured and processed using the JSON format. Providers of BCF servers enable the retrieval, viewing and editing of *Issues* by external software applications or on websites, which allows a wide range of stakeholders within a project to be integrated into the planning process.

Common Data Environments

The 2.1 version of the BCF API contains a *Public Service*, responsible for a rudimentary *User Management*, and a *Documents Service*, which can be used to interchange BIM models via the BCF server and link an *Issue* to a specific *Document* or *File*. Version 3.0 already separates the *Public Services* to an API, called the Foundation API and serves as a common base layer on an upcoming multi-tier architecture for open standards for Common Data Environments[7] (openCDE APIs). For the *Document Service*, a Documents API is under development as well. These APIs are intended to enable a standardized, vendor-independent exchange of BIM models and their associated data. BCF is an integral part of this infrastructure, as it can be used within an openCDE to document issues and create change requests across multiple documents.

General structure of BCF

Although BCF XML and BCF API are based on different data serialization formats, both variants are compatible on a structural level. They are based on the basic concepts of *Projects*, *Topics*, *Comments*, and *Viewpoints*, which are organized hierarchically. The combination of *Topics*, *Viewpoints*, and *Comments* is often referred to as an *Issue*. When talking about an Issue, this is to be equivalent to the Topic, which is the connecting element for the Viewpoints and Comments. In the

[7] https://technical.buildingsmart.org/projects/opencde-api/

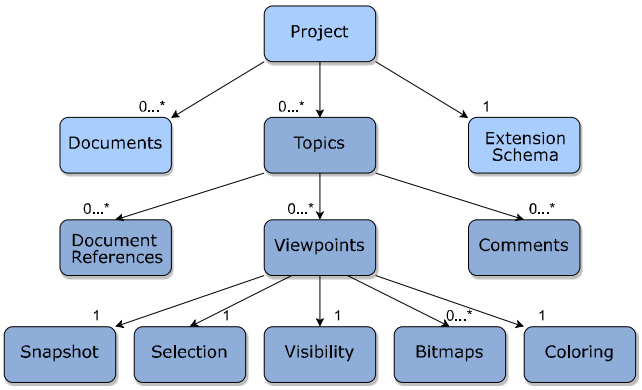


Fig. 3.3. General structure of the BCF. The dark blue boxes form the *Issue*. Each hierarchy level represents a request to the server in the BCF API.

file structure of the BCF XML, the term *Markup* is used. The general structure can be reviewed in Figure 3.3.

Except for free text parameters, the *Topics* must be filled with data defined in the *Extension Schema*. The concept of *Viewpoints* establishes a spatial link to the BIM model and conveys the information about which parts of the building are visible or selected in the respective camera view. The position and rotations are specified in a Cartesian coordinate system with vectors (X|Y|Z).

Drawbacks of current BCF implementations

In [Schulz and Beetz, 2021], the hierarchy was identified as one of the key problems of the BCF API since the nested structure demands a specific way of requesting information. If e.g. all *Comments* created by a particular person are requested from the server, all Topics must be loaded first, and their IDs can then be used to request the *Comments* associated with the *Topics*. Thus an overhead of requests to a server is created. The same matter applies to the *Viewpoints*, which contain sub-concepts that have to be requested from the server separately. This is a common problem when working with REST APIs and is regarded as the "N+1 problem"[8].

Furthermore, the connection to BIM happens via two parameters in the BCF Format. On the one hand, the *Issues* are spatially linked with the model via location and rotation in the *Viewpoints*. On the other hand, the *Components* sub-concept can link to a building element via Globally Unique Identifier (GUID). Nevertheless, currently neither allows to query BCF information in a spatial

[8] https://restfulapi.net/rest-api-n-1-problem/

context or a context of the accompanying building elements referenced by the *Issue*. Already downloaded *Issues* and BCF XML could be queried with custom-designed functions in such a way, but just in the context of the BCF information currently loaded into the model.

BCF in Linked Data

A first approach of mapping the BCF format to Linked Data can be seen in [Bernal Ferrer, 2017] where a BCF ontology was created by converting the XSD schema of BCF XML, using the XsdImport plugin for TopBraid Composer. The created ontology is very close to the file-based principles of BCF XML. It thereby uses concepts like the *Markup* and multiple properties that are redundant for an online distributed approach and add to the complexity of the ontology. Moreover, most ontology properties are converted into Datatype Properties and used as a string. Cardinality constraints for the properties are not provided for the ontology classes. Furthermore, the ontology was not published, and it remained as a test use case in a larger prototype.

3.1.3 bcfOWL

bcfOWL aims to lift BCF to the Linked Data domain and thereby enable the format to be put in a larger context within the Linked Building Data, including the BOT and ifcOWL ontology. The conversion should not just be a translation of the BCF API or the BCF XML but specifically tailored to the needs in a Linked Data context. At the same time, compatibility with existing formats should not be jeopardized. Therefore we considered guides, checkers, and best practices to achieve this goal:

- In [Hammar, 2021], the author argues that the complexity of classes inside an ontology should be kept simple for the ontology to be usable by implementers and system developers. This is exemplified by Container classes that hold no actual data and thus should be avoided when designing an ontology.
- In [Poveda-Villalón et al., 2013], the authors define good and bad practices, referred to as pitfalls when publishing vocabulary on the web and provide detection methods for these practices.
- In [Poveda-Villalón et al., 2014], an online tool is provided to scan ontologies for these pitfalls, which was used to validate *bcfOWL*.
- Moreover, best practices were also defined by the World Wide Web

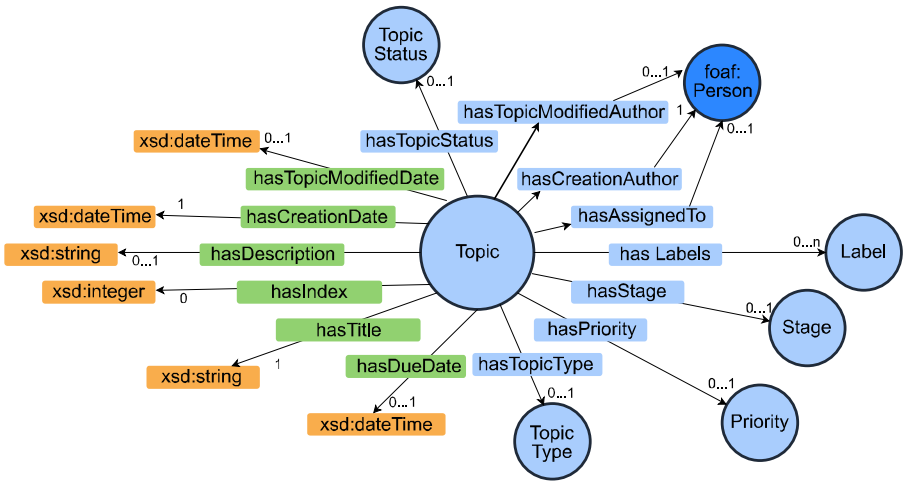


Fig. 3.4. A snippet from the ontology, displaying the *Topic* as a graph

Consortium[9] (W3C) that were also put into consideration when designing *bcfOWL*.

Since the work aims to translate the BCF format into an ontology, the main difference between the two formats is the description in the respective schema. However, a notable characteristic is that BCF exists in both a file-based and a server-based approach (Section 2). The *bcfOWL* approach exists between these two and exploits both variants.

bcfOWL is covering the core abstraction and concepts of the BCF standard relating the Issue Management. The subsets of *Event Services*, *File Services*, and *Document Services* are not included into this core ontology. Supplementary transfers of these modules into ontologies would also have to be provided and form a Linked Data Common Data Environment, together with *bcfOWL*.

Projects

The *Projects* are a core element of BCF responsible for assigning each *Issue* to a project, and thus a specific building. The link to the *Project* in the BCF API happens via URL (Listing 3.1), where the *Projects* GUID is provided. In BCF-XML, the *Project* is defined per file and there is no further assignment to it. Since resources from *bcfOWL* are not queried via parameters in the URL, and a file-based approach is not the sole scope of our work, the link to the *Project* has to be provided in the

[9] https://www.w3.org/TR/ld-bp/

classes of the *Topics*, *Comments*, and *Viewpoints*. (Listing 3.2) The *Project* itself links to the *Extension Schema*, described in the following section.

```
GET /bcf/3.0/projects/{project_id}/topics
```

Listing 3.1. GET request to a BCF server. The *Project* is provided as parameter

```
@prefix bcfOWL: <http://lbd.arch.rwth-aachen.de/bcfOWL/> .

bcfOWL:Comment rdf:type owl:Class ;
  rdfs:subClassOf
    [ rdf:type owl:Restriction ;
      owl:onProperty bcfOWL:hasProject ;
      owl:cardinality "1"^^xsd:int ] ,
    [...] .
```

Listing 3.2. The *Project* has to be provided for the *Topics*, *Viewpoints*, and *Comments* and therefore a cardinality of 1 is specified.

Extensions

In the BCF context, the *Extensions* are used to create project-specific parameters when inserting new *Issues*. In a *Project*, parameters such as *Status*, *Labels*, and *Users* are defined in advance in the *Extensions*. With parameters such as *Title* or *Comment*, on the other hand, free texts are used. Especially with the BCF API, the correct parameters are controlled by the server during the upload since it only allows the parameters mapped in the *Project's Extensions*.

bcfOWL also provides the possibility to define these *Extensions* to be used in the *Projects*. However, since the allowed values can change from project to project, we provide only a generic and backwards-compatible way to define the property sets. The extension and validation mechanisms will be studied further in future work.

Topics, Comments, and Viewpoint

Same as in the *Projects*, the *Topics*, *Comments*, and *Viewpoints* need to be linked to each other to generate a complete *Issue*. In the BCF API, these links are primarily provided in the URL parameters. For example, the link of the *Viewpoint* to its *Topic* is only provided by the URL and is not part of the JSON schema. In the BCF XML, this link is provided by the *Markup*. In *bcfOWL*, these connections are provided in the respective classes of the *Comments* and *Viewpoints* as Object Properties, linking to the *Topic*. The *Markup* concept was discarded, thus reducing

the complexity of the ontology. This is in line with BCF API, where it had been left out.

Issue transformation

The transformation of an *Issue* is an integral part of the principles of the BCF. It is described with a *Camera View Point* in a Cartesian coordinate system, a *Camera Direction*, an *Camera Up Vector* for describing the rotation of the view in the coordinate system, and an *Aspect Ratio*. Depending on if the camera is described as an *Orthogonal Camera* or a *Perspective Camera*, a *View to World-Scale* or a *Field Of View* has to be provided. The transformation of the camera is described in the *Viewpoint* of an *Issue*. To express the vectors and points, *bcfOWL* relies on the representation in well-known-text (WKT) as proposed for the ifcOWL format in [Pauwels et al., 2017]. Using WKT, compatibility with GeoSPARQL [Battle and Kolas, 2011] should be achieved, a standard of the Open Geospatial Consortium (OGC). *bcfOWL* is also inspired by BimSPARQL [Zhang et al., 2018], which describes points with the WKT. It is an extension of the SPARQL query language, with a specific focus on BIM and ifcOWL. Both query languages allow to query data in a spatial context from the graph and apply constraints to the query like a maximum distance between two points [Zhang et al., 2018].

Snapshots

In BCF, most *Issues* are accompanied by a *Snapshot* from the source application to help understand the problem described since not all models are loaded in every software in which an *Issue* is considered. Thus, the problem cannot be verified without further context. In the BCF XML, the images are stored as PNG or JPEG in the sub-folder with the *Markup*. They are linked to the *Viewpoint* via an GUID. In the BCF API, when a new *Snapshot* is created, the image is passed in base64 format, included in a JSON. When requesting a *Snapshot*, the image is passed as a binary file from the BCF server. Since including images with base64 strings in RDF leads to extensive and slow graphs, the locations of the images are linked in *bcfOWL* by URI. For compatibility reasons, the URI is the same as in the BCF API approach and the up- and download of the *Snapshots* can be achieved by simple GET and POST requests.

3.1.4 Example use-case

By introducing BCF to OWL, we can now use this format with more advanced queries, where the user can specify what data should be fetched from a server. In this section, we provide an example use case for *bcfOWL*. The queries were tested with Apache Jena Fuseki.

Querying BCF information

While a request with the BCF API, for example, for a *Topic*, can be handled via a simple GET request (Listing 3.1), the same request in the SPARQL context is more extensive (Listing 3.3). However, this problem can be partially circumvented by using template functions to map the standard requests to the BCF server used in the REST API.

```
PREFIX bcfOWL: <http://lbd.arch.rwth-aachen.de/bcfOWL/>

SELECT ?s ?p ?o
WHERE {
   ?s a bcfOWL:Topic ;
     ?p ?o .
}
```

Listing 3.3. Sample SPARQL query for getting *Topics*.

Query by Selection

An important feature to bring building data closer together with BCF is to query building elements to see if *Issues* related to them already exist. Software applications such as Solibri discovered the need for such information and have integrated a corresponding feature into their application. However, the feature only supports a local (offline) context and cannot align the *Issues* for the building elements with the *Issues* stored on a server without downloading them first. Specific queries like these are not possible with the current API, or only to a limited extent, because the existing hierarchy requires all *Issues* on the server to be searched first to get the IDs of the building elements. [Schulz and Beetz, 2021] However, by implementing the *bcfOWL* ontology it is now possible with a single request. By providing the ID of a selected building element, an application can send a query to the database with the searched for GUID and the server responses with all *Topics* referring to the GUID.

```
PREFIX bcfOWL: <http://lbd.arch.rwth-aachen.de/bcfOWL/>
```

```
SELECT ?topic
WHERE {
  ?topic a bcfOWL:Topic .
  ?viewpoint a bcfOWL:Viewpoint ;
         bcfOWL:hasTopic ?topic ;
         bcfOWL:hasSelection ?selection .
  ?selection bcfOWL:hasComponent ?component .
  ?component bcfOWL:hasIfcGuid "1ZwJH$85D3YQG5AK5ER1Wc" .
}
```

Listing 3.4. SPARQL example for querying *Topics* with a specific GUID in the selection.

Referencing Building Elements

Referencing a Building Element in the *Selection* or the *Visibility* can either be achieved by a GUID, an Authoring Tool ID, or by linking an object (by a dereferenceable URL). The first two options seem to be less in line with Semantic Web concepts but is helpful if the IFC and its elements do not exist in a Linked Data context. This ensures compatibility with a wide range of formats while linking with a Linked Data object is still possible.

```
inst:Element_1 a bcfOWL:Component;
   bcfOWL:hasIfcElement inst:BotElement_1;
   bcfOWL:hasIfcGuid "1zMXMGg3H1MOhxQ9qjUS4B" ;
   bcfOWL:hasAuthoringToolId "312213" ;
   bcfOWL:hasOriginatingSystem inst:ExampleCAD .
```

Listing 3.5. Components provide a link and/or the GUID of the element.

3.1.5 Discussion

The examples show that *bcfOWL* can address use cases beyond the capabilities of the current BCF formats. It was demonstrated by the example of querying a graph database for building elements for which *Issues* exist. The connection to the building elements can be described with the ontologies of BOT and ifcOWL via URIs and a conventional IFC or native BIM model via GUID. *bcfOWL* can thus be integrated into existing workflows. Although individual parts have been excluded from *bcfOWL* , the underlying concepts remains the same. A conversion of *bcfOWL* data to BCF XML or BCF API is still applicable.

However, this extended functionality comes at the cost of using the SPARQL language, which can be a hurdle for users and developers [Werbrouck et al., 2019]. Especially when compared to requests to a REST service, requests via SPARQL

become more complex and require a deep understanding of the format to properly represent BCF in a graph while maintaining compatibility with the existing format. By the absence of validation of newly created resources, which can cause data to be uploaded incorrectly in the context of the BCF format, this is even intensified. Furthermore, the GeoSPARQL ontology was used for describing the position of the Issues in a three-dimensional context with WKT. However, GeoSPARQL in its current version is mainly limited to a spatial context in a 2D domain. Although three-dimensional vectors can be represented in the WKT system, only the XY coordinates are checked for spatial queries to the graphs. Nevertheless, use cases for three-dimensional queries have been recognized [Abhayaratna et al., 2020] by the OGC and have been included as a milestone[10] in the further development of the format. Because of the limitation no spatial queries were tested for *bcfOWL*.

Since the concept of inverse relationships does not exist in BCF, it has not been incorporated into the ontology at this time. It should be discussed in the future whether the advantages of inverse relations justify a more complex ontology.

3.1.6 Conclusion and future work

We have shown in this paper that a transformation of BCF into a Linked Data approach is viable and opens new possibilities for the format. While BCF XML and BCF API applications can actualize and interpret the indirect GUID references of a BIM model when the BIM models are loaded to the memory, the proposed Linked Data approach allows better integration. The new data format makes it easier to retrieve relevant issue data and attach and track provenance information over different life-cycle stages. Further, deeper analysis is left for future work. Using Linked Data, the BCF and BIM data models can be integrated into a shared knowledge graph, making it possible to harness the power of semantic queries, e.g. using a SPARQL engine. Spatial localization is achieved via the WKT format, but further emphasis needs to be placed on enabling and testing spatial queries.

bcfOWL is by no means intended to replace the existing formats but merely to add further possibilities. A conceivable scenario would be to use *bcfOWL* as an internal data format for a BCF server, which can then be exposed via a SPARQL endpoint, and via an intermediary service using the BCF API. Thus the strengths and weaknesses of the respective formats could be balanced, and compatibility to already existing approaches could remain. The proposed method allows using Linked Building Data content like assertions that use ifcOWL or the BOT ontologies connected with or beside regular IFC and BIM models. The data can also be linked to available Linked Data sources.

[10] https://github.com/opengeospatial/ogc-geosparql/issues/19

When creating *bcfOWL*, we focused on the core functions necessary when creating *Issues* within a building. Subjects such as handling and linking to and from *Documents* and *Files* and user management were excluded from the *bcfOWL* definition. Like bundling the different buildingSMART APIs, a Linked Data based Open CDE infrastructure could also be applied here. First approaches to such considerations can already be observed in [Werbrouck et al., 2020]. Also, the tracking of the BCF *events* is not integrated into *bcfOWL*, and further work should show how these can be used in a meaningful way in the Linked Data context. BCF *Events* have an essential role in Issue management, as they can be used to track who changed what data in a BCF and when.

Another starting point in conjunction with *bcfOWL* could be a connection to the Shapes Constraint Language[11] (SHACL), which is used to validate RDF graphs. Since in the BCF context, *Projects* are provided with *Extensions*, and thus no parameters should be used in these *Projects* that do not appear in them, a SHACL implementation would be a potential complement to *bcfOWL*.

By introducing BCF into the Semantic Web, we hope that the application possibilities of the format will be extended. This includes increased possibilities to analyze BIM processes and harness the underlying data graphs to interlink and trace dependencies, provenance, and historical evolvement of *Issues*. We also see potentials in the interoperable interlinking of *Issues* with heterogeneous, non-IFC, and legacy information types for multi-model containers, e.g., the ICDD and OpenCDE-APIs. *bcfOWL* should enable synergies with other ontologies such as ifcOWL and the Building Topology Ontology and thus contribute to the goal of Linked Building Data.

3.1.7 Acknowledgments

This research had been funded by the EU through the H2020 project BIM4REN.

[11] https://www.w3.org/TR/shacl/

References

Joseph Abhayaratna, Linda Brink, Nicholas Car, Rob Atkinson, Timo Homburg, Frans Knibbe, Dr. Kris Mcglinn, Anna Wagner, Mathias Bonduel, Mads Holten Rasmussen, and Florian Thiery. OGC Benefits of Representing Spatial Data Using Semantic and Graph Technologies. Technical report, Open Geospatial Consortium, oct 2020.

Robert Battle and Dave Kolas. Geosparql: enabling a geospatial semantic web. *Semantic Web Journal*, 3(4):355–370, 2011.

Jakob Beetz, Jos van Leeuwen, and de Vries. IfcOWL: A case of transforming EXPRESS schemas into ontologies. *AI EDAM*, 23:89–101, February 2009. doi: 10.1017/S0890060409000122.

F. J. Bernal Ferrer. Internal design validation attestation in BIM models: a combination of semantic web technologies and BIM collaboration format. Master's thesis, Eindhoven University of Technology, Eindhoven, October 2017.

Karl Hammar. Ontology design principles for model-driven applications. In *Advances in Pattern-Based Ontology Engineering*, pages 273–278. IOS Press, 2021.

Tarcisio Mendes de Farias, Ana-Maria Roxin, and Christophe Nicolle. IfcWoD, Semantically Adapting IFC Model Relations into OWL Properties. In *Proc. of the 32nd CIB W78 Conference 2015, 27th-29th October 2015, Eindhoven, The Netherlands*, pages 175–185, 2015.

Pieter Pauwels and Anna Roxin. SimpleBIM : From full ifcOWL graphs to simplified building graphs. In Symeon Christodoulou and Raimar Scherer, editors, *eWork and eBusiness in Architecture, Engineering and Construction (ECPPM)*, pages 11–18, Limassol, Cyprus, 2016. CRC Press. ISBN 1138032808.

Pieter Pauwels and Walter Terkaj. Express to owl for construction industry: Towards a recommendable and usable ifcowl ontology. *Automation in Construction*, 63:100–133, 2016. Elsevier.

Pieter Pauwels, Thomas Krijnen, Walter Terkaj, and Jakob Beetz. Enhancing the ifcOWL ontology with an alternative representation for geometric data. *AUTOMATION IN CONSTRUCTION*, 80:77–94, 2017. ISSN 0926-5805, 1872-7891. doi: 10.1016/j.autcon.2017.03.001. Publisher: ELSEVIER.

María Poveda-Villalón, Bernard Vatant, María del Carmen Suárez-Figueroa, and Asunción Gómez-Pérez. Detecting good practices and pitfalls when publishing vocabularies on the web. In *WOP*, 2013.

María Poveda-Villalón, Asunción Gómez-Pérez, and Mari Carmen Suárez-Figueroa. OOPS! (OntOlogy Pitfall Scanner!): An On-line Tool for Ontology Evaluation. *International Journal on Semantic Web and Information Systems (IJSWIS)*, 10(2):7–34, 2014. IGI Global.

Mads Holten Rasmussen, Maxime Lefrançois, Georg Schneider, and Pieter Pauwels. Bot: the building topology ontology of the w3c linked building data group. *Semantic Web*, 11 2020. doi: 10.3233/SW-200385.

Oliver Schulz and Jakob Beetz. Image-documentation of existing buildings using a server-based bim collaboration format workflow. In Jimmy Abualdenien, André Borrmann, Lucian-Constantin Ungureanu, and Timo Hartmann, editors, *EG-ICE 2021 Workshop on Intelligent Computing in Engineering*, pages 108–107, 2021. ISBN 978-3-7983-3212-6.

Léon van Berlo and Thomas Krijnen. Using the bim collaboration format in a server based workflow. *Procedia Environmental Sciences*, 22:325–332, 2014. Elsevier.

Jeroen Werbrouck, Madhumitha Senthilvel, Jakob Beetz, and Pieter Pauwels. Querying heterogeneous linked building datasets with context-expanded graphql queries. In *7th Linked Data in Architecture and Construction Workshop*, pages 21–34, 2019.

Jeroen Werbrouck, Ruben Taelman, Ruben Verborgh, Pieter Pauwels, Jakob Beetz, and Erik Mannens. Pattern-based access control in a decentralised collaboration environment. In *Proceedings of the 8th Linked Data in Architecture and Construction Workshop*. CEUR-WS. org, 2020.

Chi Zhang, Jakob Beetz, and de Vries. BimSPARQL: Domain-specific functional SPARQL extensions for querying RDF building data. *Semantic Web*, pages 1–27, August 2018. http://doi.org/10.3233/SW-180297.

3.2 Towards describing version history of BCF data in the Semantic Web (LDAC 2022)

Image a scenario where the ventilation system's air terminal breaks down twice during the summer. The facility manager of the building wants to oversee the history of the terminal to see if there have been any remarks or perceived exceptions related to that terminal earlier.

Hypothesis 3.2 claimed that version information of buildingSMART BIM Collaboration Format (BCF) could be expressed more effectively when the data is Linked Data compared to the standard BCF approaches. The work shows four methods and ways to update and query information. We showed that the state of the values could be queried using the SPARQL query language at any time point. The same information could be derived using the BCF API Topics Events and Comments Events services. That would need JSON data parsing, developing an internal data format, and an algorithm to process the values. The extra steps required imply that the proposed new ways to process the data have an advantage.

From a data management perspective, the primary objective entailed devising a means to express the data's change history and facilitate its consumption. Given our emphasis on edits of the data, we deemed it imperative for that to hold a paramount status within the data model. Consequently, options such as employing namespaces or RDF storage extensions were precluded as potential solutions.

Table 3.2 summarises the use of the design science research method in this research.

My contribution to the paper: This conference article was written in collaboration with Oliver Schulz. Oliver Schulz and I wrote an equal part of the manuscript. In the research, my main contributions were creating methods

to compute numerical measurements to compare the methods when applying them to the use cases. My work was also the SPARQL queries that we analysed and reviewed together. Otherwise, the work results from close group work and lengthy discussions.

Table 3.2. The work of Oraskari et al. [2022a] in the Hevner et al. [2004] design science research framework

Guideline 1: Design as an artifact
We created four data patterns of the graph to express versioning for BIM Collaboration Format issue data represented as semantic web. SPARQL queries were written and published for each approach.

Guideline 2: Problem relevance
BCF issue data contains a rich set of information and could be used as a consistent source of data for a knowledge base. The business problem is whether the change history can be expressed in a form suitable for further analysis, e.g., to monitor maintenance and assets. There is an existing event logging definition for BCF API, and in the Linked Building Data Context, the Ontology for Property Management (OPM) [Holten Rasmussen et al., 2018]. No ontology describing BCF changes existed. If the BCF version data were expressed as RDF, the history data in all its richness would be available for queries.

The constituent user community of the artefact is the researchers or developers that use or aim to use BIM Collaboration Format issue data as Linked Data or are interested in expressing versioning of another domain data.
The solution belongs to the version control and change log problem classes.
This work is an enabler for the following BIM use cases: BCF Communication (buildingSMART use-case GUID:ăăB30EDF79-7A11-4E0B-9B20-104B44AD4A51 [buildingSMART International, 2022]), Monitor Maintenance, and Monitor Assets The Pennsylvania State University [2013].

Guideline 3: Design evaluation

Hevner et al. Design Research Evaluation Method: A Descriptive Scenario was used to evaluate the utility of the proposed four approaches.
In Analytical Static Analysis, SPARQL queries to add, read, and update the data were created and evaluated. We made a qualitative comparison of the four methods. The summary is shown in Table 3.3.

Guideline 4: Research contributions

Generalization of the results: We have focused on BCF data and the bcfOWL ontology in the article. This is part of the general problem class of LBD data versioning.
We used the following requirements in the design: 1. triples should only be added, and 2. backward compatibility to existing BCF standards is beneficial and so used here. 3. The version data should be part of the graph to facilitate machine inferences.
The approach is generic. Although the problem domain affects the comparison results of the approaches, the presented methods can be in a straightforward manner applied in other scenarios.

Guideline 5: Research rigor

The work was peer-reviewed. Also, the data and associated queries were made available in our GitHub repository[12].

Guideline 6: Design as a search process

In the design, we followed the principle that only triples were allowed to be added to the models. This and the presented use case guided our search for a solution in the problem domain.
We implemented versioning in three major styles: the Ontology for Property Management (OPM) style, the BCF API event log style, and two RDF star implementations. This study is a search step to find the best approach for future research.

Guideline 7: Communication of research

LDAC 2022: 10th Linked Data in Architecture and Construction Workshop, May 29, 2022, Hersonissos, Greece. CEUR Workshop Proceedings (CEUR-WS.org)

[12] https://github.com/Design-Computation-RWTH/LDAC2022_Dataset

As agreed with the LDAC 2022 conference organizers, the work was made available as an open-access publication under the Creative Commons License Attribution 4.0 International (CC BY 4.0), available at https://creativecommons.org/licenses/by/4.0/legalcode. This work is not part of any other thesis. The work is originally published in [Oraskari et al., 2022a]. The content presented here is based on our final draft of the article.

Jyrki Oraskari Department of Design Computation, RWTH Aachen University, Aachen, Germany, Jyrki.Oraskari@dc.rwth-aachen.de
Oliver Schulz Department of Design Computation, RWTH Aachen University, Aachen, Germany, schulz@dc.rwth-aachen.de
Jakob Beetz Department of Design Computation, RWTH Aachen University, Aachen, Germany, j.beetz@caad.arch.rwth-aachen.de

Abstract: The buildingSMARTs BIM Collaboration Format (BCF) is a vendor-neutral standard to communicate BIM-based issues. The aspirations are well in line with open BIM and the Linked Building Data (LBD) community to use open data standards in Architecture, Engineering and Construction industry. Our previous research introduced the bcfOWL ontology to express BCF data using the open standards of the World Wide Web Consortium (W3C) as part of the LBD. The current server-based BCF-approach (BCF API) expresses changes in the model as event logs that are indirectly described. bcfOWL does not yet provide an equivalent implementation.

This paper presents four different approaches for expressing these temporal changes as Linked Data. Two RDF-star approaches and a state construction inspired by the Ontology for Property Management (OPM) are introduced. Moreover, we show an event system method that is close to the original BCF API. To compare the approaches, queries to get the current data and search the history are provided on an external repository. Finally, the results and their adaptation are assessed.

keywords: BCF, RDF-star, States, Version history, Linked Data

3.2.1 Introduction

The traceability of editions and design choices made over time is an integral part of the Architecture, Engineering and Construction (AEC) industry. Time after time, construction projects end up in court, where the responsibility for issues in the building or the planning process is litigated.

The BIM Collaboration Format (BCF) is mainly used in the AEC industry to communicate issues in the digital models. In the standard, an issue is described by a problem description (Topic) and an arbitrary number of Comments and virtual camera positions (Viewpoints) that locate the problem. It helps users assign responsibilities for issues and monitor their completion status. As buildingSMART stated in [buildingSMART International, Ltd., 2022], BCF can be used to document quality assurance and quality checking for inter-domain coordination, annotating design alternatives, bidding items, expressing cost- or supplier information, reporting facility changes, and saving owner notes.

The bcfOWL ontology [sch, 2021] was previously developed by the authors. It enables expressing BCF information in the Linked Building Data (LBD) context. The aspiration was to create a proposal for an ontology for the core concepts of the BCF standard to represent the information in the Semantic Web using the best practices available. Thereby, bcfOWL is not intended to replace current integrations of the standard but rather to provide a data model that is self-documenting, queryable, machine-readable and extensible. With bcfOWL, it is still possible to have a web service that connects to the graph and provides the data via JSON responses.

In its server-based format (BCF API[13]), BCF comes with Event Services (REST API endpoints) that list changes made to the Topics and Comments. Since it was regarded as out of scope, bcfOWL does not yet provide an equivalent implementation.

In this paper, we study different approaches for including ways to express the history and evolution of issues in bcfOWL. The focal points are in expressing the source or provenance data, i.e. traceability of changes, changes made, the ability to reason behind the changes and enable queries on the history data. Although we concentrate on the BCF Topic - the core concept for describing issues in the standard - the study results should also apply to other BCF concepts. As non-repudiation is an essential aspect when describing the history of the data, it is one of this work's goals that the existing data content - once created - is not changed. Therefore, the constraint arises that the structure should only be based on addition, not removal operations.

This paper is organized as follows: Section 2 focuses on the current research around relating work. In Section 3, we introduce the different approaches for describing the history of BCF Topics and provide sample graphs. The following Section compares the approaches and provides measurements to evaluate their complexity. In Section 5 we discuss the results and the final Section concludes the work and gives an outlook for future work.

[13] BCF API: https://github.com/buildingSMART/BCF-API accessed 06.03.2022

3.2.2 Related Work

BCF Events

In 2016, the BCF group discussed adding an array of history items to the BCF specifications [14]. They decided only to add an Event System to the BCF API, and since version 2.1, the API has defined four endpoints to query data modifications concerning Topics and Comments. The Topics are the central concept of an Issue, containing attributes - such as a status, an author, and a due date. The Comments are a messaging concept within the format used to communicate with other stakeholders in a BCF Project.

The overall structure of these endpoints services is mainly the same. With queries containing globally unique identifiers of Topics or Comments, the user can fetch all Events related to those identifiers. The returned Event contains the author, the date and which parameters were changed. The BCF API returns all its responses in a JSON serialization.

```
{"events": [{
         "type": "status_updated",
         "value": "Active"
        }]}
```

Listing 3.6. JSON snippet of the timed action in the Events System.

The JSON body contains a list of timed actions, where every action represents an interaction with a Topic. Every action contains a list of Events that display which Topic attribute (the type) was changed to a new value (listing 3.6). The actions attributes are not sharing the types of the original Topics and Comments concepts but are represented in a combined string (e.g. *"status_updated"*). In total there are three different types of events: *created*, *updated*, and *deleted*.

The theory behind this data structure is that every time a Topic related data is created or edited, the server creates a log entry concerning the Topic change. Therefore, It is not enough to query only the Topic attributes to discover the complete picture of the history. All its Events have to be queried too.

[14] BCF history discussion: https://github.com/BuildingSMART/BCF-XML/issues/97 accessed 21.04.2022

Reification in RDF

Reifications are constructed by linking a node (the reification node) with the subject, predicate, and object of a statement in concern. The RDF reification mechanism, defined in the RDF Primer [Manola et al., 2004], is a known attempt to express statements about RDF assertions. Still, as Watkin and Nicole mentioned in [Watkins and Nicole, 2006], the reification is only indirectly related to the original statement. Instead of using reification, Watkin and Nicole suggested using Named Graphs introduced in [Carroll et al., 2005]. It was also presented how secure digests (signed check-sums of the triple contents) can be used to affirm the author and integrity of the graph. Cassidy and Ballantine extended the use of reifications to express RDF Patches in the TriG syntax [Bizer and Cyganiak, 2010]. An RDF Patch is a group of reifications to describe changes in the model.

RDF-star [Hartig, 2017] is an alternative, less wordy approach to describe reification in the Turtle syntax and SPARQL query language. It can be combined with standard reification and differs from the RDF pattern treating a triple as a subject. RDF-star introduces the concept of "Quoted Triples", which allows using an RDF Term (subject predicate object) as a subject or object of a triple.[15]

Describing States in Linked Building Data

Bonduel proposed a level system for the W3C Linked Building Data Community Group (W3C LBD CG)[16] consisting of three layers of complexity [Mathias, 2018], whereas the third layer can describe states of things. The Ontology for Property Management (OPM) is also founded on this architecture and is based on a three-layer modelling. With OPM, changes of properties can be described with the help of states [Holten Rasmussen et al., 2018].

The Ontology for Managing Geometry (OMG) [Wagner et al., 2019] builds on top of this level system so that either a thing can be:

- directly linked to the geometry description by a datatype or an object property (level 1).
- linked to one or many geometry nodes that are pointing to the description values (level 2), thereby adding the possibility to have many different geometric descriptions of the same thing.

[15] RDF-star and SPARQL-star https://w3c.github.io/rdf-star/cg-spec accessed 21.04.2022
[16] LBD Community Group: https://www.w3.org/community/lbd/ accessed 06.03.2022

- linked to a geometry node (as in level 2), but with an additional node in between. This allows the description of different states for the individual geometries (level 3).

Version Control and Versioning

Kiryakov and Ognyanov stated in [Ognyanov and Kiryakov, 2002] that a triple (or assertion) is the smallest changeable unit in an RDF graph. It implies that a change in a model can only be made by applying sets of triple removals and additions. However, in [Auer and Herre, 2006], *atomic graphs* - i.e. the surroundings of the blank notes - are the smallest unit to describe changes to avoid duplicating blank nodes. In this work, the context is how to express changes using bcfOWL, which is not relying on blank nodes. Therefore, no explicit blank nodes are needed.

In the light of the definition of Ball et al. [Ball et al., 1997], a version control system is a way to store and reconstruct past versions of data. Here we explore methods to express BCF data in RDF so that both aspects (store and reconstruct) are achievable. Other factors of version control, like the capability to undo and merge changes and synchronization, are out of this study's scope. On a graph level, versioning is also implemented by the different vendors. For example, GraphDB offers a "Data history and versioning" plugin [17] that can be queried for changes in the triples. However, this feature is not implemented in every graph storage and does not allow versioning on the RDF file level. Therefore, we decided to investigate how to express changes on the graph.

3.2.3 Approaches

The data models that we consider in this work are designed to reflect the design work responsibilities. Hence, each approach is crafted to remove no assertions from the model, but every modification adds new triples to the data model. We provide four different approaches for describing the data:

1. Event approach (Section 3.2.3)
2. States approach (Section 3.2.3)
3. Statements of Statements RDF-Star approach (Section 3.2.3)
4. Object Property Annotations RDF-Star approach (Section 3.2.3)

[17] GraphDB Documentation: https://graphdb.ontotext.com/documentation/9.8/standard/data-history-and-versioning.html accessed 05.03.2022

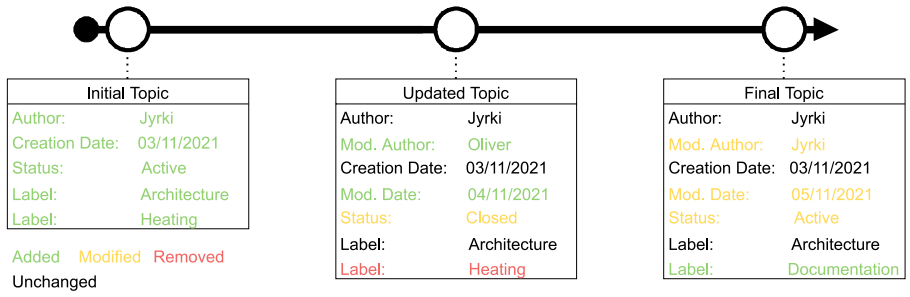


Fig. 3.5. Jyrki creates a new Topic, which is the most current version of it now. The Topic contains parameters such as the creation date, a title, two labels (Architecture and Heating) and a status set to "Active". On the next day, Oliver sets the Topic's status to "Closed" and removes the label "Heating". Another day later, the Topic gets reopened (set to "Active") by Jyrki and the label "Documentation" gets added.

Further, a common scenario was chosen to ensure comparability between the variants. The use-case is described in Fig. 3.5.

We have chosen the example because it displays (1) how a single attribute that is always "updated" in the context of BCF (e.g. the status) can be handled and (2) how lists (the labels) are to be treated that can contain at every stage different values. These circumstances are common in issue management scenarios, where an issue can be marked as closed, but other participants disagree and reopen the issue. Although we focus exclusively on the Topic in this paper, the same types of changes also appear in the context of Comments. The third central concept of BCF - the Viewpoint - is not considered because it should never be changed according to the current definition of the buildingSMART standard. The alternative approaches are presented in the following subsections. The complete graphs of the approaches and example queries can be retrieved from our GitHub repository[18].

Event approach

In the BCF API, the Topic Events Service is a log of changes for the Topic. It lists creations, updates, and removals of specific Topic event types. In the Events approach, we describe this logbook-like behaviour by creating a new log entry with all operations created as soon as an interaction with the Topic happens.

In the BCF APIs Event Service, the Events coexists next to the Topics and Comments. This behaviour mainly comes from the fact that the BCF standard does

[18]GitHub repository with sample queries and graphs: https://github.com/Design-Computation-RWTH/LDAC2022_Dataset

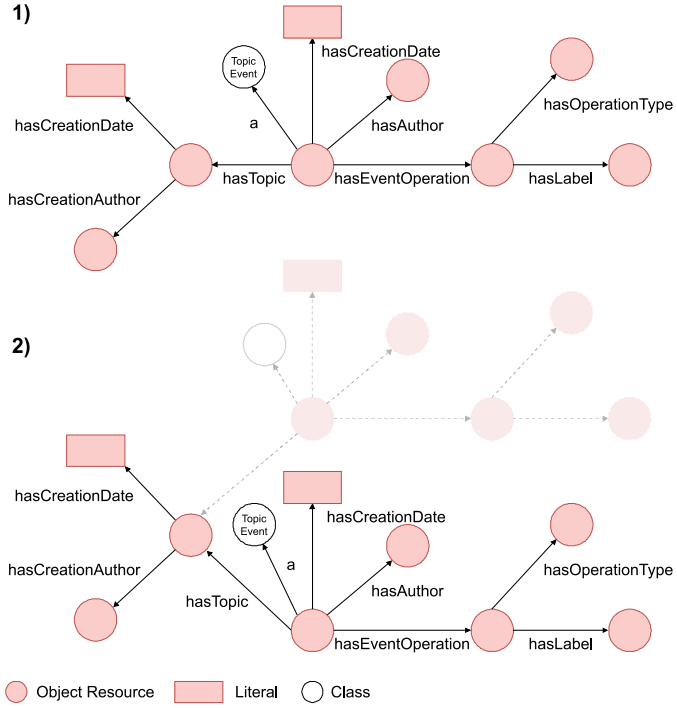


Fig. 3.6. The Event System approach is based on the BCF API. A Topic Event points to the related Topic and its operations that specifically define what type of interactions have happened. For example, if a property was removed, updated or created. Next to the type of operation, the attribute's value is stated.

not describe how the data is saved but is a way of communicating it. With bcfOWL, we have, on the one hand, a data format (RDF) and, on the other hand, the SPARQL query language for communicating the information from and to the graph. This is where the structure of the bcfOWL Events differs from the BCF API by not creating redundant data and allow using the Events to infer the Topics information.

An Event is described by the class of bcfOWL:TopicsEvent that states its author, the creation date and which Topic it belongs to. Furthermore, the object-type property bcfOWL:hasEvent-Operation links the Event to the different operations that occur in it (see Fig. 3.6). These operations state if it is an update, a creation or a deletion, and state the propertys current value. So if the status of a Topic was set from Active to Closed, this would be described as an Update operation, and the current value would be Active. The event approach comes with a change to the overall structure of bcfOWL. The Topics are now not responsible for providing all the attributes. Everything that can be changed in the context of BCF has now to be inferred from the Events. The newest value for an attribute is also its current value. One exception has to be made here because the property bcfOWL:hasLabel

usually provides a list. Therefore, it is not enough to select the newest Event, but the query has to consider if a label was removed at one point in time or not.

States approach

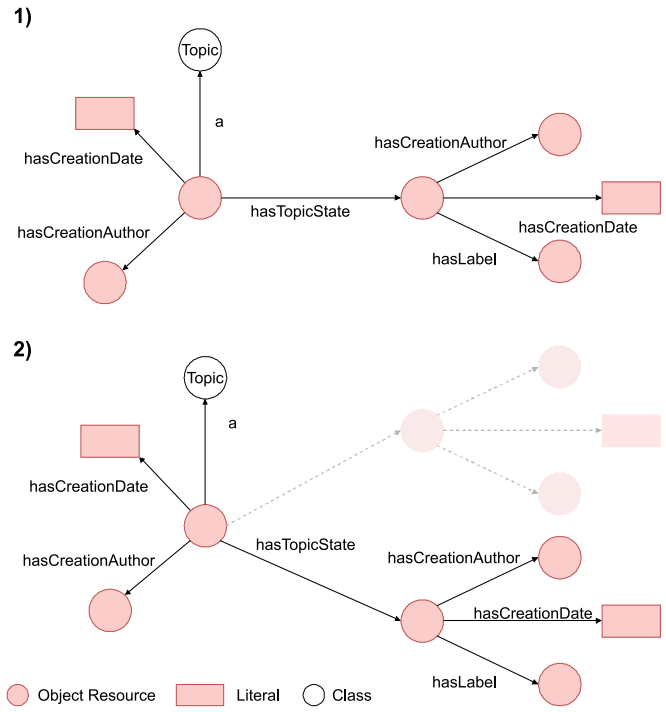


Fig. 3.7. The different states are referenced by the Topic with the property bcfOWL:hasTopicState. The states provide the complete picture of the Topic during a specific time. In order to find the current version of a Topic we have to query for the newest date. An example query is provided on the GitHub repository.

The States approach is splitting the attributes of the bcfOWL:Topic class into two parts and is transferring all the attributes that can change throughout the project to bcfOWL:TopicState. This approach is similar to OPM and OMG (Section 3.2.2), but due to the structure of the Topics, it requires only two levels. Every State provides the whole picture of its current status, meaning it provides all attributes, even if just one value of a new state has changed. A State comes with a creation author and a creation date. These are equivalent to the Modified Author and Modified Date of the BCF API. When retrieving the current values of a Topic, the State with the newest date has to be queried. (see Fig. 3.7)

RDF-star approaches

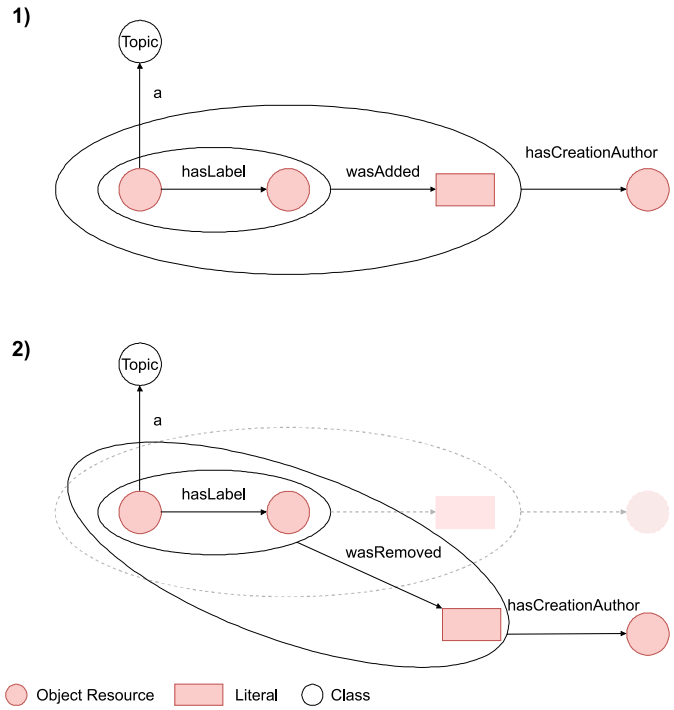


Fig. 3.8. Statements can be marked as added or removed with a time-stamp. The author of this change is added as a statement of the first statement. Even though the word itself states a deletion, resources are never removed from the graph to allow retrieval of the complete history of the Topics.

RDF-star can be used to make statements about triples and thereby attach meta information to them (Section 3.2.2). The following two approaches use RDF-star to express all interaction- and change-related content. To describe a change, we need to state when something was changed, who authored it, and semantics about whether it was an addition or removal. Since RDF-star allows many possible approaches, we decided to provide two different examples, 1) by using statements of statements and 2) by using object properties linking to annotation triples.

Statements of Statements RDF-star approach By using RDF-star with statements of statements (shown in Fig. 3.8) we can describe with the first statement when e.g. the triple for a "bcfOWL:hasTopicStatus" was added by using "bcfOWL:wasAdded" and a time string. This statement then gets annotated with its author. Since attributes can change many times - a status can change from "Active" to "Closed" and back again to "Active" - these annotations are constantly

added as soon as a triple is interacted with. The annotation with the newest date should always be considered the Topic's current state. Same as in the previous sections, the bcfOWL:hasLabel is an exception here, and all the labels should be treated as current as long as they were not annotated as removed in their current state.

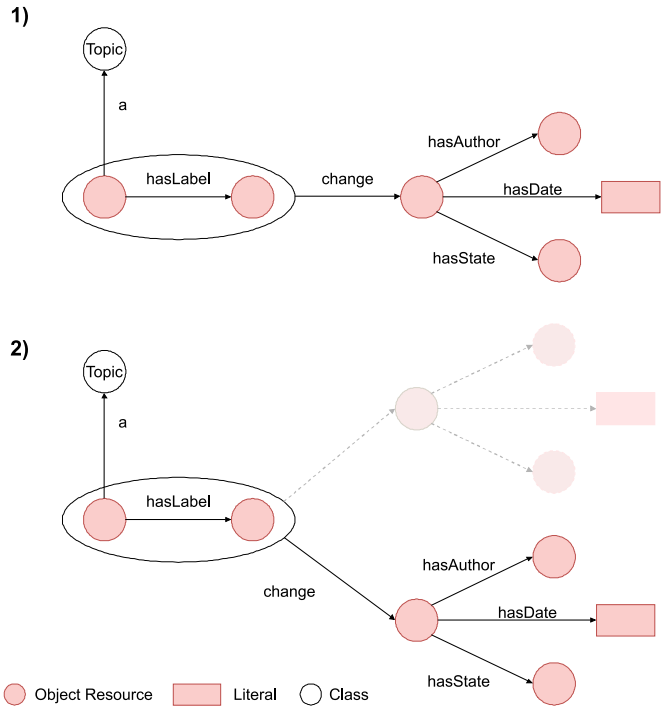


Fig. 3.9. Object property annotations only contain one level of statements and point to an object that contains all change-related information.

Object Property Annotations RDF-star approach The other RDF-star approach adds only one level of statements. There, every change-related property of the Topic is annotated by the bcfOWL:change (Fig. 3.9) object property, which points to a sub-graph that states the author, the date and a state (added or removed).

3.2.4 Results

The evaluation criteria were chosen to attempt to measure the expected simplicity of the various integration approaches. Therefore, we measured the triple counts, the maximum reasoning steps needed, diameters of the models, and the number

of variables needed to query the models using SPARQL. We expect that shorter RDF path lengths imply simplicity for writing queries. The same applies for more concise graphs. A lower number of triples in the graphs and the variables for the queries were considered "good".

All four handcrafted alternative data sets express the same information. The initial triple counts of the data and the added triples that describe the changes were counted using the Apache Jena API. They are shown in Table 3.3 and the counted triples are listed in separate files at the GitHub repository (Section 3).

The States approach had the lowest initial triple count and had the second-lowest number of added triples. The Statements of Statements approach performed best regarding the final triple count.

Although reification is not present in the triple counts of RDF-star, the reification triples are implicitly present in the SPARQL-star functions: SUBJECT, PREDICATE, OBJECT. Therefore, it is possible to make the approaches comparable by making the relations explicit in the graph. So, instead of many connected sub-graphs in the Apache Jena data presentation, as shown in Table 1, the alternative models have only a single connected sub-graph. It means they are traversable from any node in the model to any node as an undirected graph. Thus, it allows having an idea of the complexity of the graph. For example, the diameter of the model and the maximum steps from a Topic to a value can be evaluated and compared.

Three SPARQL queries were written for each alternative approach to estimating how easy the model is to query. The queries were to fetch 1) the current Topic data, 2) a state at a single point of time, and 3) the initial values for the Topic data. The needed numbers of SPARQL variables were counted (see Table 3.3). The Event system needed more variables to handle the two different object type cases: direct literal values and a list of enumerated type objects. The complete data is shown in the repository.

3.2.5 Discussion

Overall, the results were well in line with our experience when handcrafting the different approaches.

The Event approach (Section 3.2.3) is closely tailored to mirror the structure of the BCF API data model. So, the graph is never depicting the current state directly since it has to be inferred out of the different Event Operations, making the graph harder to query.

Table 3.3. Comparison of the bcfOWL version description approaches.

Approach	Events	States	RDF-Star: Statements of Statements	RDF-Star: Object Property Annotations
Initial triples in the data model	29	14	21	24
Triples added in the first update	12	8	7	13
Triples added in the final update	15	9	7	17
Triples at the end	57	31	35	54
Max. steps to a leaf.	3	2	3	3
Initial graph: model diameter	4	3	4	4
Final graph: model diameter	6	4	5	6
Final graph: The number of connected sub-graphs in Jena Model	1	1	23	8
SPARQL variables needed to query:				
- the current state	9	6	6	6
- state at a specific time	9	6	6	6
- state at initial time	6	3	4	5
blue - good, red comparatively bad				

The State approach (Section 3.2.3) is related to current developments in the LBD group and is following their examples. We would argue that it is easy to implement in bcfOWL and ontologies in general. Moreover, as its name says, it always expresses the complete data snapshot of a Topic, even though sometimes just a single value has changed. It is relatively space-saving compared to the other methods (Table 3.3). However, there is a risk of storing redundant parameters in the graph due to the complete state description. It lacks the granularity of the other approaches since it is impossible to see how a single parameter has changed from one state to the next without calculating the differences between the data sets.

However, with RDF-star, it is possible to describe states and track granular changes. Thus, in a way, it represents a middle ground between the other two methods. Both variants, which are based on RDF-star, are non-invasive to the original structures of the ABox statements. A legacy bcfOWL graph could thus easily be annotated, describing the triples' history. The RDF-star approach is not exclusive to BCF but could be a valuable addition in general. Because there is no deletion in our model, there will not be any dangling nodes in the system. While we cannot enforce non-deletion, each quoted triple in RDF-star is a separate assertion and thus, should not be considered dangled even if the referred node would be deleted.

Since timestamps are used in the Statements of Statements approach (Section 3.2.3), it has the limitation that the annotations could not be distinguished in the hypothetical case when an exactly simultaneous annotation of a triple would be made. This limitation does not exist with the Object Property Annotation approach (Section 3.2.3).

The RDF-star approach's drawback is that it is not yet supported by all the graph

database providers and Linked Data tools. If the scope of a project depending on RDF-star is limited to a graph database supporting this feature, no issues should be expected. Nevertheless, suppose the goal is to federate the data over multiple and distributed graphs where there is no control over the chosen database type. In that case, it can lead to compatibility issues. Furthermore, it is not yet clear where statements made about statements can end. Without a clear structure, these statements can be chained endlessly, adding complexity to the graphs.

3.2.6 Conclusion and Future Work

In this work, we have shown four different approaches to describing the history of BCF Topics changes. All four versions display the same information, only using different structures.

As the change information is an integral part of the issue management, our focus has been on how to make the version data available as a semantic web. So, instead of addressing the change management at a higher level, e.g. the named graph level or trying to use GIT, we have focused on the RDF graph level. We hypothesise that the approach allows better to trace editions and design choices made over time to the data.

We have pointed out that information should never be removed from the graph or changed, although enforcing this requirement may be challenging. Hence, in future work, we want to study the options for signing changes in the graph to assure the stakeholders that the values have not been altered. How the different approaches would perform in distributed and container-based environments should also be evaluated. Also, one engaging direction is to study how provenance data can be expressed and used in the BCF context. The provenance indicated in the graphs in this paper is currently only a placeholder, as the provenance of the data in bcfOWL has not yet been explicitly considered. Here the PROV ontology [Missier et al., 2013] is an interesting candidate. It can be studied how to use the ontology to express the ownership of the data, who played a role or contributed creating it, and how the content has been changed.

RDF-star performed reasonably well in our studies, so we think it should be further investigated for use in the LBD domain. Nevertheless, although RDF-star can be easily integrated into existing graph structures, we have not yet found examples of how this approach could be described in a Tbox statement for reasoning. An ontology extension for bcfOWL using RDF-star could help us tackle the complexity of statements made about statements.

The results of the study serve as a basis for further development for the bcfOWL ontology. For example, the shortest graph diameter and the shortest maximum path from a Topic to a value node can imply if the model is a good candidate as data representation for the BCF change data. In order to verify the results, we further have to simulate these approaches using larger data sets, covering a more considerable amount of changes and history data.

3.2.7 Acknowledgments

The EU funded this research through the H2020 project BIM4REN.

References

bcfOWL: A BIM collaboration ontology, 2021. URL https://ceur-ws.org/Vol-3081/12paper.pdf. International Workshop on Linked Data in Architecture and Construction, LDAC; Conference date: 11-10-2021 Through 15-10-2021, Online; accessed Nov. 08. 2024.

Sören Auer and Heinrich Herre. A versioning and evolution framework for rdf knowledge bases. In *International Andrei Ershov Memorial Conference on Perspectives of System Informatics*, pages 55–69. Springer, 2006.

Thomas Ball, Jung-Min Kim, Adam A Porter, and Harvey P Siy. If your version control system could talk. In *ICSE Workshop on Process Modelling and Empirical Studies of Software Engineering*, volume 11. Citeseer, 1997.

Chris Bizer and Richard Cyganiak. Proposed trig specification. *https://www.w3.org/2010/01/Turtle/Trig*, 2010. Online; accessed Oct. 15. 2023.

buildingSMART International, Ltd. Bim collaboration format (bcf) - an introduction, 2022. URL https://technical.buildingsmart.org/standards/bcf/. Online; accessed Oct. 15. 2023.

Jeremy J. Carroll, Christian Bizer, Pat Hayes, and Patrick Stickler. Named graphs, provenance and trust. In *Proceedings of the 14th international conference on World Wide Web*, pages 613–622, 2005.

Olaf Hartig. Foundations of rdf* and sparql*:(an alternative approach to statement-level metadata in rdf). In *AMW 2017 11th Alberto Mendelzon International Workshop on Foundations of Data Management and the Web, Montevideo, Uruguay, June 7-9, 2017.*, volume 1912. Juan Reutter, Divesh Srivastava, 2017.

Mads Holten Rasmussen, Maxime Lefrançois, Mathias Bonduel, Christian Anker Hviid, and Jan Karlshøj. Opm: An ontology for describing properties that evolve over time. In *CEUR Workshop Proceedings*, volume 2159, pages 24–33. CEUR Workshop Proceedings, 2018.

Frank Manola, Eric Miller, Brian McBride, et al. Rdf primer. *W3C recommendation*, 10(1-107):6, 2004.

Bonduel Mathias. Towards a PROPS ontology, 2018. URL https://github.com/w3c-lbd-cg/lbd/blob/a76232b682a979ff707ad0150c9336ff5b7b0f8e/presentations/props/presentation_LBDcall_20180312_final.pdf. Online; accessed Oct. 15. 2023.

Paolo Missier, Khalid Belhajjame, and James Cheney. The w3c prov family of specifications for modelling provenance metadata. In *Proceedings of the 16th International Conference on Extending Database Technology*, pages 773–776, 2013.

Damyan Ognyanov and Atanas Kiryakov. Tracking changes in rdf (s) repositories. In *Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web*, pages 373–378, 2002.

Anna Wagner, Mathias Bonduel, Pieter Pauwels, and Rüppel Uwe. Relating geometry descriptions to its derivatives on the web. In *Proceedings of the 2019 European Conference for Computing in Construction*, pages 304–313. European Council on Computing in Construction (EC3), 2019. ISBN 978-1-910963-37-1. doi: 10.35490/ec3.2019.146. URL http://hdl.handle.net/1854/LU-8633667. ISSN: 2684-1150, Online; accessed Oct. 15. 2023.

E. Rowland Watkins and Denis A. Nicole. Named graphs as a mechanism for reasoning about provenance. In *Asia-Pacific Web Conference*, pages 943–948. Springer, 2006.

3.3 Live Web Ontology for buildingSMART Data Dictionary (FBI 2021)

Image a scenario where an issue related to the ventilation system in a building and an air terminal needs to be replaced. The facility manager had originally handed IFC-based linked data. The system is intelligent and offers all the available information about the air terminal and its uses, e.g. buildingSMART Data Dictionary to explain the general AirTerminalShape property of the terminal.

Hypothesis 3.3 claimed that the proposed dynamic buildingSMART Data Dictionary (bSDD) ontology improves the consistency of the LBD data sets compared to the earlier methods. In the earlier approaches, separate exports had been translated to ontology definitions, or model property values were linked to an online but non-RDF entry description of the data. In this approach, the content of the buildingSMART Data Dictionary database is converted into an ontology description each time the ontology is accessed online. Thus, it is up to date, and since it is expressed using the W3C Web Ontology Language (OWL), the definitions are accessible for Semantic Web queries. Consequently, the descriptions of the LBD dataset property values are machine-readable. Since all information is expressed using the W3C Semantic Web standards, also used for other LBD data, the approach is more consistent than the earlier one.

In addition, compared with earlier exports, the new approach has a class concept of the named property sets, which better reflects how property sets are used in IFC and mapped to LBD.

The proposed method also has limitations. For example, at the example implementation, the conversion is triggered at any access, which is overkill, as the expected frequency of the changes in the database is not that high. In addition, if a couple of separate subsets of the classification content are exported into sets of ontologies, consistent naming of

the used namespaces must be considered.

Table 3.4 summarizes the use of the design science research method in this research.

My contribution to the paper: All the work is my contribution. The work was started in buildingSMART International bSDD Hackathon in March 2021 and then analysed and written into this article.

Table 3.4. The work of Oraskari [2021] in the Hevner et al. [2004] design science research framework

Guideline 1: Design as an artifact
I created an API interface that reads classification information for IFC version 4 from buildingSMART Data Dictionary (bSDD) API and creates an OWL ontology description for that on the fly. The work is a proof of construction for a dynamic bSDD ontology mirror of the bSDD data. Also, human-readable pages of the dynamic ontology description were created. As proof of construction, it demonstrates that the idea can be implemented. It can easily be extended to other classifications and data sources. Furthermore, although the implementation was made using Java Jersey and Apache Jena framework, the presented ideas are generic and can easily be implemented using other frameworks.

Guideline 2: Problem relevance

The business problem here was that, although there is a good amount of descriptive information available in the bSDD database, the available data for LBD applications has been either not current or available in a suitable format, which made it not straightforward to use it in applications (e.g., to offer users description of the data fields, e.g., here the property set entries), or in data validation.
The existing implementations were either not using a semantic web standard (RDF) to describe the properties or were not up to date and not online. Also, the earlier RDF exports focused on the properties missing the semantics of the property sets. It is debatable how much semantics is in that grouping.

The constituent user community of the artefact is the researchers of the Linked Building Data Community, but also developers in the AEC sector that want to use bSDD classification data in connection with other Linked Data in the project.
The solution belongs to the LBD ontologies problem class. *This work is an enabler for the following BIM use cases:* building operation and maintenance [Borrmann et al., 2018b], and handover the data to the facility manager.
When IFC/LBD data is handed handover to the client and then utilized in facility management systems for operation and management, linking to the ontology can enrich the data and improve usability.

Guideline 3: Design evaluation

Hevner et al. Design Research Evaluation Method: In a Controlled Experiment, ontology description of an IFC 4.0 IfcDoor was created and studied. Also the full IFC version 4 ontology description was written to test the scalability of the construction.
This was a feasibility study of an Linked Data approach that was compared with the earlier implementations (e.g., URL pointers to buildingSMART Data Dictionary and static file exports translated into RDF properties).

Guideline 4: Research contributions

There has been no such implementation made yet.
Generalization of the results: To my knowledge, no other dynamic ontologies have been built on a data dictionary API. The general idea was to provide the information in the forms, e.g., the W3C Web Ontology Language (OWL), that is useful for the community. In the general context, this is a step to make the data machine usable. While APIs are machine-readable and can offer OpenAPI documentation, they still miss the power of ontologies to describe things universally and to be machine understandable. It also is a step to make LBD graphs self-contained, i.e., you would not need external toolsets or services to understand the content.

Guideline 5: Research rigor

The work was peer-reviewed. Also, the source code was made available in my GiyHub repository[19].

[19] https://github.com/jyrkioraskari/bSDD_OpenAPI_OWL_demo

Guideline 6: Design as a search process

There are many ways to implement a dynamic ontology that uses the information from the buildingSMART Data Dictionary API. Java-based OpenAPI implementations that are easy to test online were already in use using Enunciate framework[20], so it was a natural selection. The backend communication uses an auto-generated template from the Swagger[21] description of the buildingSMART Data Dictionary API, which made coding efficient and easy to adapt. The idea to use the HTTP GET method to get the ontology imitates technics that a client software uses to fetch a file description of the ontology from the net, i.e., although the content is generated, for an web application, it is a standard ontology description file.

Guideline 7: Communication of research

Presented at 32rd Forum Bauinformatik 2021 and included in the 32rd Forum Bauinformatik 2021. Darmstadt, 32rd Forum Bauinformatik, Darmstadt, 09th - 10th September 2021 (DOI: 10.26083/tuprints-00019496) conference proceedings. The work was also presented at the buildingSMART International Virtual Summit March 2021.

[20] https://enunciate.webcohesion.com/ (Accessed August 8, 2023)
[21] https://swagger.io/ (Accessed August 8, 2023)

The work was made available as an open-access publication under the Creative Commons License Attribution 4.0 International (CC BY 4.0). The work was originally published in [Oraskari, 2021]. The content presented here is based on our final draft of the article.

Jyrki Oraskari Department of Design Computation, RWTH Aachen University, Aachen, Germany, Jyrki.Oraskari@dc.rwth-aachen.de

Abstract: The aspiration of Linked Building Data is in accelerating construction processes, improved collaboration, facilitating facility management, and an interlingua for the BIM domain. Linked Building Data is widely used in research projects and actively advances by the researchers in the World Wide Web Consortium Linked Building Data Community Group.

buildingSMART is a worldwide organization that develops standards to enable collaboration and digital BIM workflows. The buildingSMART Data Dictionary (bSDD) is an international standard that defines a hierarchically structured encyclopedia of object concepts and their properties used in building information models. It is provided as is an online service.

While there have been earlier work where data exports of bSDD have been expressed as the Web Ontology Language (OWL) and used in linked data, a consistent approach for keeping them automatically up to date has been missing.

In this work, I study how to publish the buildingSMART Data Dictionary classifications and terms as a web ontology, use them in the Linked Building Data context, and keep the data in sync with the buildingSMART Data Dictionary primary data. I present an online ontology generator that ad-hoc provides up-to-date ontology descriptions of concepts in the buildingSMART database.The earlier work is presented in this work, and the feasibility of the online live updates for the ontologies that reflect the bSDD database data is shown. The results are reflected and compared with the earlier work. This paper summarizes ways to publish the bSDD vocabularies as linked data in a sustainable way.

keywords: LBD, Linked Building Data, OWL, bSDD, Ontology

3.3.1 Introduction

The aspiration of Linked Data in the Architecture, Engineering, Construction, Owner Operator (AECOO) sector is in accelerating construction processes, improved collaboration, optimized facility management, and an interlingua for

the BIM domain. Linked Building Data (LBD) has a basis on the Semantic Web Stack of technologies [Gerber et al., 2008] that share the vision of automated information retrieval and a common framework for data. The technologies aim to facilitate the common understanding of the concepts, make data accessible on the web, and enable rule-based reasoning.

buildingSMART is a worldwide organization that develops standards to enable collaboration and digital BIM workflows. Industry Foundation Classes (IFC) is one of the standards. It is an open, vendor-neutral and international standard (ISO 16739-1:2018)[The International Organization for Standardization, 2018] to facilitate interoperability in the AEC domain.

The common extension mechanism of IFC to express properties that are not predefined in the standard is to use an IfcPropertySet container to hold the extra property values. They can be of type IfcSimpleProperty used for single property values and IfcComplexProperty that can compose more complex data structures.

Although they can be user-defined, i.e., any, there are also predefined property sets by buildingSMART, e.g., Pset_WindowCommon and Pset_RailingCommon. [Zhiliang et al., 2011]. The predefined sets of properties are preserved and made available in the buildingSMART Data Dictionary (bSDD) [Beetz, 2014] that is an international and hierarchically structured encyclopedia of object concepts and their properties used in building information models. It is provided as is an online service. In this work, I concentrate on how these can be presented as Linked Building Data.

On the other hand, Ontology for Property Management (OPM) [Bonduel et al., 2018, Holten Rasmussen et al., 2018], an extension of the Smart Energy-Aware Systems (SEAS) ontology and Provenance Ontology (PROV-O), is an approach to express evolving values of a property attribute in the Linked Building Data context. It offers three levels of complexity. In Level 1, OWL Datatype properties and OWL Object properties are used to link the element individuals with the properties. It offers the best accessibility for the data. Level 2 follows the approach defined in the W3C and OGC Semantic Sensor Networks (SSN) ontology and expresses the property of the instance using an intermediate instance of class seas:Property. Level 3 defines an additional level that can be used to enunciate the temporal validity of the expressed value and so supports versioning the data [Rasmussen et al., 2018].

In [Bonduel, 2018], Bonduel has compared how ifcOW, ifcWOD, simpleBIM, BIMSO-BIMBO, but in particular, how property sets can be addressed in the Building Topology Ontology (BOT) and the Ontology for Property Management context.

When expressing bSDD classification data in Linked Building Data, I aim to connect the new work with this earlier research around Linked Building Data. This paper shows a feasibility study to test how a live update can be implemented for bSDD classification extracts expressed as Linked Data ontologies.

One of the core Linked Data principles is that each data item has a unique web identifier that can be used to get more information about the concept [Bizer et al., 2008, Consortium et al., 2014]. Descriptions of the specification should be available in one of the languages of the Semantic Web stack. Partial solutions exist for the buildingSMART Data Dictionary (bSDD), but there has not been a solution that reflects the fundamental idea of the dynamic data content of the database of the buildingSMART Data Dictionary and the fact that the database is the master data, i.e., the ontology descriptions should without delay, reflect the changes.

In this work, I study how to publish the buildingSMART Data Dictionary classifications and terms as a web ontology, use them in the Linked Building Data context, and keep the data in sync with the buildingSMART Data Dictionary primary data. The feasibility of the solution is tested. I present my online ontology generator that ad-hoc provides up-to-date ontology descriptions of concepts in the buildingSMART database. The new approach is compared with the earlier work.

This paper summarizes ways to publish the bSDD vocabularies as linked data in a sustainable way.

In the following chapters, I first review the related work. In Chapters 3, my implementation is presented. Finally, the findings are discussed and conclusions summarised.

3.3.2 Related Work

There has been earlier work to convert buildingSMART Data Dictionary property definitions to various formats. They are exports to YAML, Extensible Markup Language(XML), JavaScript Object Notation (JSON), and Microsoft Net Resource files (RESX). They were limited to represent the data as structural information. For example, XML Schema Definition (XSD) language was not used.

There has been unpublished work to define bSDD as an Linked Data linked data file based ontology. In [Zhang et al., 2014], Zhang et al. also mention that the bSDD have been transformed into OWL resources. Peter Willems' IFC4-PSD [Willems, 2021] is a rather direct translation of the above mentioned data exports where data

exports of bSDD have been expressed as the Web Ontology Language (OWL) files that instead of classes expresses the bSDD data as OWL individuals.

Oraskari et al. [et al., 2020a] formatted and made IFC4 PSD ontology that was earlier only available as a set of OWL files online referenceable and human-readable. Also, this was a static extract of the bSDD database content and was missing the named property set concepts.

Furthermore, [et al., 2020b] present OWL ontology description for building elements and Mechanical, Electrical and Plumbing (MEP) as OWL files in the repository. The files contain rdfs:seeAls links to point to Non-RDF bSDD references.

The challenge has been to keep the data up to date. Compared to the solution that I offer in this paper, they have also not offered dereferenceable URLs. Also, in [Pauwels et al., 2016], Pauwels et al. have shown an example conversion result for the Calcium silicate board concept in the bSDD as Resource Description Framework (RDF). Particularly, they mentioned that no standard method exists to generate Resource Description Framework (RDF) [Lassila et al., 1998, Brickley and Guha, 2014] graphs from the bSDD API. That is, there are no standard tools to translate the data to the Semantic Web Stack language.

On the other hand, the Building Product Ontology [Wagner and Rüppel, 2019], buildingSMART Data Dictionary (bSDD) taxonomy entries has been addressed as non-RDF data. The linking from a semantic model was made implicitly using properties that contains a globally unique identifier of a buildingSMART Data Dictionary (bSDD) entry found manually in buildingSMART Data Dictionary Concept search results. Therefore, the buildingSMART Data Dictionary (bSDD) information that describes the standard was not available using Semantic Web stack languages. First, if an URI is represented as a literal, the software needs to parse it before use and, second, if the API is not returning RDF in any of the RDF serialization types, the semantic web tool chain is not in used and for example reasoning or semantic queries are not available.

3.3.3 bSDD OWL OpenAPI implementation

Figure 3.10 shows the basic idea of the solution. In short, my service provides a web service to get the Web Ontology Language (OWL)[World Wide Web Consortium and others, 2012] description for the associated property sets and contained properties of an IFC element. In the background, the service accesses the buildingSMART Data Dictionary API[buildingSMART International, 2021] to get

classifications data for the IFC element and formats the results by mapping the piece of data into a small ontology.

The bSDD API result for an Classification query for an IFC element class is mapped so that the result of IFC class related property set is linked in a OWL Object Property definition, where Domain links to the IFC element class and Range the creates property set. Then for each property of the property set, is mapped using OWL Datatype Property. Supported data types are XSD.xdouble, XSD.xboolean, and XSD.xstring.

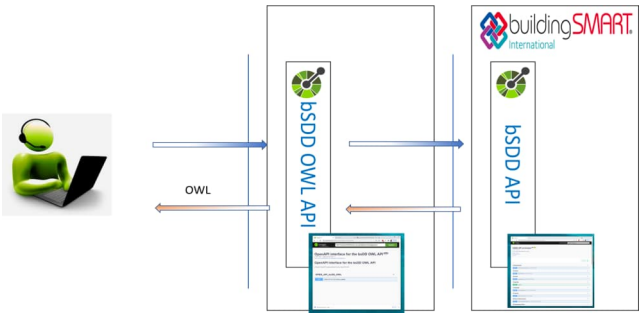


Fig. 3.10. bSDD OWL OpenAPI implementation. When the user (on the left) uses client software to send a query to the ontology generator API (at the middle), the query is translated and conveyed to the buildingSMART API back-end. The result is then mapped into the OWL ontology description language and send to the client.

The benefits of implementing a standard OpenAPI implementation is that it has the following two parts. First, it provides a standard Restful interface. For this work, it was beneficial that the created Application Programming Interface (API) could be accessed using standard HTTP 1.1 [Fielding and Reschke, 2014] access methods. As HTTP GET is the method that software uses to get more information about an ontology reference using the associated URI, it was possible to implement a dynamic API to create the ontology description file. The solution was transparent for the software. That is, it made it possible to offer the generated ontology description so that the URIs of the defined ontology classes and data or object properties can be dereferenced. Therefore, if the ontology is used in a Linked Building Data graph, information about the meaning of the properties can be found, and for the standard property sets and the associated properties, the description is the one that is provided by buildingSMART. The second part is the user interface that documents the API usage and lets the user test it using the forms on the site. No extra tools are needed.

Figure 3.11 shows the user interface and OWL output the HTTP GET interface when the address is accessed by software.

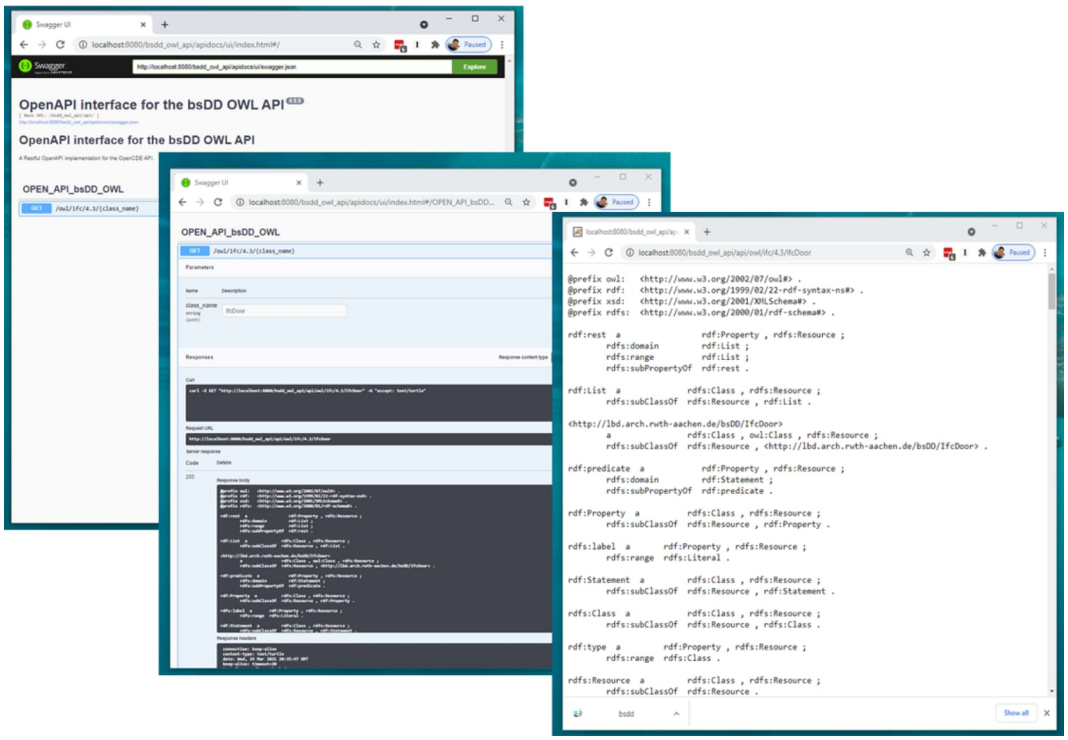


Fig. 3.11. The OpenAPI SWAGGER user interface of the solution. The first screen on the left shows the introductory page. In the middle is shown the result for the query on the user interface. On the right, the API output in OWL in Turtle - Terse RDF Triple Language is shown when the API is accessed directly.

The software architecture for the implementation is shown in Figure 3.12. It was written in Java programming language. The standard OpenAPI specification [Software, 2021] compatible interface was written using the combination of Jersey and Enunciate. The machine-accessible RESTful interface was written using the standard Jersey reference implementation for Jakarta RESTful Web Services (JAX-RSl), formerly known as Java API for RESTful Web Services. Then the Enunciate [Cohesion, 2021] plugin in Eclipse integrated development environment (IDE) was used to generate the human-readable Swagger HTML pages. The complete source code of the implementation is available in my bSDD_OpenAPI_OWL_demo GitHub repository [Oraskari, 2021].

Although the OWL description could have been output straightforward, to ensure the syntactical correctness of the output, the output, the ontology description was generated using Apache Jena Ontology API. The output for the IfcDoor export has been validated visually using Web-based Visualization of Ontologies (WebVOWL), and using the University of Manchester OWL Validator tool [Horridge, 209]. The tests were passed.

The result of this research is an online API interface software. The API can have a permanent and public address in future work, which would make the Ontology pitfall scanner! (Oops!) [Poveda-Villalón et al., 2014] applicable.

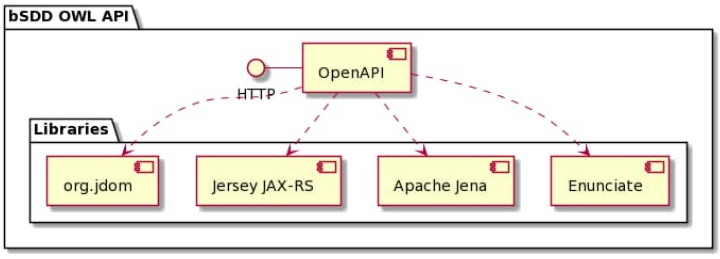


Fig. 3.12. The software architecture for the implementation.

Furthermore, Widoco [Garijo, 2017] toolkit was used to create the human readable ontology description (Figure 3.13). Instead of using the normal interactive mode of the toolkit, a template file was created, and with the template, the update of the pages was made automated. However, while the RESTful API and the OWL description generation are triggered when accessing the address, the Widoco updates can be updated when the OWL description is generated. In the case only or mainly the ontology would be accessed, a times process could be used. While the bSDD database is not expected to change in real time, scheduling can be a decent solution.

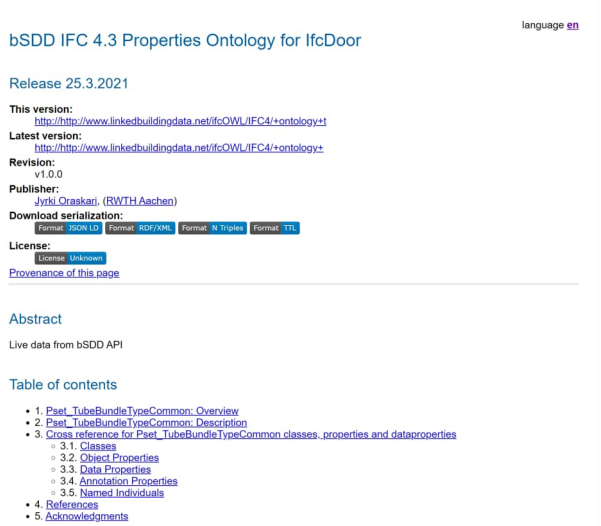


Fig. 3.13. The Widoco generated and human-readable description of the ontology results from an automated process that uses a generic document template to compose a human-friendly version of the on-the-fly generated OWL ontology description from the API. The automated process allows the data to be up to date and align with the machine-readable OWL.

In the first tests, the whole IFC4 classification was converted into an OWL description. It turned to take time (more than 60 seconds) and consisted of many subqueries. To limit the server traffic and improve performance, an ontology description was generated based only on the associated IFC class. The solution has pros and cons. It was implemented using one HTTP access. An example OWL generation time was 2112ms. The cons are that it creates a strong dependence between the IFC class type and the property set. In principle, the solution could also be based on a local cache that is time by time updated. Then the whole classification could be in a single OWL description.

For example, the ontology description for an IfcDoor in the IFC4 classification in bSDD was generated. The result is shown in Figure3.15. The visual evaluation shows that the property makes sense. So, the solution was feasible. However, in the solution, the IfcDoor class was given a URI with a local base URI. To associate it with the elements of a Linked Building data model can be done using the context of the property, for example, when converting a IFC data model using IFCtoLBD [Oraskari et al., 2021] to RDF.

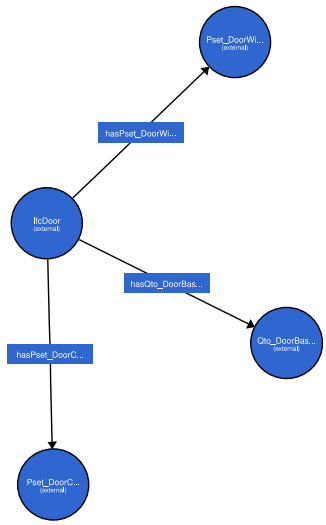


Fig. 3.14. The generated ontology description for IfcDoor.

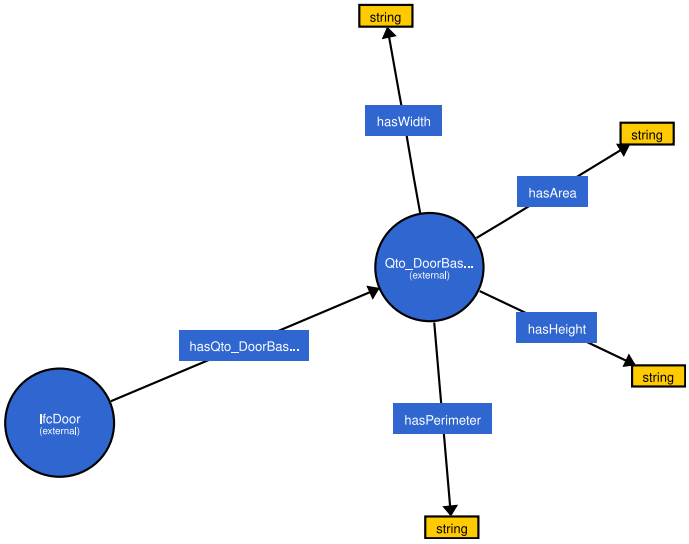


Fig. 3.15. The generated ontology description for a property set associated with IfcDoor.

3.3.4 Conclusions

This work showed that it is feasible to create a dynamic web API solution that dynamically creates ontologies from buildingSMART Data Dictionary database. Because the ontology description is created in seconds for each access, it is always reflecting the master data that is in the buildingSMART Data Dictionary database.

In the work, I concentrated on the IFC4 classification, but the buildingSMART Data Dictionary database API specification was also studied. The conclusion is that the work can be generalized to all the classifications in the database, but while the data content may vary, further tests are needed.

When compared with the earlier solutions, this has a clear advantage: it is update. So the data is not only align with the master data in buildingSMART, but when new data sets are created, no extra effort is needed to export it.

Although the presented solution shows the ontology generator as a separate web service, which was easy to implement, in the end, in future work, this could be fully integrated into the buildingSMART Data Dictionary API, possibly accessing the internal database directly.

In the future work, I will study how this can be integrated into the Ontology for Property Management (OPM) and Building Topology Ontology (BOT) context.

3.3.5 Acknowledgments

This research is part of the BIM4Ren project which has received funding from the European Unions Horizon H2020 research and innovation programme under grant agreement No 820773.

References

Jakob Beetz. A scalable network of concept libraries using distributed graph databases. In *Computing in Civil and Building Engineering (2014)*, pages 569–576. ASCE Press, 2014.

Christian Bizer, Tom Heath, Kingsley Idehen, and Tim Berners-Lee. Linked data on the web (ldow2008). In *Proceedings of the 17th international conference on World Wide Web*, pages 1265–1266. Association for Computing Machinery New York, NY, United States, 2008.

M. Bonduel. Towards a props ontology (2018), 2018. URL https://github.com/w3c-lbd-cg/lbd/blob/gh-pages/presentations/props/presentation_LBDcall_20180312_final.pdf. [Online; accessed 24-July-2021].

Mathias Bonduel, Jyrki Oraskari, Pieter Pauwels, Maarten Vergauwen, and Ralf Klein. The ifc to linked building data converter-current status. In *6th Linked Data in Architecture and Construction Workshop*, volume 2159, pages 34–43, 2018.

Dan Brickley and Ramanathan Guha. Rdf schema 1.1 w3c recommendation 25 february 2014. https://www.w3.org/TR/rdf-schema/, 2014. [Online; accessed 24-July-2021].

buildingSMART International. buildingsmart data dictionary api. http://bsdd.buildingsmart.org/docs/, 2021. [Online; accessed 25-July-2021].

Web Cohesion. Enunciate, articulate your web service api. http://enunciate.webcohesion.com/, 2021. [Online; accessed 25-July-2021].

World Wide Web Consortium et al. Best practices for publishing linked data. https://www.w3.org/TR/ld-bp/, 2014. [Online; accessed 24-July-2021], World Wide Web Consortium.

Jyrki Oraskari et al. Psd. https://www.linkedbuildingdata.net/IFC4-PSD/PsetDef/index-en.html, 10 2020a. [Online; accessed 24-July-2021].

Pieter Pauwels et al. Product ontologies for the construction industry. https://github.com/pipauwel/product, 2020b. [Online; accessed 24-July-2021].

Roy Fielding and Julian Reschke. Hypertext transfer protocol (http/1.1): Semantics and content, rfc 7231, june. https://datatracker.ietf.org/doc/html/rfc7231, 2014. [Online; accessed 24-July-2021], CERN; IETF, W3C.

Daniel Garijo. Widoco: a wizard for documenting ontologies. In *International Semantic Web Conference*, pages 94–102. Springer, Cham, 2017. doi: 10.1007/978-3-319-68204-4_9. URL http://dgarijo.com/papers/widoco-iswc2017.pdf. [Online; accessed 24-July-2021].

Aurona Gerber, Alta Van der Merwe, and Andries Barnard. A functional semantic web architecture. In *European Semantic Web Conference*, pages 273–287. Springer, 2008.

Mads Holten Rasmussen, Maxime Lefrançois, Mathias Bonduel, Christian Anker Hviid, and Jan Karlshøj. Opm: An ontology for describing properties that evolve over time. In *CEUR Workshop Proceedings*, volume 2159, pages 24–33. CEUR Workshop Proceedings, 2018.

Matthew Horridge. Owl 2 web ontology language document overview. http://visualdataweb.de/validator/, 209. [Online; accessed 27-July-2021].

Ora Lassila, Ralph R Swick, et al. Resource description framework (rdf) model and syntax specification. *W3C recommendation*, 1998.

Jyrki Oraskari. jyrkioraskari/bSDD_OpenAPI_OWL_demo: bSDD_OpenAPI_OWL_demo, July 2021. URL https://doi.org/10.5281/zenodo.5136543. [Online; accessed 25-July-2021].

Jyrki Oraskari, Mathias Bonduel, Kris McGlinn, Pieter Pauwels, Freddy Priyatna, Anna Wagner, Ville Kukkonen, Simon Steyskaland, and Joel Lehtonen. jyrkioraskari/ifctolbd: Ifctolbd 2.30.1, May 2021. URL https://doi.org/10.5281/zenodo.4885102. [Online; accessed 24-July-2021].

Pieter Pauwels, J. Plateaustraat, Thomas Krijnen, and J.akob Beetz. Making sense of building data and building product data. *Information Systems in the Built EnvironmentEAISI High Tech SystemsEAISI Foundational*, 2016. URL https://termcoord.eu/wp-content/uploads/2016/03/11.-Pauwels-et-al.-Making-sense-of-building-data-and-building-product-data.pdf. [Online; accessed 24-July-2021], Terminology Coordination Unit.

María Poveda-Villalón, Asunción Gómez-Pérez, and Mari Carmen Suárez-Figueroa. OOPS! (OntOlogy Pitfall Scanner!): An On-line Tool for Ontology Evaluation. *International Journal on Semantic Web and Information Systems (IJSWIS)*, 10(2):7–34, 2014. IGI Global.

Mads Rasmussen, Maxime Lefrançois, Mathias Bonduel, Christian Hviid, and Jan Karlshøj. Opm: An ontology for describing properties that evolve over time. In *6th Linked Data in Architecture and Construction Workshop*, pages 24–33, 2018.

SmartBear Software. Openapi specification version 3.0.3. https://swagger.io/specification/, 2021. [Online; accessed 25-July-2021].

The International Organization for Standardization. Iso 16739-1: 2018: Industry foundation classes (ifc) for data sharing in the construction and facility management industries–part 1: data schema, 2018.

Anna Wagner and Uwe Rüppel. Bpo: the building product ontology for assembled products. In *Proceedings of the 7th Linked Data in Architecture and Construction workshop (LDAC 2019), Lisbon, Portugal*, page 12, 2019. URL https://ceur-ws.org/Vol-2389/08paper.pdf. Online; accessed Oct. 15. 2023.

Peter Willems. Ifc4-psd. http://linkedbuildingdata.net/ifcOWL/IFC4-PSD/Pset_ActionRequest/index-en.html, 2021. [Online; accessed 25-July-2021].

World Wide Web Consortium and others. Owl 2 web ontology language document overview. https://www.w3.org/TR/owl2-overview/, 2012. [Online; accessed 24-July-2021],Word Wide Web Consortium.

Chi Zhang, Jakob Beetz, and Bauke de Vries. An ontological approach for semantic validation of ifc models. In *Proceedings of the 21st International Workshop on Intelligent Computing in Engineering (EG-ICE)*, 2014.

Ma Zhiliang, Wei Zhenhua, Song Wu, and Lou Zhe. Application and extension of the ifc standard in construction cost estimating for tendering in china. *Automation in Construction*, 20(2):196–204, 2011. Elsevier.

3.4 Related publications

I have also published the following articles around the theme. They extend the work presented above. My contribution has been as an expert of the Linked Building Data.

Lessons Learned from Designing and Using bcfOWL [Schulz et al., 2023]

Oliver Schulz Department of Design Computation, RWTH Aachen University, Aachen, Germany, schulz@dc.rwth-aachen.de
Jyrki Oraskari Chair of Individualized Production (IP), RWTH Aachen University Aachen, Germany, oraskari@ip.rwth-aachen.de
Jakob Beetz Department of Design Computation, RWTH Aachen University, Aachen, Germany, j.beetz@caad.arch.rwth-aachen.de

Abstract: The bcfOWL ontology has been developed as part of the EU Horizon 2020 BIM4Ren project to enable communication between BIM Collaboration Format (BCF) Issues and Linked Building Data (LBD) concepts described in the Semantic Web. This paper evaluates the current approach in bcfOWL based on its use in the BIM4Ren project. The ontology serves as an interlanguage for component-based communication, providing a gateway to the systems of different domains. During its use in the project, new insights into the usability of the ontology were gained. We discuss these findings and provide suggestions to complement the original design principles of bcfOWL, targeting the LBD domain. Our work should guide future research in component-based communication in building-related projects and provide helpful considerations for future ontology designs. We also discuss potential areas of improvement for bcfOWL, including versioning, Ontology Design Patterns and validation. Overall, bcfOWL aims to improve querying capabilities and connectivity with Linked Building Data, making it a valuable tool for building-related projects

dstv: An ontology-based extension of the DSTV-NC standard for the use of linked data in the automation of steel construction [Kirner et al., 2023]

Lukas Kirner Chair of Individualized Production (IP), RWTH Aachen University Aachen, Germany, kirner@ip.rwth-aachen.de
Jyrki Oraskari Chair of Individualized Production (IP), RWTH Aachen University Aachen, Germany, oraskari@ip.rwth-aachen.de
Victoria Jung Chair of Individualized Production (IP), RWTH Aachen University Aachen, Germany, jung@ip.rwth-aachen.de
Sigrid Brell-Cokcan Chair of Individualized Production (IP), RWTH Aachen University Aachen, Germany, brell-cokcan@ip.rwth-aachen.de

Abstract: To meet the demands of automated steel construction, there is a need for innovative ways to link process data, measured deviations, and tolerances. Our current research in robotic steel fabrication aims to tackle this challenge by creating an adaptable information model interface that can seamlessly incorporate cross-process considerations required for precise and efficient fabrication beyond current Building Information Modeling (BIM). The goal is to improve existing information interfaces and increase the utilization of flexible and partially automated robot concepts in steel construction. Our approach uses existing standards and product interfaces, such as DSTV-NC in steel construction, which we convert and enhance through an ontology that includes tolerances and process parameters. The outcomes of our study contribute to the development of automated systems in construction and support small and medium-sized enterprises in steel construction by addressing challenges related to skills shortages, productivity, and occupational safety.

The semantic link between domain-based BIM models [Teclaw et al., 2023b]

Wojciech Teclaw SINTEF, Trondheim, Norway,

wojciech.teclaw@sintef.no
Mads Holten Rasmussen NIRAS International Consulting, Allerød, Denmark, mhra@niras.dk
Nathalie Labonnote SINTEF, Trondheim, Norway, nathalie.labonnote@sintef.no
Jyrki Oraskari Chair of Individualized Production (IP), RWTH Aachen University Aachen, Germany, oraskari@ip.rwth-aachen.de
Eilif Hjelseth Norwegian University of Science and Technology (NTNU), eilif.hjelseth@ntnu.no

Abstract: Over the past few years, the construction industry has undergone technological advancements to improve efficiency and productivity. One of the latest innovations is using semantic web technologies to address interoperability issues and achieve machine interpretability of data. Despite several implementations of Industry Foundation Classes (IFC) to graph model converters, there has been no analysis of the semantic linkages between duplicated elements. This study aims to fill this gap by providing a semantic framework for linking elements in the graph representations of IFC models. This is achieved by reviewing commonly used ontologies, IFC to semantic technology converters, and the use of the owl:sameAs predicate. The study presents a methodology for generating additional links between duplicated elements in IFC model graph representations using selected geometrical features to address interoperability issues. The methodology is tested on domain-based IFC models and provides an efficient way to link models into a federated source of information about interdisciplinary Building Information Modelling (BIM) models. The study's findings are expected to enhance the interoperability and semantic capabilities of BIM models, promoting collaboration and improving the efficiency of the construction industry.

Building system data integration using semantic representation [Teclaw et al., 2023a]

Wojciech Teclaw SINTEF, Trondheim, Norway, wojciech.teclaw@sintef.no

Jyrki Oraskari Chair of Individualized Production (IP), RWTH Aachen University Aachen, Germany, oraskari@ip.rwth-aachen.de
Mads Holten Rasmussen NIRAS International Consulting, Allerød, Denmark, mhra@niras.dk
Nathalie Labonnote SINTEF, Trondheim, Norway, nathalie.labonnote@sintef.no
Eilif Hjelseth Norwegian University of Science and Technology (NTNU), eilif.hjelseth@ntnu.no

Abstract: In recent decades the construction industry started a rapid digitalisation process resulting in the widely used BIM methodology. Despite developing multiple Industry Foundation Classes (IFC) schemas, the interoperability and data integration between models remains an issue. The study analyses the missing relations and limitations of the IFC schema and provides the answer by providing a methodological framework. The framework aims to fill this gap by extending a semantic representation of IFC models by missing component relations. This methodology is tested based on a case study containing multiple BIM models, presenting the improved interoperability between BIM models using SPARQL queries. The study's findings are expected to enhance the connectivity between MEP components placed in different models and provide knowledge representation for developing the Digital Twin concept.

3.5 Discussion and Conclusions

In this chapter, I continue the publication theme of data management discussed in Chapter 2, now dealing with terminological axioms (Tbox) to define ontologies. As I pointed out in Section 1.3, ontologies play an essential role in Linked Data.

The section 3.1 shows how to represent building related issue data in the context of LBD. Another important topic is the best practices for creating a new LBD ontology and the design process described in the article to achieve

this result. This work provides an example how new LBD ontologies could be created around existing open standards. More has been done than could fit into the article. For example, the Apache Jena Ontology Library, LODE [22] and OntOlogy Pitfall Scanner![23] were used. The bcfOWL ontology[24] was also published in the prefix.cc[25] namespace lookup service. There are also still minor things to be done (e.g. to test that HTTPS is being used): the work could be published in LOV[26] and a permanent redirect from https://w3id.org/ could be added. Gaps in explicitly using the ontology design patterns [Gangemi and Presutti, 2009] have been handled in our works in Schulz et al. [2023] and Kirner et al. [2023].

Version control has been discussed in two separate articles, presented in Section 2.1 and 7.1. Section 3.2 continues the theme. It focuses on exploring how BCF versioning data should be expressed. As an alternative, we looked more closely at the applicability of the Ontology for Property Management (OPM) [Holten Rasmussen et al., 2018], but now with the additional condition that data may only be added to the graph. With the additional condition, we wanted to emulate audit logs. The idea was to enable further development to allow data to be signed, creating a chain of transactions whose contents would be difficult to dispute later if someone only had a copy of the data. The four options presented provide an insight into the directions in which the bcfOWL ontology could be developed. The work also provides insight into how the RDF-star could be applied to this problem domain.

On the other hand, the section 3.3 addresses two aspects: how to create dynamic ontologies and how to provide information from the buildingSMART Data Dictionary (bSDD). bSDD is essential in the LBD context, and in a more integrable form, RDF, it is more accessible to the LBD developer community.

[22] https://essepuntato.it/lode/
[23] https://oops.linkeddata.es/
[24] http://lbd.arch.rwth-aachen.de/bcfOWL
[25] https://prefix.cc/bcfowl
[26] https://lov.linkeddata.es/dataset/lov/

In this chapter I have addressed three ontology related topics: creation, version control and dynamic ontologies in the LBD context. The hypotheses and research questions chosen reflect these. They are not intended to be exhaustive, but rather to provide guidelines on how the topics might be addressed. The artefacts created can serve as a basis for new applications and research.

Chapter 4

Publish Data as Common Data Environment

In this chapter, I address the research questions: How can linked data be used to create the highest envisioned maturity level of BIM collaboration and how can a federated Common Data Environment be organized?

I have chosen the following hypotheses that express the chosen focal points that I have researched in more detail.

Hypothesis 4.1 *Standard Information Container for Document Delivery (ICDD) container-based building information can be published online using the building industry buildingSMART OpenCDE and W3C Linked Data Platform (LDP) standards, increasing the accessibility of the information compared to a file-based standard. This is handled in Sections 4.2 and 4.3.*

Hypothesis 4.2 *Using Linked Building Data a ISO 19650 BIM Stage 3 mature Common Data Environment (CDE) can be created. The claim is handled in Sections 4.4 and 4.5*

Image a scenario where a facility manager keeps all issue data in a BCF server, and then gets a set of as-built model

data of the associated building packaged in a standard Information Container for Document Delivery (ICDD). BCF API and Document API are part of buildingSMART OpenCDE API and so implementation of a CDE. So, when the facility manager uploads the building data using the Document API, it becomes accessible and can be interlinked with the BCF data.

In this Chapter, I will introduce what support I have for the claims and discuss arguments that are against them. In Section 4.6, I will summarize findings presented in the chapter.

In Section 4.1.1 I will introduce the concept of Common Data Environmen (CDE) and in Section 4.1.2 the ISO 19650 Stage 3 maturity level. After that in Sections 4.2 and 4.3 our work how container models like Information Container for linked Document delivery (ICDD) [ISO, 2018a,b] can be published online as a CDE is presented. In Sections 4.4 and 4.5 our work around distributed CDE is introduced.

Figure 4.1 summarises how the research questions, hypotheses and their handling are connected. Then, Figure 4.2 shows my articles of the chapter at a glance.

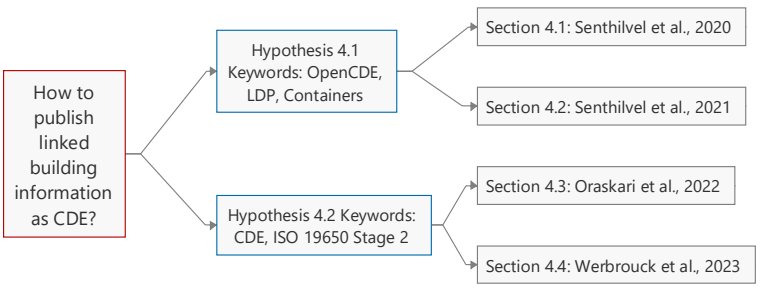


Fig. 4.1. How the research questions, hypotheses, and their handling in articles and the sub-sections are connected.

Publication	Objectives	Method	Key findings
Senthilvel et al., Common Data Environments for the Information Container for linked Document Delivery., 2020.	• How CDEs benefit from ICDD containers, which special focus on structuring and linking heterogeneous data on meta and sub-document levels?	The Design Science Research Method: Analytical Static Analysis	• DIN SPEC 91391-2 leaves many things for the implementer to decide. OpenCDE-API interactive flow limits automation. • The OpenCDE metadata can express ICDD metadata. • The LDP's flexibility allows us to use it for ICDD data.
Senthilvel et al., Implementing Information Container for linked Document Delivery (ICDD) as a micro-service., 2021.	• How CDEs benefit from ICDD containers, which special focus on structuring and linking heterogeneous data on meta and sub-document levels?	The Design Science Research Method: Analytical Static Analysis	• DIN SPEC 91391-2 leaves many things for the implementer to decide. OpenCDE-API interactive flow limits automation. • The OpenCDE metadata can express ICDD metadata. • The LDP's flexibility allows us to use it for ICDD data.
Oraskari et al., Enabling Federated Interoperable Issue Management in a Building and Construction Sector, 2022	• How can a single source of truth over the project data and the advantages of the central approach be realised in a distributed setup, enabling decentralised authentication and stakeholders' ability to control their data?	The Design Science Research Method: Analytical Static Analysis	• We proposed a tree-like structure that combines the traits of BCF XML and the BCF API • BCF project information can be queried using a one-stop access point implemented using distributed Solid architecture
Werbrouck et al., A Generic Framework for Federated CDEs applied to Issue Management, 2023	• How can a single source of truth over the project data and the advantages of the central approach be realised in a distributed setup, enabling decentralised authentication and stakeholders' ability to control their data?	The Design Science Research Method: Analytical Static Analysis	• A federated framework for project-related, interdisciplinary collaboration for CDEs is presented. • A generic data patterns for issue management that can - be aggregated - restructured to match industry standards like BCF API.

Fig. 4.2. My articles related to publishing data as CDE at a glance

4.1 Common Data Environment Concepts

4.1.1 Common Data Environmen (CDE)

A Common Data Environment (CDE) is a central location where all data and documents related to a specific project or organisation are stored and managed. A CDE aims to provide a single source of truth for project data, ensuring that all stakeholders have access to the latest and most accurate information. A CDE typically includes document management, version control and collaboration tools that allow team members to easily share and collaborate on project data. It may also include tools for reporting, data analysis and integration with the organisation's systems and tools. In general, a CDE improves data management efficiency by providing a central, organised and secure location for storing and accessing project data. This can help teams reduce the risk of errors, avoid duplication of effort and make better informed decisions.

According to ISO 19650-1:2018[International Organization for Standardization, 2018b], a Common Data Environment (CDE) consists of two parts. The first aspect is that it is a source of information about any matter of value to a company or organisation. This can be building information models of the building elements, asset issue information and product data. It can be seen as a single point of access to the project data for the project members, i.e. when a user searches for the description of an item, he or she can be sure that it is the most up-to-date knowledge of that item. The second aspect is that the information is brought together in a standard format in named records where it is managed. The naming conventions of the containers are intended to help locate relevant information from the data repository and to aid in grouping and versioning, while the data standards and structure are intended to aid in its consumption.

The currently existing CDE implementations listed by Guerrero [2022] are Autodesk BIM 360, Connect,

ProjectSight, Trimble Viewpoint, Oracle Aconex, Bentley Project Wise, Think Project, Dalux and Acca Software's usBIM.platform. In addition, the TNO Building Information Model Server, BIMServer, can be considered a CDE.

In the Linked Data paradigm, where each data element has a unique identifier that can be dereferenced, the representation of construction data as Linked Data implements the idea of a 'single source of information' at the data element level. The Drumbeat platform is an example of this approach. In the following sections, I will also show how container models can be published as CDEs and how the planned Stage 3 maturity level of ISO 19650 can be addressed.

4.1.2 ISO 19650 Stage 3

ISO 19650 [International Organization for Standardization, 2018a] is a series of international standards that provide guidelines for Building Information Modelling (BIM) in the construction, operation and maintenance of built assets. The series includes three levels of maturity: 1, 2 and 3.

Level 3 is the intended most advanced level of BIM maturity. It involves the use of BIM throughout the lifecycle of a built asset, from planning and design to construction, operation and decommissioning. At this level, BIM is used to its full potential, providing a comprehensive and integrated view of the asset that can be used for decision making and project management.

To achieve Level 3, an organisation must have implemented BIM processes and technologies across all project lifecycle phases and demonstrated their effectiveness through case studies or other evidence. It also requires the integration of BIM with other business processes and systems, and the development of a culture of collaboration and innovation within the organisation.

ISO 19650 Stage 3 aims to achieve the highest defined level of digital collaboration maturity. When comparing Stage 2 and Stage 3, the most notable difference is that Stage 3 systems (1) are no longer file-centric but data-driven, i.e. the information is at the core and the software architecture only facilitates its use (2) uses a database and (3) allows the data to be queried. Allows the data to be queried. These are all characteristics of Linked Data. Everything else is already at level 2.

I have chosen to examine four criteria related to ISO 19650 BIM Stage 3 maturity.

1. The system should comply with the buildingSMART openBIM principles.
2. Instead of being file-based, data should be accessible as a database (RDF storage), and data should be federated and searchable.
3. Consideration of the ISO 19650 MVD Exchange specifications and data states.

4.2 Common Data Environments for the Information Container for linked Document Delivery (LDAC 2020)

Image a scenario where a facility manager Bob gets a set of as-built model data of a building packaged as a standard ISO 21597 Information Container for Document Delivery (ICDD) file. He wants all the given information to be integrated with existing knowledge so that he has a consistent view of the current status of all assets of the supported premise. The new facility management system implements buildingSMART OpenBIM standards. So, he uploads the file by using the buildingSMART OpenCDE Document API. When his colleague Alice accesses the data later to analyze the energy efficiency of the building, she knows that the provided data is the most current and complete knowledge of the state of the property and so does not have to search for other data sources.

Hypothesis 4.1 claimed that building information shared as a standard ICDD container can be published online using the buildingSMART OpenCDE, and W3C Linked Data Platform (LDP) standards, and it increases the accessibility of the information compared to a file-based standard. A metadata mapping was presented, elucidating the process of publishing identical data in the selected standards. As the metadata and contained data were projected without any data loss, you could claim that the ICDD containers can be published on both online systems.

The idea of the single source of truth (SSOT) for the organisations data assets is in the CDE concept in the ISO 19650 standard [International Organization for Standardization, 2018b]. Therefore, an online CDE publication offers an SSOT for the organisations data assets. The publication improves access to the data when compared with the file-based ICDD containers. Online access also allows linking data content to any relevant content online, improving relevant data access. These can be ontologies to interpret the projects content, product catalogues, and status data. Furthermore, while a file can contain links to external sources of information, the online publication allows bidirectional linking, which can be beneficial when, e.g., notifying status changes of elements or changes in a product catalogue.

Table 4.1 summarises the use of the design science research method in this research.

My contribution to the paper: The article was written together with Madhumitha Senthilvel and supervised by professor D.Sc. Jakob Beetz. We shared the work so that Senthilvel focused on the Linked Data Platform while I focused on Common Data Environment (CDE) and the OpenCDE API. Discussion and conclusions were made together.

Table 4.1. The work of Senthilvel et al. [2020] in the Hevner et al. [2004] design science research framework

Guideline 1: Design as an artifact

Mapping rules from ICDD definition metadata to W3C Linked Data Platform(LDP) and OpenCDE specifications (German DIN SPEC 91391-2, and buildinSMART OpenCDE) were presented. The used upper abstraction level is independent of any implementation framework. It was also shown how the non-interactive version of the buildingSMART OpenCDE for a ICDD file can be realised using the existing information in the ICDD container.

Guideline 2: Problem relevance

Some parties use ICDD containers to share information about the projects. They are using files, so there is no guarantee of a single source of truth about the project assets. The business problem is how these could be published online so that the benefits of network access could be gained and the SSOT feature realized.

The constituent user community of the artefact is the researchers of the Linked Building Data Community and software architects of a Common Data Environment.
The solution belongs to the Common Data Environment and conversion problem classes.
This work is an enabler for the Common Data Environment problem class, which explores ways to house construction project information to collect, manage, collaborate and share information for the artefact organization.

Guideline 3: Design evaluation

Hevner et al. Design Research Evaluation Method: Analytical Static Analysis as used as an evaluation method.

[0] https://www.w3.org/TR/ldp/

Guideline 4: Research contributions

The results provide the research community with the knowledge of how the meta information of ICDD can be mapped to LPD and OpenCDE and how ICDD containers could be presented on these CDE services. The benefit is that the popular container model can not only be used to publish the content online so that the contained Linked Data can be linked to but also, the data can be served as a single source for the information.

Guideline 5: Research rigor

The work was peer-reviewed.

Guideline 6: Design as a search process

Guideline 7: Communication of research

International Workshop on Linked Data in Architecture and Construction(LDAC),2020.

The work was made available as an open-access publication under the Creative Commons License Attribution 4.0 International (CC BY 4.0 [1]). The Linked Data Architecture and Construction Workshop 2020 editors, RWTH Aachen, CEUR-WS and archiving partners were given irrevocable rights to make the work available. The work is originally published in [Senthilvel et al., 2020]. The content presented here is based on our final draft of the article.

Madhumitha Senthilvel Department of Design Computation, RWTH Aachen University, Aachen, Germany, senthilvel@dc.rwth-aachen.de
Jyrki Oraskari Department of Design Computation, RWTH Aachen University, Aachen, Germany, Jyrki.Oraskari@dc.rwth-aachen.de
Jakob Beetz Department of Design Computation, RWTH Aachen University, Aachen, Germany, j.beetz@caad.arch.rwth-aachen.de

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

4.2.1 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

[1] https://creativecommons.org/licenses/by/4.0/legalcode

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 [D'Esposito, 2019].

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) [Fuchs et al., 2010] was originally defined by Scherer et al.[Scherer and Schapke, 2011, Scherer and Katranuschkov, 2019] in the Mefisto project for multi-model exchange [Schapke, 2020]. 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[2] (ICDD) [ISO, 2018a,b] is an upcoming standard for multi-model container approach that also allows using Linked Data (LD) [Bizer et al., 2009, 2011] and also Linked Building Data[3] (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) [Beckett and McBride, 2004] 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) [Council, 2013] 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)[Malhotra et al., 2015], 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

[2]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"

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

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.

4.2.2 Concepts and Related work

Information container for linked document delivery

ICDD is described in ISO 21597 [ISO, 2018a,b] 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. 4.3): which three major components: an Ontology folder containing the schema of the files in the ICDD Zip file format [Katz, 1993], 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 [Bizer et al., 2009a].

Common Data Environment

According to the yearly industry report [Erik Thomas and Spare, 2018] 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

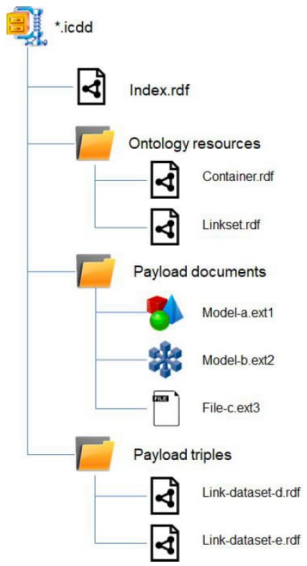


Fig. 4.3. The ICDD folder structure

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 [Council, 2013] and BS 1192 [Richards, 2010]. The documents describe the BIM Level 2 compliance requirements for construction company projects. [Boxall, 2018]

OpenCDE German DIN SPEC 91391-2 [DIN, 2019] 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 [Daigneau, 2012]. The interface is specified in OpenAPI 3.0 format and defines a RESTful [Fielding, 2000] 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[4] .

Linked Data Platform LDP is a specification containing definitions for reading-writing Linked Data architecture. As Bizer summarized in [Bizer et al., 2009], "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 [Berners-Lee, 2008]. The rules are: 1. Uniform Resource Identifiers (URIs) are used to name things uniquely. 2. Hypertext Transfer Protocol (HTTP) [Fielding et al., 1999] 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) [Manola et al., 2004] format and SPARQL Query Language [Seaborne et al., 2008]. 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 [Fielding, 2000, Hausenblas, 2009, Malhotra et al., 2015].

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. [The World Wide Web Consortium, 2019]. A very recent initiative, SOLID [Berners-Lee et al., 2017] 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.

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

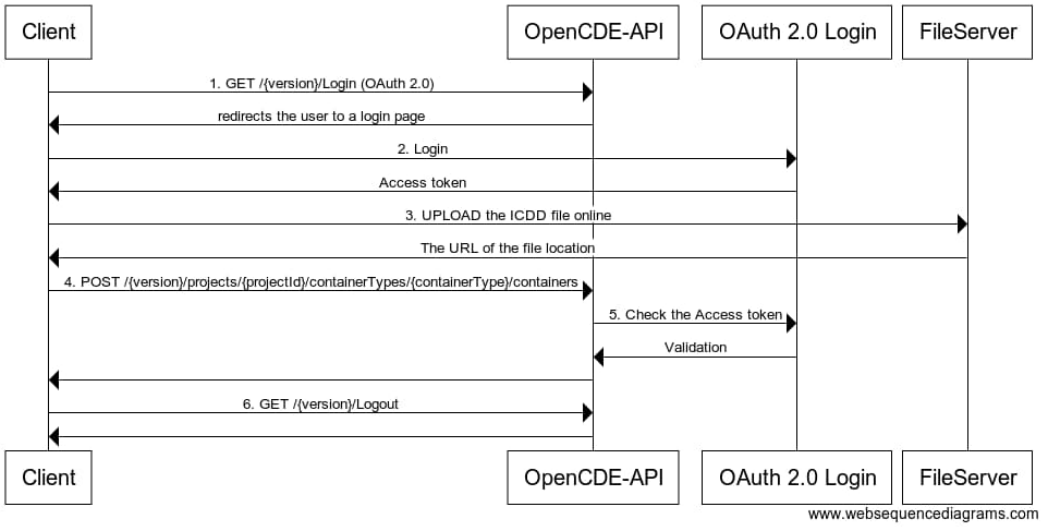


Fig. 4.4. Publish an ICDD container into a DIN SPEC 91391-2 Server

4.2.3 ICDD in the CDE Context

ICDD in DIN SPEC 91391-2

Multi-containers are specified as a container type in DIN SPEC 91391-2. BIM-LV containers [DIN, 2016] are explicitly mentioned in the standard [DIN, 2019] and according to DIN SPEC 91350 are specific version of the multi-model container and correspond to the multi-model scheme published at [Schapke, 2020].

ICDD is also a multi-model and can be interpreted addressed implicitly at the standard [Scherer and Katranuschkov, 2019]. For consistency, all multi-models should be published in the same 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. 4.4. 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

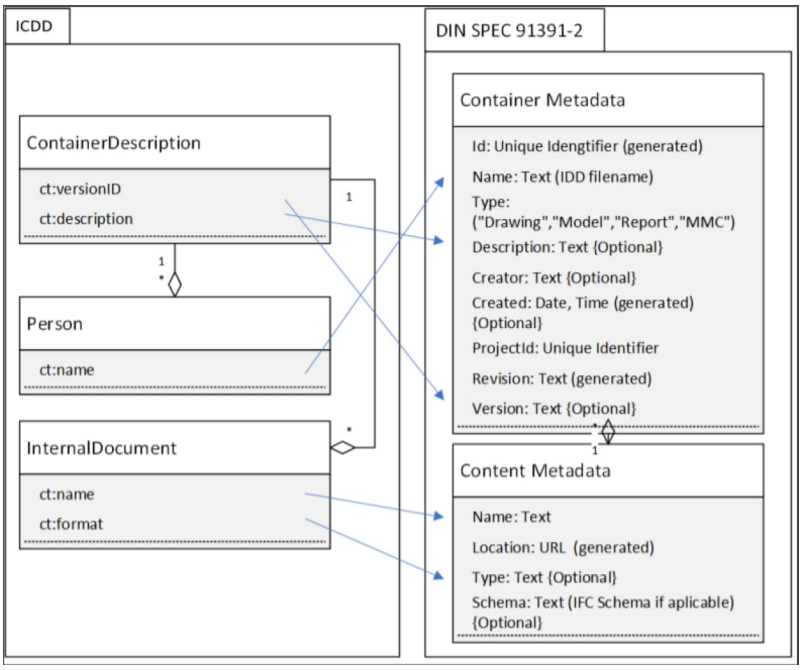


Fig. 4.5. Mapping between ICDD and the DIN SPEC 91391-2 OpenCDE metadata

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. 4.5 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 [Chappell, 2011] filters specified in DIN SPEC 91391-2 [DIN, 2019] offers an implementer specific ways to filter API results by metadata. OData contains operator to 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 [Werbrouck et al., 2019], the queried ICDD containers can be queried using the GraphQL[5]. The setup is shown in Fig. 4.6. Steps 1, 2, and 4 are for OAuth 2.0 user authentication. Step 3 gets a list of the ICDD containers

[5]https://graphql.org/

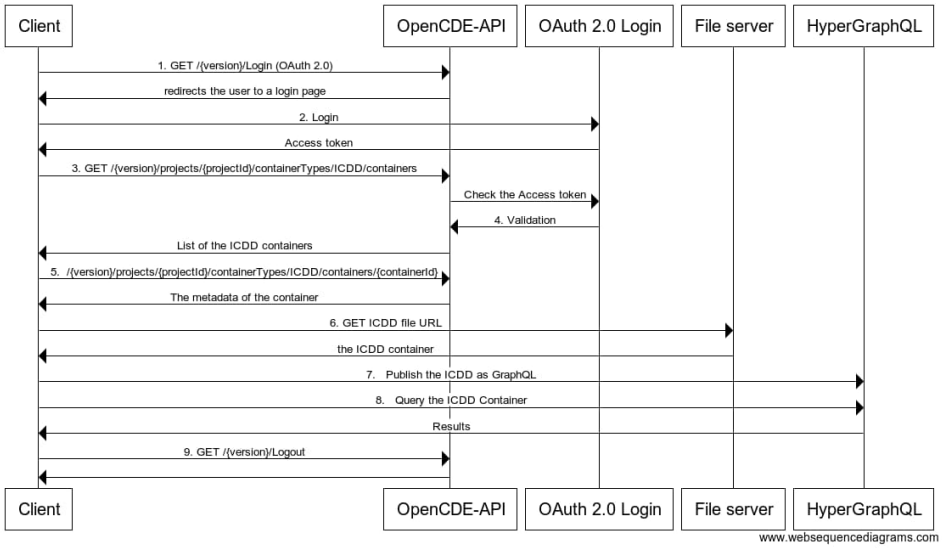


Fig. 4.6. Query an ICDD container published in a DIN SPEC 91391-2 Server

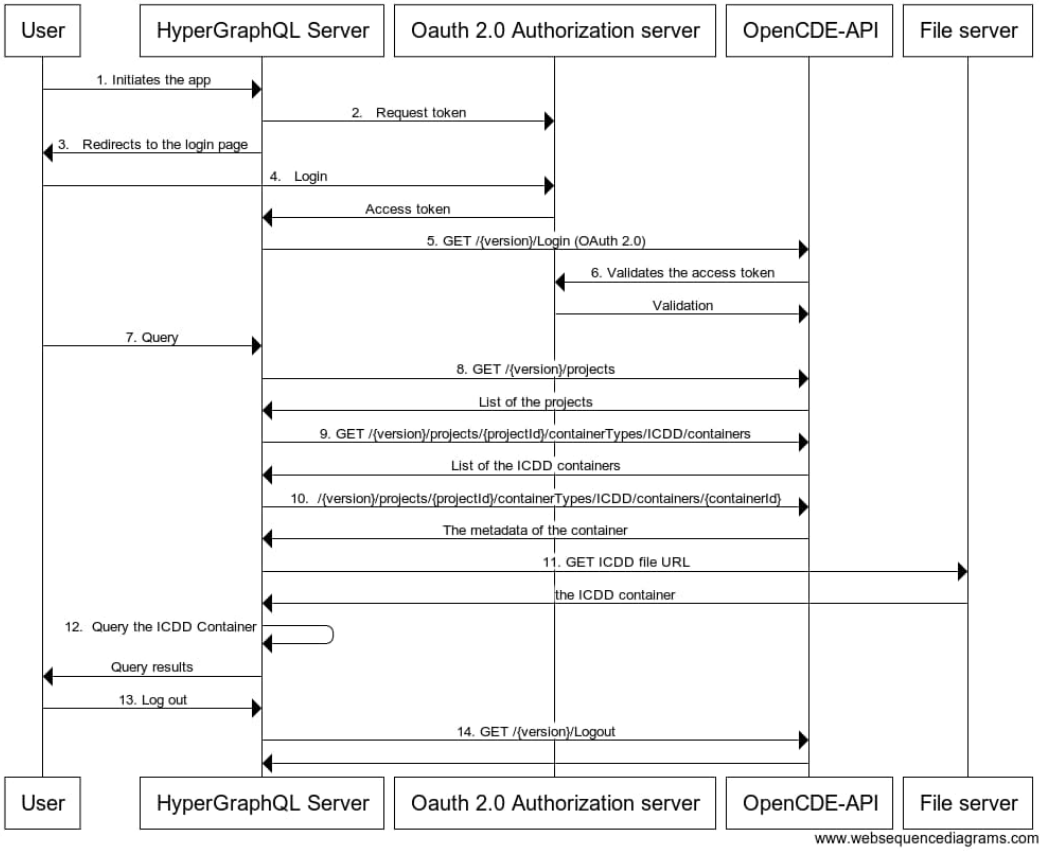


Fig. 4.7. Query an ICDD container published in a DIN SPEC 91391-2 Server

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 [Werbrouck et al., 2019] 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. 4.7. 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 [Wittern et al., 2018], 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 [Werbrouck et al., 2019].

ICDD in buildingSMART OpenCDE-API

As illustrated in Fig. 4.8, 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. 4.9) 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. 4.10 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.

ICDD in Linked Data Platform

LDP defines containers as a special resource, which has the capability to respond to HTTP requests [Fielding et al., 1999] and modifications including, including the addition of new resources. There are two containers: *Basic Container*, and a

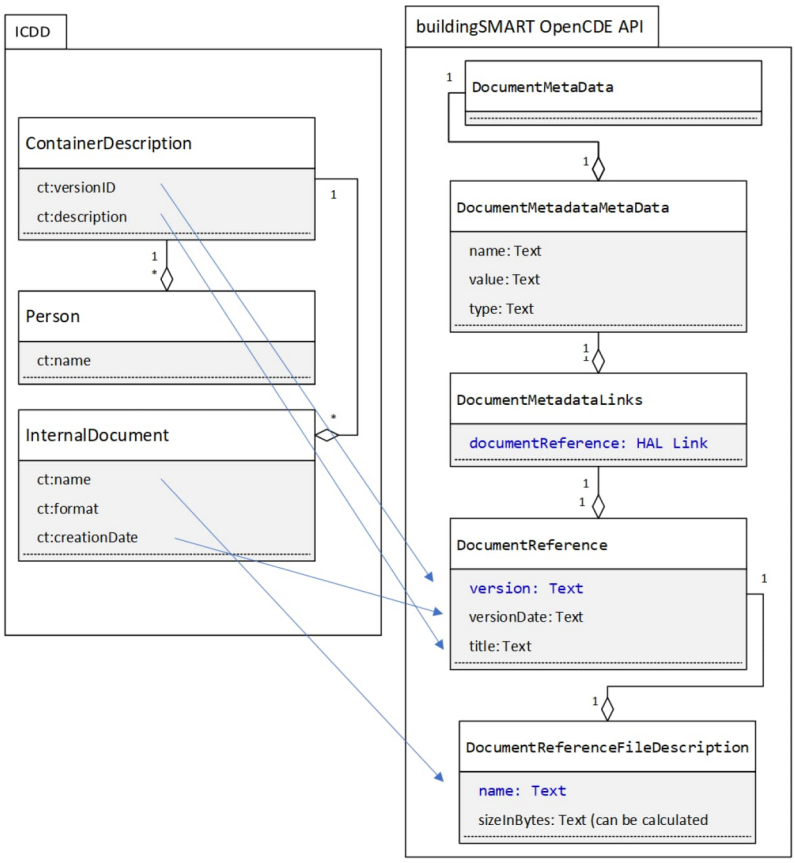


Fig. 4.8. Mapping between ICDD and the buildingSMART OpenCDE-API metadata

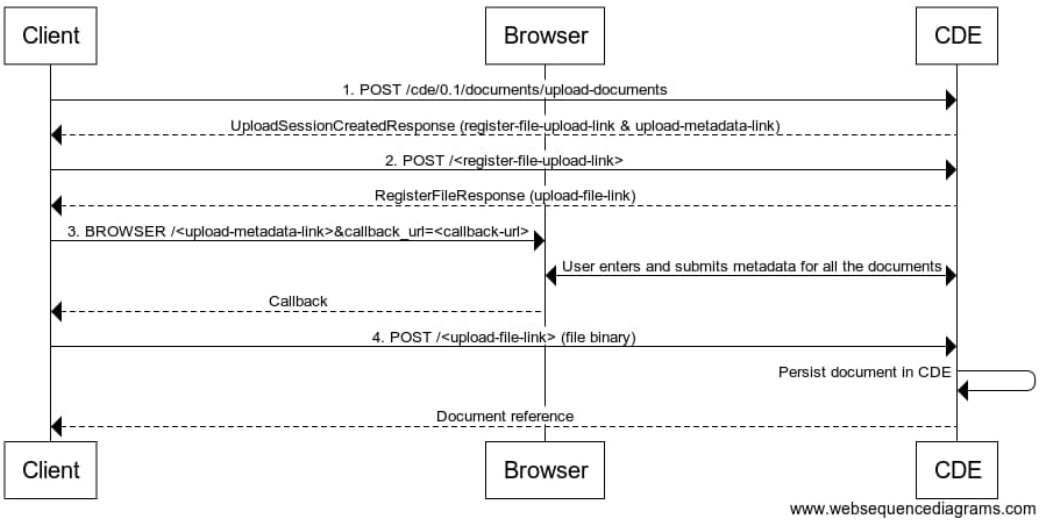


Fig. 4.9. Steps to upload an ICDD into buildingSMART OpenCDE-API

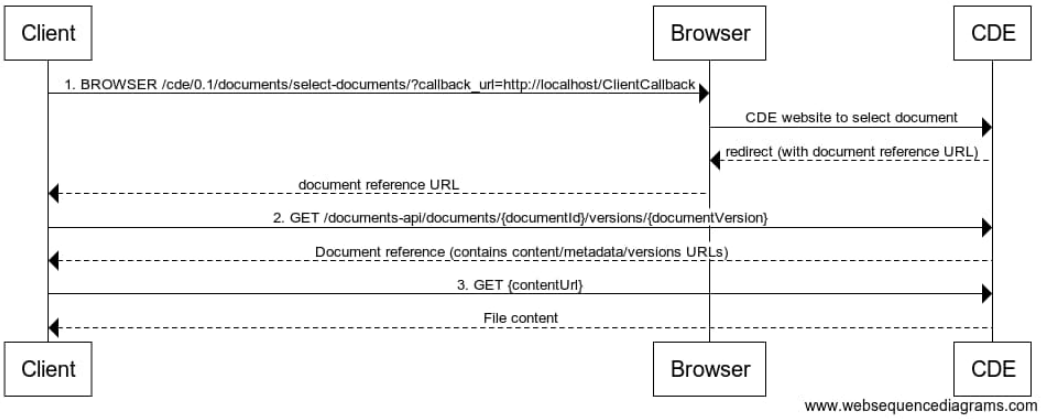


Fig. 4.10. Actions needed to fetch an ICDD from the OpenCDE-API

Direct Container. The *Basic Container* can define links to a *Document resource* (both 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. 4.12 shows the workflow for publishing ICDD containers in the SOLID ecosystem, while Fig. 4.11 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 [Senthilvel, 2020].

In Fig. 4.4 and Fig. 4.11, the general operations for publishing and accessing data for further operations like querying were shown. Fig. 4.13 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.

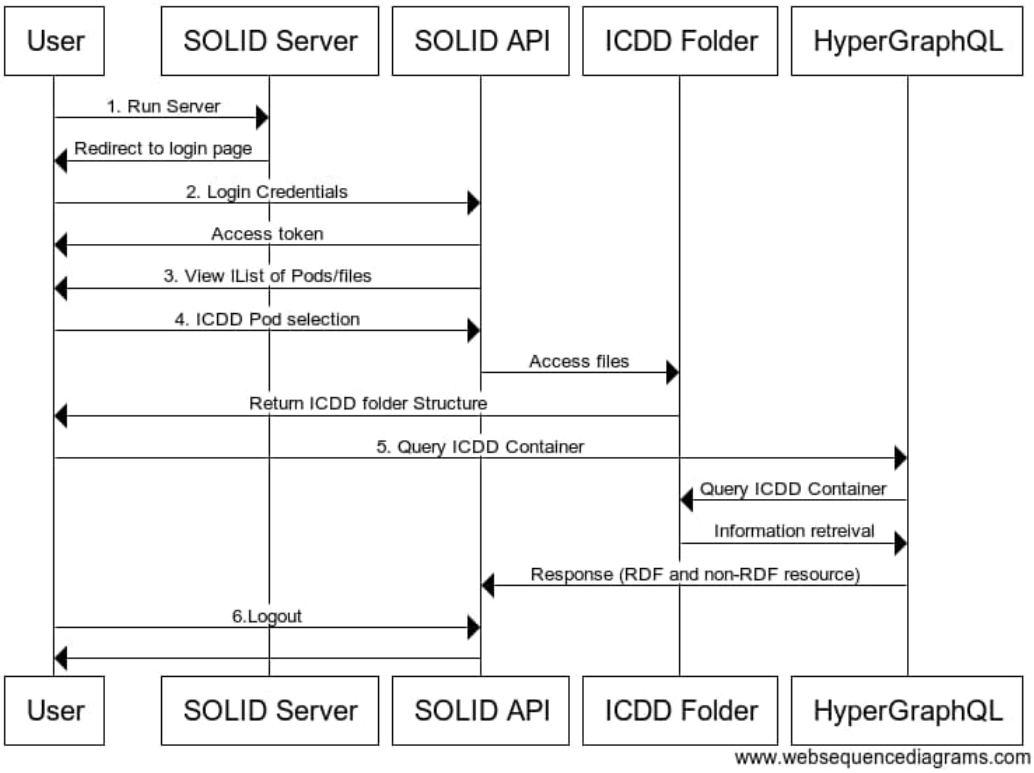


Fig. 4.11. Querying ICDD Container in SOLID

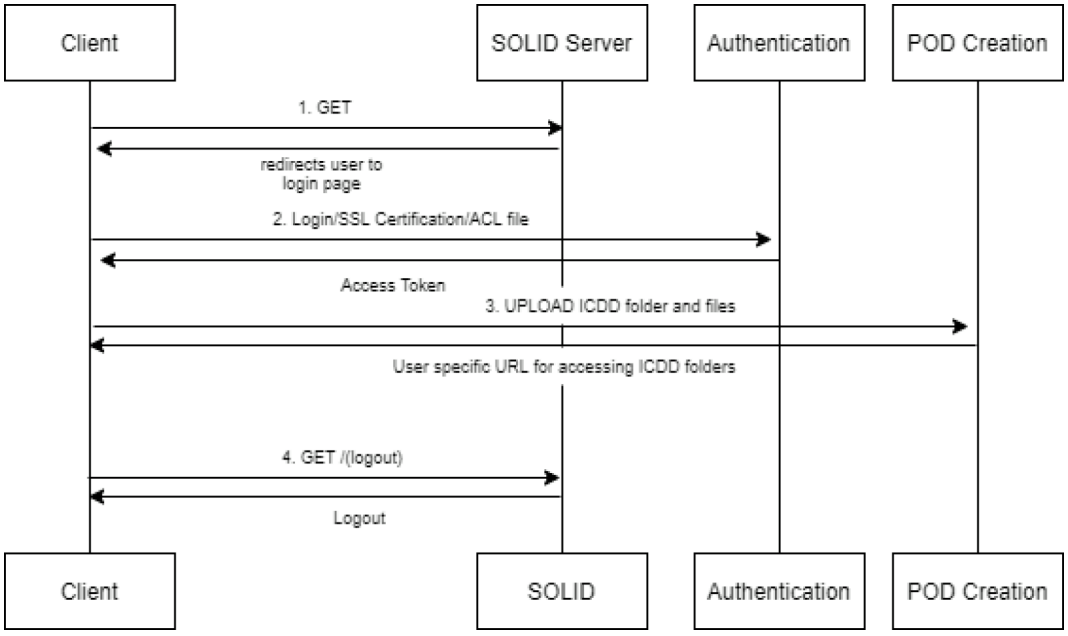


Fig. 4.12. Publishing ICDD Container in SOLID

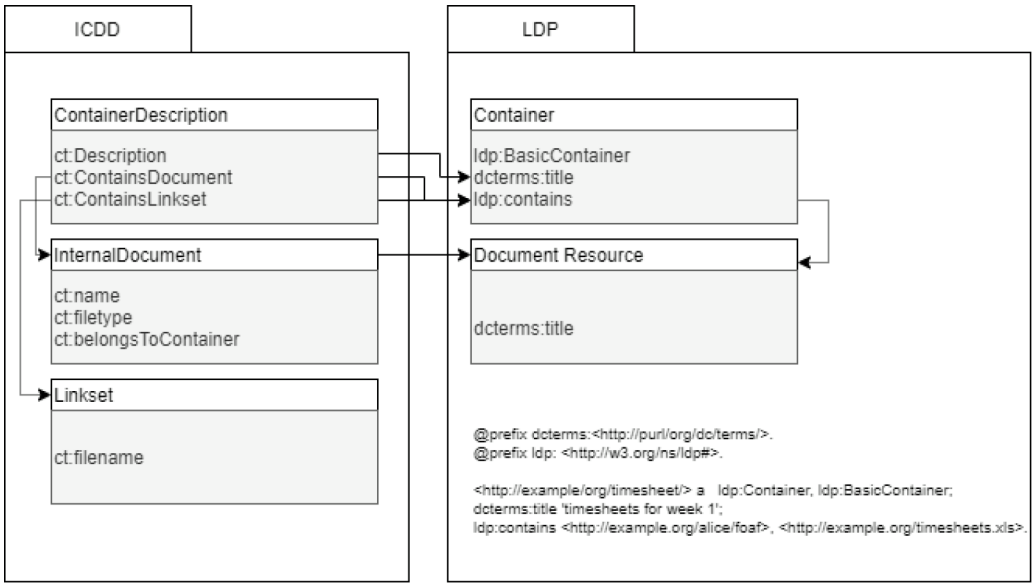


Fig. 4.13. Mapping between ICDD and LDP Container Concepts

4.2.4 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 [Bader et al., 2017], while Oracle Aconex[6] 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 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." [Speicher et al., 2015]. 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.

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

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 using 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.

4.2.5 Acknowledgements

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

References

Sebastian R. Bader, Alexander Wolf, and Felix Leif Keppmann. Evaluation environment for linked data web services. In *SEMANTICS Workshops*, 2017.

D. Beckett and B. McBride. Rdf/xml syntax specification (revised). *W3C recommendation*, 10, 2004.

Tim Berners-Lee. Linked Data - Design Issues, June 2008. URL https://www.w3.org/DesignIssues/LinkedData.html. Online; accessed Oct. 15. 2023.

Tim Berners-Lee, Eric Prud'hommeaux, Melvin Carvalho, and Ruben Verborgh. Solid, 2017. URL https://solid.mit.edu/. Online; accessed Oct. 15. 2023.

Christian Bizer, Tom Heath, and Tim Berners-Lee. Linked data - the story so far. *International Journal on Semantic Web and Information Systems*, 5:1–22, 2009a.

Christian Bizer, Tom Heath, and Tim Berners-Lee. Linked data-the story so far. *Semantic services, interoperability and web applications: emerging concepts*, pages 205–227, 2009b.

Christian Bizer, Tom Heath, and Tim Berners-Lee. Linked data. *Evolving the Web into a Global Data Space*, 2011.

E. Boxall. 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-2019], Oracle.

David Chappell. Introducing odata. *Data Access for the Web, The Cloud, Mobile Devices, and More*, pages 1–24, 2011.

Construction Industry Council. Pas1192-2: 2013 specification for information management for the capital/delivery phase of construction projects using building information modelling. https://www.hfms.org.hu/joomla/images/stories/PAS/PAS1192-2-BIM.pdf, 2013. Online; Accessed 17-oct-2023,London: British Standards Institution.

Robert Daigneau. *Service Design Patterns: fundamental design solutions for SOAP/WSDL and restful Web Services*. Addison-Wesley, 2012.

Henry D'Esposito. The state of construction technology - 2019, 2019.

DIN. Din spec 91350:2016-11, verlinkter bim-datenaustausch von bauwerksmodellen und leistungsverzeichnissen, 11 2016.

DIN. Din spec 91391-2:2019-04, gemeinsame datenumgebungenä(cde) für bimprojekteä funktionen und offener datenaustausch zwischen plattformen unterschiedlicher herstelleră teilă2: Offener datenaustausch mit gemeinsamen datenumgebungen, April 2019.

Jay Bowman Jay Snyder Erik Thomas, Peter Schott and Natalie Spare. 2018 industry report: Construction disconnected, 2018.

Roy Fielding, Jim Gettys, Jeffrey Mogul, Henrik Frystyk, Larry Masinter, Paul Leach, and Tim Berners-Lee. Hypertext transfer protocol–http/1.1. https://www.w3.org/Protocols/rfc2616/rfc2616.html, 1999a. Online; accessed Oct. 17. 2023.

Roy T. Fielding, James Gettys, Jeffrey C. Mogul, Henrik Frystyk Nielsen, Larry Masinter, Paul J. Leach, and Tim Berners-Lee. Hypertext transfer protocol – http/1.1. RFC 2616, RFC Editor, June 1999b. URL http://www.rfc-editor.org/rfc/rfc2616.txt. Accessed on Jun. 06. 2019.

Roy Thomas Fielding. *Architectural styles and the design of network-based software architectures*. University of California, Irvine, 2000.

S Fuchs, P Katranuschkov, and RJ Scherer. A framework for multi-model collaboration and visualisation. *Proc. ECPPM 2010*, pages 115–120, 2010.

Michael Hausenblas. Linked data applications. First Community Draft, DERI, 2009. Citeseer.

ISO. Information container for linked document delivery Exchange specificaton - Part 1: Container, March 2018a.

ISO. Information container for linked document delivery Exchange specificaton - Part 2: Dynamic semantics, March 2018b.

Phil Katz. Appnote. *TXT. ZIP file format specification, version*, 2, 1993.

A. Malhotra, J. Arwe, and S. Speicher. Linked data platform specification. *W3C Recommendation*, 2015.

Frank Manola, Eric Miller, Brian McBride, et al. Rdf primer. *W3C recommendation*, 10(1-107):6, 2004.

M. Richards, Richards. Building information management: A standard framework and guide to bs 1192, bsi standards. In *Building Information Management: A Standard Framework and Guide to BS 1192 Paperback 31 July 2010*. BSI British Standards Institution, 2010.

Sven-Eric Schapke. Mmc: buildingsmart mmc multimodell-container. In *GitHub*. Zenodo, March 2020. doi: 10.5281/zenodo.3727286. URL https://doi.org/10.5281/zenodo.3727286. Online; accessed Oct. 15. 2023.

Raimar J. Scherer and Peter Katranuschkov. Context capturing of multi information resources for the data exchange in collaborative project environments. *Proceedings of the 2019 European Conference on Computing in Construction*, 2019.

Raimar J. Scherer and S.-E. Schapke. A distributed multi-model-based management information system for simulation and decision-making on construction projects. *Advanced Engineering Informatics*, 25(4):582–599, 2011. Elsevier.

Andy Seaborne, Geetha Manjunath, Chris Bizer, John Breslin, Souripriya Das, Ian Davis, Steve Harris, Kingsley Idehen, Olivier Corby, Kjetil Kjernsmo, et al. Sparql/update: A language for updating rdf graphs. *W3c member submission*, 15, 2008.

Madhumitha Senthilvel. Icdd hypergraphql. In *GitHub*. GitHub, Zenodo, May 2020. doi: 10.5281/zenodo.3846727. URL https://github.com/Design-Computation-RWTH/ICDD-HyperGraphQL. Online; accessed Oct. 15. 2023.

Steve Speicher, John Arwe, and Ashok Malhotra. Linked data platform 1.0. *W3C Recommendation, February*, 26, 2015.

The World Wide Web Consortium. LDP implementations - w3c wiki, 2019. URL https://www.w3.org/wiki/LDP_Implementations. Online; accessed Oct. 15. 2023.

Jeroen Werbrouck, Madhumitha Senthilvel, Jakob Beetz, and Pieter Pauwels. Querying heterogeneous linked building datasets with context-expanded graphql queries. In *7th Linked Data in Architecture and Construction Workshop*, pages 21–34, 2019.

Erik Wittern, Alan Cha, and Jim A. Laredo. Generating graphql-wrappers for rest (-like) apis. In *International Conference on Web Engineering*, pages 65–83. Springer, 2018.

4.3 Implementing Information Container for linked Document Delivery (EG-ICE 2021)

Image a scenario where a facility manager Bob gets a set of updated as-built model data of a building packaged as a standard ISO 21597 Information Container for Document Delivery (ICDD) file. The new facility management system implements buildingSMART OpenBIM standards. So, he uploads the file by using the buildingSMART OpenCDE Document API. When his colleague Alice accesses the details of a broken space heater later, she knows that the provided data is up to date and complete. When ordering a replacement, she can include a link to the data so the spare part supplier can offer a replacement product.

Hypothesis 4.1 claimed that building information shared in a standard ICDD can be published online using the buildingSMART OpenCDE and W3C W3C Linked Data Platform (LDP) standards, and it increases the accessibility of the information compared to a file-based standard. The work is a continuation of the earlier work and shares the reasoning about the hypothesis of Section 4.2. However, in the implementation, I faced the immaturity of the buildingSMART OpenCDE standard; i.e., the non-interactive branch of the standard was not available. Therefore, an extra API interface was designed and implemented to automate metadata import. This serves as a proposal for how the non-interactive standard half could be materialized.

Table 4.2 summarizes the use of the design science research method in this research.

My contribution to the paper: The article was written together with Madhumitha Senthilvel and supervised by professor D.Sc. Jakob Beetz. We shared the work so that Senthilvel focused on the Linked Data Platform while I focused on the Common Data Environment (CDE) and the OpenCDE API. I also wrote the OpenCDE-API ICDD implementation and evaluated it. Discussion and

conclusions were made together.

Table 4.2. The work of Senthilvel et al. [2021] in the Hevner et al. [2004] design science research framework

Guideline 1: Design as an artifact
A comparison of the implementations was made at the functionality level. As proof of construction, two RESTful implementations, based on LDP and OpenCDE-API, were created to support ICDD file uploads. They were written in JavaScript using the Node.js framework.

Guideline 2: Problem relevance
This continues the work of Section 4.2. Therefore the business problem is the same: Some stakeholders use ICDD containers to share project information. They are using files, so there is no guarantee of a single source of truth (SSOT) about the project assets. The business problem is how these could be published online so that the benefits of network access could be gained and the SSOT feature realized. The constituent user community of the artefact is the researchers of the Linked Building Data Community and software architects of a Common Data Environment. The solution belongs to the Common Data Environment and conversion problem classes. This work is an enabler for the Common Data Environment problem class, which explores ways to house construction project information to collect, manage, collaborate and share information for the organization of the artefacts.

Guideline 3: Design evaluation

Hevner et al. Design Research Evaluation Method: Analytical Static Analysis was used to evaluate the results of the work.

Guideline 4: Research contributions

As the non-interactive part of the standard was still missing, this provides an analyse of how the standard could be developed further: (1) to allow non-interactive mode, and (2) to publish ICDD data online as a single source of information.

Guideline 5: Research rigor

The work was peer-reviewed.

Guideline 6: Design as a search process

The criteria for the work were to find a solution that combines the ISO 21597 ICDD and CDE standards. buildingSMART OpenCDE-API was selected as the CDE standard, as it is the open industry standard of buildingSMART International and offers an OpenAPI definition, which suits well for development.

Guideline 7: Communication of research

International Workshop on Intelligent Computing in Engineering(EG-ICE), 2021.

We were granted the non-exclusive right of use to the Technische Universität Berlin and it does not affect the right of the author to publish the work in other ways. Also, we agreed to the publication of the work under the Creative Commons license CC BY 4.0: Creative Commons Attribution 4.0 [7]. The work is originally published in [Senthilvel et al., 2021]. The content presented here is based on our final draft of the article.

Madhumitha Senthilvel Department of Design Computation, RWTH Aachen University, Aachen, Germany, senthilvel@dc.rwth-aachen.de
Jyrki Oraskari Department of Design Computation, RWTH Aachen University, Aachen, Germany, Jyrki.Oraskari@dc.rwth-aachen.de
Jakob Beetz Department of Design Computation, RWTH Aachen University, Aachen, Germany, j.beetz@caad.arch.rwth-aachen.de

Abstract: Common Data Environments (CDE) are seen as the next steps towards a truly open, accessible and interoperable BIM. Information Containers form the core of component based on which CDEs store, manage, exchange and process data with stakeholders in a project. While specifications for information containers have gradually evolved with progressive standards, the recently introduced ISO 21597 standard for Information Containers for linked Document Delivery contains specifications for accommodating linked building data management. Since this standard is intended for file-based information exchange, in this paper we explore the implementation of ICDD using two web-based CDEs: OpenCDE-API and Linked Data Platform. For this, the functional and operational requirements for using ICDD in both the CDEs are summarized, which are then implemented in a minimalistic prototype for each CDE.

4.3.1 Introduction

With the gradual establishment of Building Information Modelling (BIM) for information management across all phases of construction projects, there has been increasing interest in the emphasis on the approach of Common Data Environments (CDEs) (Bucher and Hall, 2020). These environments facilitate interoperability between different authoring tools and formats through specifications for standardized information storage, access rights, information management etc. At the core of CDEs lies the concept of information containers, which are used to structure meta-information on the data residing within them. In their most rudimentary form, they are virtual enclosures to store relevant data together. These data can be uploaded files, files stored in the CDE itself, or even graphs from Triple Stores.

[7] http://creativecommons.org/licenses/by/4.0/

The notion of containers and CDEs are closely entwined with many existing CDEs containing specifications focusing on Container data representation and management. This can be traced back to the fundamental intention of CDEs: to enable collaboration, and centralization of information management. Presently available CDE standards and approaches such as the ISO19650[8], DIN SPEC 91391-2[9], Linked Data Platform (LDP)[10], Open CDE-API[11] have varying levels of specifications for structuring these containers. In addition, a recently introduced standard ISO 21597 Information Containers for linked Document Delivery (ICDD)[12], also focuses on multi-model containers. This approach is particularly relevant in the context of a decentralized open-world BIM, wherein containers and the information inside them also be used for linking disparate information.

In our previous work, we evaluated how the concepts in ICDD can be leveraged in the existing CDEs introduced earlier (Senthilvel et al., 2020). One of the main issues identified in the previous work was that ICDD is intended for file-based containers, which store the ontology, payload documents, and the links between these documents in RDF serialization in folders. In this paper, we explore how these container concepts can be implemented in a decentralized web, with all the documents existing as parts of the graph and de-referenceable on the web, accessible by URIs. An implementation for this approach is also briefly presented to demonstrate the concept. In the end, this implementation is intended to serve as a micro-service, to be plugged into other CDEs, for extending their container systems.

4.3.2 Background

Since the concept of Information Containers has been closely tied with CDEs, there exist numerous commercial implementations such as BIM 360, Oracle Aconex, Trimble Connect, etc. However, most of these are designed as document management systems, for storing and sharing information as files. While some such as BIM 360 contain added meta-data for version tracking, author information, issue tracking etc., at present there is no mechanism for linking of documents within the containers (Gumpert, 2019). As mentioned in the

[8]Builds on the previous PAS1192:2003 standard with additional specifications for meta-data for container

[9]German standard defining OpenCDE through requirements for communication interface between CDEs and software

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

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

[12]Formerly “Information Container Data Drop”

previous section, ICDD provides an approach which uses linked data for creating such linking of information. However, in order to use ICDD in a CDE, the standard does not itself contain specifications such as the use of HTTP requests for information orchestration over the web. Consequently, from the existing standards already introduced in the earlier section, we selected Linked Data Platform (LDP) and OpenCDE-API as candidates for implementing ICDD as a web-based microservice.

For easier comprehension, this section presents a brief introduction to the ICDD standard, OpenCDE-API approach and the Linked Data Platform.

ISO 21597:2020 - Information Container for linked Document Delivery (ICDD)

The ISO 21597 (from here on referred to as ICDD) is a two-part standard containing specifications for linking documents using Linked Data approach and the container which contains this information. Its conceptual origins can be traced to the multi-model containers from the Mefisto project and the COINS approach, which introduced preliminary container specifications for managing multiple models (van Nederveen, 2010). Part 1 of this 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 link set ontology with corresponding data types and properties (Technical Committee : ISO/TC 59/SC 13 Organization and digitization of information about buildings and civil engineering works, including building information modelling (BIM), 2020). Part 2 defines additional types of links which form
the extended link set (Technical Committee : ISO/TC 59/SC 13 Organization and digitization of information about buildings and civil engineering works, including building information modelling (BIM), 2020).

The container a folder structure with three major components: an Ontology folder containing the schema of the files in the ICDD Zip file format, a link set folder containing the links, and a payloads folder containing the documents themselves. Figure 1 provides a visual representation of a sample ICDD container.

OpenCDE-API

OpenCDE-API is a buildingSMART initiative, which aims to improve the interoperability within the AEC CDE software ecosystem through closely woven, domain-specific APIs. A project supporting OpenCDE-API would effectively be able to stream information (in containers) between different CDEs with negligible loss in meta-data by providing a common interface standard. The proposed framework contains a host of APIs: Foundations (for authentication, Authorization, and Conventions), BCF (for design issues tracking), Documents (for information management), Data Exchange (for proper exchange at component/object level), etc.[13]. Of these APIs, the Documents API and the Data Exchange API are relevant to information management since they handle how information is stored and transmitted. At the time of writing, both of the above APIs are still in early development and not many details are available[6]. Most of the functionalities covered by the above APIs can be traced back to the CDE requirements set by earlier standards such as the ISO 19650, and the DIN

[13]https://github.com/buildingSMART/OpenCDE-API/blob/master/Documentation/20201102.BSI.Summit.Update.p

SPEC 91391. A more detailed discussion on this topic is presented in Section 1.4.

As introduced in the earlier section, ICDD focuses to specifying container structuring, meta-data management and linking of information inside the container. Consequently, the container concepts from it should be technically feasible to implement in as per OpenCDE-API specifications. Presently, the current API is based on OpenAPI (Swagger) specification which describes a REST interface for the CDEs.

Linked Data Platform (LDP)

LDP is a specification containing definitions for reading-writing Linked Data architecture. LDP focuses on the use of HTTP to Create, Read, Update, and Delete Linked Data Resources that are part of a collection. To accomplish the above, this W3C recommendation defines guidelines for interacting with both RDF and non-RDF data. These guidelines are focused on what resource content notations and content serialization formats should be used, how a client can handle changes to resources, and container issues including data and meta-data retrieval using GET and updating containers using POST.

In the LDP specification, containers are considered a very specific web resource, with the capability to respond to HTTP requests relating to the creation and modification operations of resources. Two types of containers are defined, based on how resources are contained, created and deleted: a Basic Container and a Direct/Indirect Container [14].

In the former, links to a 'document resource' (as well as a 'child container') can be defined by using a predefined predicate. In the Direct and Indirect Container, additional relationships such as domain-specific vocabularies can be used for the link relationships, offering more flexibility for the user to define custom resource relationships, beyond LDP definitions.

It should be noted that in the LDP approach, 'anything can be a Container', and these containers can have corresponding RDF links to their resources in three different ways depending on which container is used. However, when this approach is scaled to real building projects, more complex information is usually encoded using specific definitions for links between files/documents, 'sub-document level linkages' etc. As LDP is domain agnostic, it provides the flexibility to define link relationships on any domain-specific vocabulary - the

[14] https://www.w3.org/TR/ldp/\#ldpc

links in a Containers are left to the discretion of the creator. Currently numerous implementations of LDP exists[15].

4.3.3 Implementing ICDD in CDE

With the establishment of the conceptual background of ICDD, OpenCDE and LDP in the previous section, the implementation details of ICDD in both the CDEs are elaborated in this section.

In order to publish an ICDD container in a CDE platform, two aspects need to be addressed: 1) the requirements in terms of ontologies, vocabularies for meta-information management of data inside containers and 2) requirements for the micro-service architecture of the container. To address the former, we present an assessment on the scope of the vocabularies/scheme defined by the existing standards in conjunction to the ones by ICDD. These can include vocabularies for defining meta-information for documents/graphs, specifying base ontologies/schema, version management, file naming conventions, folder structuring and management. A summary of these terms and their corresponding availability in both ICDD-Open CDE API and ICDD – LDP is presented in Table 3.1. The ones with a tick mark (✓) are the functionalities implemented in this paper. As seen in Table 4.3,

To achieve the second aspect, as mentioned in section 2, in this section the functionalities of Table 3.1 are implemented in both OpenCDE-API and LDP. There are three reasons for the selections. First, we base our experiment on the known practices of how to publish linked data using RESTful architecture, for which we believe that LBD contains the necessary specifications. Second, we study the Open CDE API since it is an initiative which focuses on the Building Data domain; thus catering to specific requirements which arise from the AEC industry. It is a good representative of the domain-specific standard for data starting using the micro-service architecture, with its new and developing specification that can be adopted widely in the building data domain. Showing how this can be applied and improved adds knowledge in the field. Third, both specifications are technically detailed, and there are existing implementations. Thus solutions are already established as technically feasible, and it is possible to have a snapshot of the definitions and implement solutions that can be repeated by other groups to validate the results.

Commonly available tools and libraries like React, Node.js, Express etc. are used to add the missing parts of the two technical approaches, do the needed

[15]https://www.w3.org/wiki/LDP_Implementations

Table 4.3. 1 Consolidated Functionalities as specified by ICDD, OpenCDE-API and LDP

Meta-Classification	Requirement/CDE	ICDD - Open CDE API	ICDD - LDP
Container Structuring	Workflow creation of project templates	×	×
	Information Container Classification System	×	✓
	Nested Containers	×	✓
	Customisable folder structure with highly controlled access	×	✓
Versioning & History	Audit trail and version management	✓	✓
	Status Codes for Information in Container	×	×
	Container history log (status change for username, dates, revisions)	×	×
Container Information Mining	Meta-data based information search (query, filter)	×	×
Access Control	Document Control and Document Actions via workflow engine	×	×
	Access control at Information Container level	×	✓
Meta-Information	Customisable meta-data	✓	×
	Asset tagging	✓	✓
	Definitive meta-data classification based on project context (phases)	×	×
File Naming Conventions	Unique ID for each Information Container	✓	✓
	Definitive document naming rules	×	×
Linking Containers	Document Level	✓	✓
	Sub-Document (model element) level	✓	✓
	Link Type specification for Document Level	✓	✓
	Link Type specification for Sub-Document Level	✓	✓
Container CRUD	CRUD	✓	✓
Container States	Archived	×	×
	Work in Progress	×	×
	Shared	×	×
	Published	×	×
Data Integrity	Data conformance tools	×	×

data translations, and orchestrate the experiments. The focus is on the minimal implementation to reveal the differences between the approaches. The implementation decisions are documented. The source code, and the used sample data sets are published at our public GitHub source repository.

Open CDE API

OpenCDE-API for ICDD implements the buildingSMART OpenCDE-API OpenAPI interface specification in the buildingSMART/OpenCDE-API[16] GitHub repository so that the RESTful API can be used to evaluate the ICDD container publication in the server. The code was written in TypeScript for NodeJS and Express and made available in our GitHub repository[17]. The server implements the session creation, file registration, metadata handling, file upload and download, versioning, and version listing. On the other hand, the specification's interactive flow operation has been modified to be tested without user actions to keep the implementation minimalistic.

The buildingSMART International, Ltd. draft for the OpenCDE-API specification can be found under The buildingSMART International Documents API standard[18]. The contained OpenAPI 3.0 document defines the RESTful API endpoints for an OpenCDE-API server. The accompanying sequence diagrams for the document upload and download flow of actions illustrate the interface's use. In this research, the interface and the associated data types define the Minimum Viable Product (MVP) of the concept (Frank, 2016).

In this implementation, the OpenAPI interface specification is applied so that the RESTful API can be used to evaluate the ICDD container publication in the server. The code was written in TypeScript for NodeJS and Express and made available in our GitHub repository. The server implements the session creation, file registration, metadata handling, file upload and download, versioning, and version listing. On the other hand, the specification's interactive flow operation has been modified to be tested without user actions to keep the implementation minimalistic.

The ICDD publication is implemented as an API new interface. The method gets the ICDD input, unzips the content to the document directory of the server, and creates an OpenCDE-API document description for each Internal Document listed in index.rdf in the ICDD container. Furthermore, each document

[16][1] https://github.com/buildingSMART/OpenCDE-API
[17][2] https://github.com/jyrkioraskari/OpenCDE-API_ICDD
[18]https://technical.buildingsmart.org/projects/opencde-api/

node's literal property is mapped into a metadata attribute and saved in the application database, and the OpenCDE-API for is document is created using the metadata.

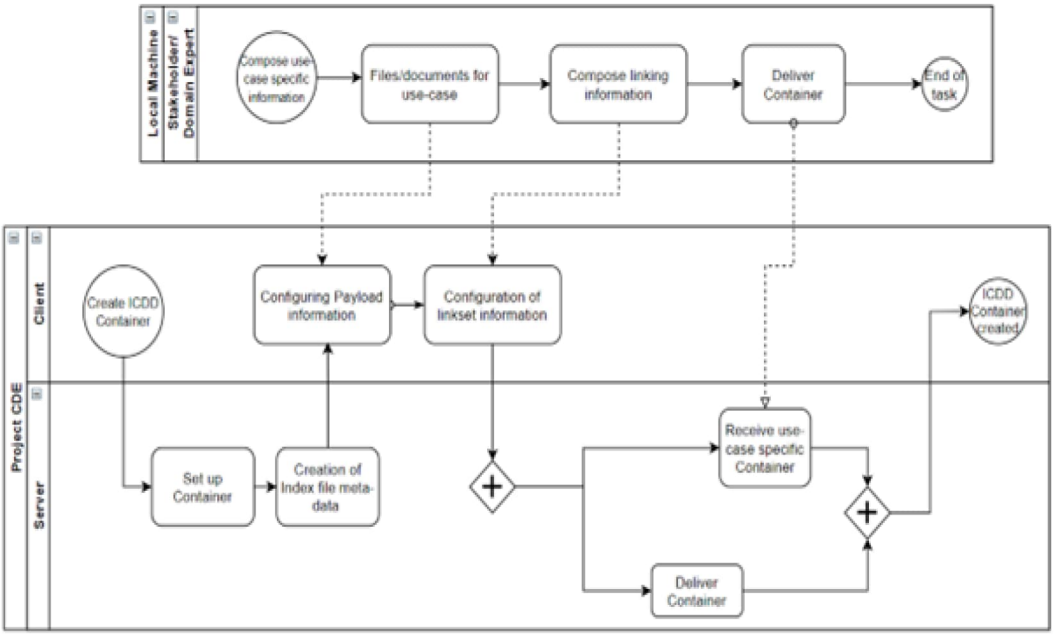


Figure 2 ICDD-OpenCDE-API Interface Usecase

Since the ICDD publication method fills the OpenCDE-API data structures and file conventions, the files and the metadata can be accessed using the same API calls. So, an ICDD container can be published online so that it is accessible through OpenCDE-API.

Linked Data Platform

Existing documentation from W3C defines a list of operations which are must/should/may in order to specify whether a platform conforms to LDP (Speicher and Fernández, 2014). Tables 4.4 and 4.5 document a sub-set of these operations that are tested for implementing ICDD in conjunction with LDP container specifications. Of these, only the ones in a green highlight in the table are tested for implementation in this paper.

In the above table, ICDD container can exist in its native form as a Basic Container where the index file directly contains all the meta-data. However, it can also exist as a Direct or Indirect Container, where the links between documents can be present in other containers (such as nested containers). Figure 3 illustrates the workflow

Table 4.4. Applicable container operations (as specified by W3C) for the ICDD- LDP implementation 1/2

S.No	Meta-Classification for Operations	Operation/Container	Basic	Direct	Indirect
1	MUST	ConformsBcLdpContainer		N/A	N/A
2		ContainerSupportsHttpLinkHeader			
3		MemberResourceTriple	N/A		N/A
4		MemberRelationOrIsMemberOfRelation TripleExists	N/A		
5		RelativeUriResoulutionPost			
6		PostResponseStatusAndLocation			
7		ActAsIfInsertedContentRelationTripleEx ists	N/A		
8		PostResourceUpdatesTriples	N/A		
9		DeleteResourceUpdatesTriples	N/A		N/A
10		PostContainer			
11		ConformsDcLdpContainer	N/A		N/A
12		ConformsIcLdpContainer	N/A	N/A	
13		RequestedInteractionModelCreateNotAll owed			
14		RequestedInteractionModelHeaders			
15		AcceptTurtle			
16		NullRelativeUri			
17		AcceptPostResponseHeader			
18		DeleteRemovesContainmentTriple			
19		PostJsonLd			
20		ConformsContainerRdfResource			
21		RelativeUriResolutionPut			
22		PutReplacedsResource			
23		PutSimpleUpdate			
24		GetResource			
25		RestrictClientInference			
26		GetResourceAcceptTurtle			
27		JsonLdRepresentation			
28		PublishConstraintsReadOnlyProp			
29		PublishConstraintsUnknownPRop			
30		PutReadOnlyProperties4xxStatus			
31		PutPropertiesNotPersisted			
32		ConformsRdfSourceLdpResource			
33		IsHttp11Manual			
34		EtagHeadersGet			
35		ETagHeadersHead			
36		LdpLinkHeader			
37		GetResponseHeaders			
38		ConditionFailedStatusCode			
39		PreconditionRequiredStatusCode			

Table 4.5. Applicable container operations (as specified by W3C) for the ICDD- LDP implementation 2/2

40		PutBadETag			
41		Head			
42		AcceptPatchHeader			
43		Options			
44		OptionsAllowHeader			
45		PostResource	N/A	N/A	
46		ContainerHasInsertedContentRelation	N/A	N/A	
47	**Should**	NoRdfBagSeqorList			
48		PreferContainmentTriples			
49		ContentTypeHeader			
50		PostNoSlug			
51		CreateWithoutConstraints			
52		RestrictUriReUseSlug			
53		RejectPutModifyingContainmentTriples			
54		RestrictPutReUseUri			
55		PatchMethod			
56		ContainsRdfType			
57		ReUseVocabularies			
58		UseStandardVocabularies			
59		GetResourceAsTurtleNoAccept			
60		4xxErrorHasResponseBody			
61		ResponsePropertiesNotPersisted			
62		PutRequiresIfMatch			
63		UseMemberPredicate	N/A		
64		PreferMembershipTriples	N/A		

of the implemented LDP interface for ICDD using a sample use case. As seen from the figure, an authentication functionality is added to the workflow for retrieving containers corresponding to each user. Other parts of the workflow remain the same as the OpenCDE API-ICDD workflow introduced in Figure 2.

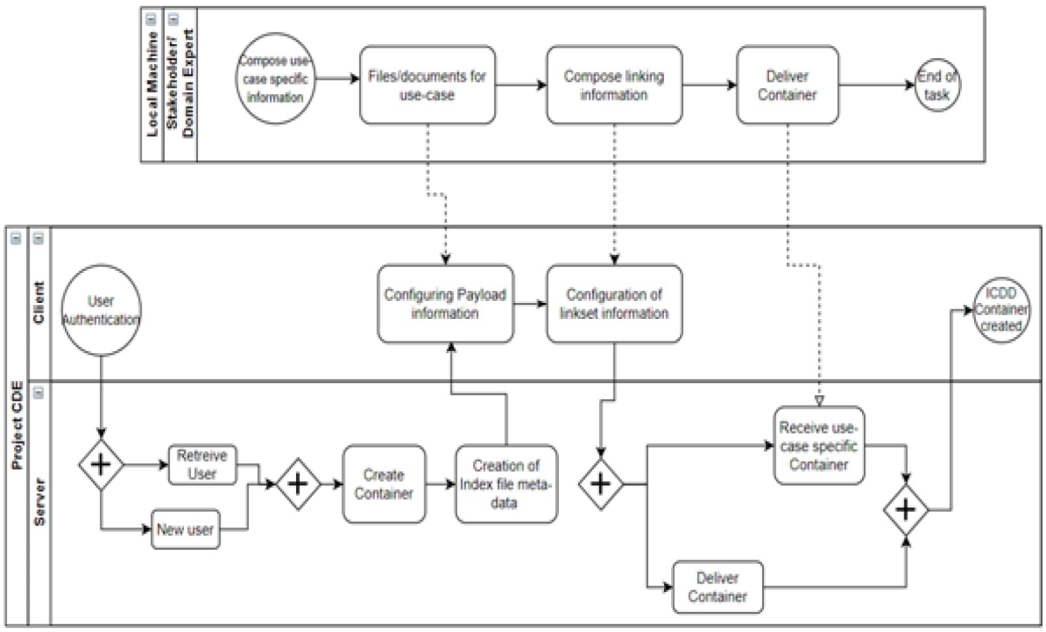


Figure 3 ICDD LDP integration Interface Usecase

4.3.4 Discussions and Conclusion

The design of containers depends significantly on their indented use case. Standards, approaches and recommendations can be used to define the minimum requirements for both the information contained and the API specification. In this paper, we investigated the requirements for the practical aspects of the development of an ICDD container in a CDE based on existing functionalities as specified by the CDEs.

The benefits of publishing an ICDD container using OpenCDE-API and LDP are 1. The containers and files inside them can be accessed online. They have a Uniform Resource Locator (URL) that can be used for interlinking. 2. The versions of the files may be browsed online. 3. The ICDD metadata can be used to query the container files. They can also be queried between containers. While the level of specifications to conform to varies with each CDE, the overall web architecture of these can still be used for implementing ICDD. As seen in Table 3.1, OpenCDE-API does not have explicit specifications for container nesting, meta-classification for containers, access control for documents inside containers, thus necessitating the use of external ontologies and specifications for implementing them. On the other hand, LDP based approach does not contain specifications for operations such as

meta-classification for containers, container history and version management etc.

The implementation demonstrates that an ICDD container and be published as OpenCDE-API so that metadata is automatically read from the container files. This can be extended so that, instead of using the OpenCDE API data model, an RDF graph storage is used for the containers' linked data content. The RDF parsing and the memory RDF database for the document query show that this is feasible.

We presented two REST-ful implementations, based on LDP and OpenCDE-API, compared quantitatively and qualitatively. We use the data validation use case to evaluate the solutions. At present the approaches implemented in this work focus on uploaded files and the future world would include support for data referenceable through triple stores. Additional areas of work would be implementing the features as listed in Table 3.1 by borrowing specifications from ISO 19650, DIN SPEC 91391 etc.

The work presented in this paper is one of the potential ways in which Linked Data supported containers can be represented in CDEs. By combining ICDD's container concepts with the two CDEs, we can leverage ICDD's functionality of accessing linked information at both object and attribute level. This work's main contribution lies in the identification and amalgamation of container concepts from different approaches in order to implement a functional ICDD container in a REST compliant CDE.

4.3.5 Acknowledgement

This research was funded through a doctoral research grant from the Deutscher Akademischer Austauschdienst (DAAD) and partly through BIM4Ren EU project (Grant No. 820773).

4.3.6 References

Bucher, D., Hall, D., 2020. Common Data Environment within the AEC Ecosystem: moving collaborative platforms beyond the open versus closed dichotomy, in: EG-ICE 2020 Proceedings: Workshop on Intelligent Computing in Engineering. Presented at the 27th International Workshop on IntelligentI Computing in Engineering (EG-ICE 2020) (virtual), Universitätsverlag der TU Berlin, pp. 491500. https://doi.org/10.3929/ethz-b-000447240

Frank, R., 2016. A proven methodology to maximize return on risk [WWW Document]. URL https://www.syncdev.com/minimum-viable-product/ (accessed 3.15.21).

Gumpert, S., 2019. BIM360 Coordinate Linking in your BIM360 Design Files for Clashing | Autodesk ANZ Tech Team Blog [WWW Document]. URL https://blogs.autodesk.com/anztechteam/2019/07/22/bim360-coordinate-linking-in-your-bim360-design-files-for-clashing/ (accessed 3.15.21).

Senthilvel, M., Oraskari, J., Beetz, J., 2020. Common Data Environments for the Information Container for linked Document Delivery, in: Proceedings of the 8th Linked Data in Architecture and Construction Workshop. Presented at the 8th Linked Data in Architecture and Construction Workshop, CEUR Workshop Proceedings, Dublin, Ireland (virtually hosted), pp. 132145. http://ceur-ws.org/Vol-2636/

Speicher, S., Fernández, S., 2014. Linked Data Platform Implementation Conformance Report [WWW Document]. Frédéric GRAND. URL https://dvcs.w3.org/hg/ldpwg/raw-file/default/tests/reports/ldp.html (accessed 3.8.21).

Technical Committee: ISO/TC 59/SC 13 Organization and digitization of information about buildings and civil engineering works, including building information modelling (BIM), 2020a. ISO 21597-1:2020 Information container for linked document delivery Exchange Specification - Part 1: Container.

Technical Committee: ISO/TC 59/SC 13 Organization and digitization of information about buildings and civil engineering works, including building information modelling (BIM), 2020b. ISO 21597-2:2020 Information container for linked document delivery Exchange specification Part 2: Link types.

van Nederveen, 2010. Building Information Modelling in the Netherlands: A Status Report, in 18th CIB World Building Congress. Presented at the CIB W78, CIB, The Lowry, Salford Quays, United Kingdom, p. 13.

4.4 Enabling Federated Interoperable Issue Management in a Building and Construction Sector (EG-ICE 2022)

Image a scenario where a facility manager keeps all issue data in a BCF server and then gets an updated set of as-built model data of the associated building packaged as an Information Container for Document Delivery (ICDD) file. The facility manager wants all the given information to be integrated with existing knowledge so that she has a consistent view of the current status of all assets of the supported premise. The new facility management system implements buildingSMART OpenBIM standards. So, she uploads the file by using the buildingSMART OpenCDE Document API. The system automatically connects relevant information with the existing issue descriptions, and when she later accesses the issue descriptions, are given updated information about the building.

Hypothesis 4.2 claims that using Linked Building Data, an ISO 19650 BIM Stage 3 mature Common Data Environment (CDE) can be created.

The selected criteria of ISO 19650 Stage 3 are listed in Section 4.1.2:

1. The system should comply with the buildingSMART openBIM principles: First, the work is based on the BCF standard, and we have suggested adding a compatible BCF API interface. In addition to buildingSMART open standards, we use the W3C Linked Data Platform (LDP). Although it does not belong to buildingSMART standards, it follows the principles.
2. Instead of being file-based, data should be accessible as a database (RDF storage). Data should be federated and searchable: The interpretation is that, in a system, (1) the focus should be on the data. When using Linked Data and when applying bcfOWL, this is fulfilled. Also, in federated queries, data is

aggregated, offering a single source of truth for asset information. (2) The queries over the data fulfil the database approach. A counterargument for the non-file approach claim is that the LDP protocol-based system can also be used file-based (as files could be searched collectively). However, the proposed hierarchy offers one coherent view of the data.

3. Consideration of the ISO 19650 MVD Exchange specifications and data states: these are not covered in work. In [Werbrouck et al., 2022], Werbrouck suggests an approach for data stages, where annotation triples could be used to express the stages. On the other hand, different containers could represent the stages and the associated authorizations.

Table 4.6 summarizes the use of the design science research method in this research.

My contribution to the paper: The work was equally shared between me, Oliver Schulz and Jeroen Werbrouck. My focus was on the Introduction, Discussion and Conclusion parts.

Table 4.6. The work of Oraskari et al. [2022b] in the Hevner et al. [2004] design science research framework

Guideline 1: Design as an artifact
Example construction for the federated BCF data in the Solid platform was implemented.

Guideline 2: Problem relevance

The business problem is that in a longer lifespan, the AEC design offices cooperate in alliances that may decide to use various CDE providers. To make efficient use of cumulated knowledge, a solution is needed. A federated and Linked Data approach is promising from this point of view.

The constituent user community of the artefact is the researchers of the Linked Building Data Community and the software architects of a Common Data Environment. The solution belongs to the Common Data Environment problem class.
This work is an enabler for the Common Data Environment problem class, which explores ways to house construction project information to collect, manage, collaborate, and share information for the artefact organization.

Guideline 3: Design evaluation

Our design was assessed using the use case of Issue Management.
Hevner et al. Design Research Evaluation Method: Analytical Static Analysis.

Guideline 4: Research contributions

Guideline 5: Research rigor

The work was peer-reviewed, and the results are verifiable: our source code is available for inspection.

Guideline 6: Design as a search process

Guideline 7: Communication of research

29th EG-ICE International Workshop on Intelligent Computing in Engineering, Aarhus, Denmark, July 6-8, 2022. The EG-ICE 2022 Best Paper Award.

The work was originally published in [Oraskari et al., 2022b]. The content presented here is based on our final draft of the article.

Jyrki Oraskari Department of Computer Science, Aalto University, Konemiehentie 2, 02150 Espoo, Finland, Jyrki.Oraskari@aalto.fi
Oliver Schulz Faculty of Architecture, Design Computation, RWTH Aachen University, schulz@dc.rwth-aachen.de
Jeroen Werbrouck Faculty of Engineering and Architecture, Ghent University, Faculty of Architecture, Design Computation, RWTH Aachen University, jeroen.werbrouck@ugent.be
Jakob Beetz Department of Design Computation, RWTH Aachen University, Aachen, Germany, j.beetz@caad.arch.rwth-aachen.de

Abstract: A Common Data Environment (CDE) is defined as an agreed-upon source of building-related project information to collect, manage, and exchange data between different stakeholders. The approach in the AEC domain is to use buildingSMARTs BIM Collaboration Format (BCF) as the digital issue communication part of CDEs. Contrasting with the federated nature of the AEC industry, CDEs are typically organised in a centralised fashion. This work proposes a potential transition of BCF into a distributed environment that serves as an example for further developments for the distribution of CDEs and CDE-independent data management. We show how a single source of truth over the project data and the advantages of the central approach can be realised in a distributed setup using a Solid architecture environment, enabling decentralised authentication and stakeholders ability to control their data.

4.4.1 Introduction

In recent years, multiple international standards and specifications have been agreed upon to facilitate the information exchange in the Architecture, Engineering and Construction (AEC) industry. According to ISO 19650, a Common Data Environment (CDE) is defined as an agreed-upon source of information for a building-related project (ISO 19650-1, 2018). It should serve as a single source of truth and is therefore helping to structure the information and data exchange process in the course of a project between the different stakeholders.

The German standard DIN SPEC 91391-1 (2019) extends on the ideas of ISO 19650 and emphasises that a CDE for a building can change during the different phases of its life. Hence, there is a need for the different CDEs to communicate. The specification's authors suggest that standardised RESTful interfaces should be used to ensure the exchange of information containers between different CDEs

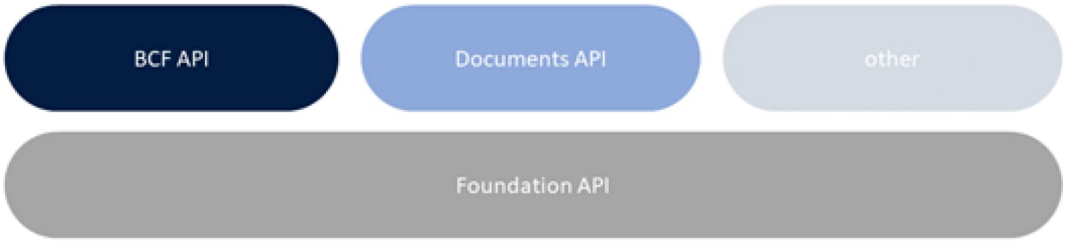


Fig. 4.14. Abstraction of buildingSMART's vision of OpenCDE APIs, originally presented by Yoram Kulbak and Pasi Paasiala in October 20191. The BCF API is part of buildingSMARTs OpenCDE APIs.

at the BIM maturity stage 2.

A standardised approach for digital issue communication in AEC projects is the BIM Collaboration Format (BCF), which is commonly used in combination with CDEs (Preidel et al., 2018). BCF exists in two flavours: a file-based version and a centrally organised, server-based approach (van Berlo and Krijnen, 2014), allowing project managers and clients to study the entire body of issue descriptions via APIs to infer insights into the design team's work. This server-based approach is regarded by buildingSMART as part of the OpenCDE APIs[19] (Figure 4.14).

Communication using BCF enables the user to communicate problems on a component-by-component basis, making BCF an integral part of the planning process. It supplements conventional e-mail communication about issues and can be seen as a central interface for communicating changes in the model. Hence, we focus on this format for this work.

Even though the standards mentioned above describe the exchange of information in detail, implementation remains scarce regarding container-based information exchange or the interconnection of CDEs. While streamlining information exchange is beneficial to the industry as a whole, it is not for commercial CDE providers. As in many sectors, the construction industry has followed the general trend toward Big Data and built data-driven ecosystems that centralise as much data as possible on their central servers. There are many challenges in this model. Data access is regulated via vendor-specific APIs, which often limit query parameters. Duplicated data in the silos can lead to a situation where the users do not have a single source of truth, making it harder to get insights into the project's status. Also, the legal aspects and the responsibilities of the data content, like the General Data Protection Regulation (GDPR) in

[19] OpenCDE-API Documentation: https://github.com/buildingSMART/OpenCDE-API/tree/master/Documentation (Accessed 20.02.2022)

the European Union, are harder to control if the data are not at the hands of their producers. The Solid initiative (Mansour et al., 2016) aims to change the overall course into a model where individual players control the data produced by them and about them. For this to work, these players need a decentralised authentication mechanism which is included in the Solid specification.

This work proposes a potential transition of BCF into a distributed environment that can serve as an example for further developments for the distribution of CDEs and CDE-independent data management. We show how a single source of truth over the project data and the advantages of the central approach can be realised in a distributed environment, which enables decentralised authentication, lower risk of system failure, and stakeholders' ability to control their data.

The paper is structured as follows: the next section introduces the technologies and related works regarded for this paper. Section 3 describes the proposed way to express the federated structure of BCF data on the Solid platform. The paper concludes with a discussion on the proposed framework, including future prospects on this topic.

4.4.2 Background

In this section, we describe the background technologies on which the proposed framework will be based and related research projects. Although this paper bases heavily upon Semantic Web technologies, an elaborate discussion on these topics is outside the scope of this work. For a deeper understanding of technologies such as the Resource Description Framework (RDF) and the SPARQL Protocol and RDF Query Language (SPARQL), the reader is referred to (Hendler et al., 2020).

BIM Collaboration Format and Common Data Environments

In the context of the ongoing establishment of open BIM processes, the need for a communication interface arose in order to be able to transmit information and issues within the models in a software-independent manner. Therefore, the BIM Collaboration Format (BCF) – a buildingSMART standard - was developed to provide a software- and vendor-neutral exchange format for model-based issue communication[20]. Three main parts - *Topics*, *Comments* and *Viewpoints* -

[20]BIM Collaboration Format (BCF) - An Introduction https://technical.buildingsmart.org/standards/bcf/ accessed 28.02.2022

together form the main structure behind BCF and are all connected to a *Project* concept:

1) The *Topic* carries general information about an issue. It uses properties like a current status, a definition of the type of *Topic*, the person who created it and who is responsible for fixing it.
2) *Viewpoints* are used to connect the format to BIM by providing a virtual camera located inside the model that looks at the scene that is part of a current discussion. They can also provide links to specific building elements by stating their GUID in a list.
3) A *Comment* concept provides the textual information and an author in a discussion. It is linked to the *Topic* and can reference a *Viewpoint* as well.

The BCF data can either be exchanged in a file-based format using BCF XML[21] or by using a server (van Berlo and Krijnen, 2014) via a REST API (called BCF API[22]) that returns its information in a JSON format. Even though the data can be serialised in two different ways, the overall concept behind both formats is the same, and the structures of BCF API and BCF XML differ only slightly. For example, BCF Servers - using the BCF API - often allow archiving and downloading the *Project* and its issues as a file in the BCF XML format.

Since the communication in an open BIM process is usually an integral part of many workflows, it is often integrated directly into a Common Data Environment (CDE) (Preidel et al., 2018), which serves as a single source of truth throughout the planning and construction phase. Research regarding the decentralisation of these CDEs can also be observed. In (Werbrouck et al., 2019), the authors suggest a decentralised CDE based on Solid principles, whereas (Tao et al., 2021) describe a CDE distributed via a blockchain. The latter example also combines the principles of a CDE with BCF by distributing them as BCF XML over the blockchain.

bcfOWL

Since the current BCF serialisations lack the general contextual information and shared metamodels that RDF solutions have, the bcfOWL ontology (Schulz et al., 2021) was created to bring together the worlds of issue communication and Linked Data. In the proposed linked data model, BCF issues are stored as RDF triples and can be queried using SPARQL. Additionally, the different concepts for the *Topics* – defined in the BCF *Extensions* – can be enriched with further semantics. Therefore,

[21] BCF XML: https://github.com/BuildingSMART/BCF-XML accessed 24.01.2022
[22] BCF API: https://github.com/buildingSMART/BCF-API accessed 24.01.2022

the *Extensions* in bcfOWL can be used as a gateway to other ontologies in the context of Linked Building Data (LBD).

The ontology is not introducing new concepts to BCF and has semantic interoperability with BCF XML and BCF API. A converter can be created to serialise bcfOWL data into the standard BCF JSON and XML formats. Thus the ontology can serve as a shared foundation for both formats. By using SPARQL to query the data stored as bcfOWL, it is also possible to overcome the accessibility limitations of BCF caused by its hierarchical structure, as described in (Schulz and Beetz, 2021).

Authenticated Data Federation on the Web

Using the Linked Data Platform (LDP) specification[23], the Web of data can be accessed and managed using the read-write operations of the Hypertext Transfer Protocol (HTTP) standard version 1.1, the basis of data communication for the World Wide Web.

LDP incorporates the Linked Data principles (Bizer et al., 2009) of Tim Berners-Lee into a data container architecture. LDP relies on containerisation, where an ldp:Container refers to a specific, dereferenceable RDF graph, listing its resources (ldp:contains). By dereferencing the container URL, it is easy to discover and, in turn, dereference its content (RDF or non-RDF) in a chain of HTTP requests. As container URIs are also RDF resources themselves, nesting containers is possible, resulting in a data organisation system that very much resembles file storage on a computer – yet now file paths are URLs.

While LDP on its own is well-suited for serving *open* data on the Web, it does not specify access-control mechanisms for protected datasets. Where a centralised data store often relies upon local storage of credentials, this is no viable option in a decentral environment. Since a client may combine hundreds of web resources to find what it needs, it is not feasible to maintain an account for each of these sources separately.

Established technologies like OpenID Connect (OIDC)[2] allow outsourcing this part of identity management to specialised identity providers (IDPs) that act as a service in the middle, e.g., Facebook, Google, or GitHub. The Solid initiative (Mansour et al., 2016; Sambra et al., 2016) eliminates the need for a third party: it provides the specifications to create an online identity based on a personal URL (a "WebID"[24]) on a domain chosen by the user. An office can thus become its own IDP

[23] Linked Data Platform: https://www.w3.org/TR/ldp/ accessed 28.01.2022
[24] WebID 1.0: https://www.w3.org/2005/Incubator/webid/spec/identity/ accessed 31.01.2022

and maintain its own credentials for authenticating to decentralised construction management services (Werbrouck et al., 2019). A WebID is associated with a Personal Online Data storage ("Pod"): a data vault based on the LDP specification, but now enabling fine-grained access control to containers and resources defined using the Web Access Control (WAC) [25] ontology. The resources that govern access control in Solid, ACL (Access Control List) resources, use the WAC ontology to link specific access rights to specific WebIDs, agent groups *(vcard:Group)* or (un)authenticated agents (*acl:Agent* or *acl:AuthenticatedAgent*). This way, it can be easily verified by a Solid Pod provider whether an actor can interact with a resource in a specific way: read, append, write or control, i.e., modifying the ACL document itself.

Apart from the ACL resources that govern access rights, the Solid specifications define '.meta' resources: RDF resources that contain metadata statements related to an ldp:Container instance. A .meta resource is served upon dereferencing a container URL. The details of .meta resources are yet to be agreed upon within the Solid community, but by default, they include the LDP containment triples, modification dates etc. Attaching a custom, persistent .meta resource to a Solid container with domain-specific metadata is possible.

LBDserver

A related initiative that uses Solid to store AEC data in a decentral way is the ongoing LBDserver project (Werbrouck et al., 2021). This project proposes data structures to discover project resources, metadata storage and cross-document linking of heterogeneous datasets using the LBDserver vocabulary[26]. Throughout this paper, we will re-use certain patterns proposed in the LBDserver ecosystem (Section 3). However, where the LBDserver proposes a very generic way for data organisation, the BCF specification has a distinct way of structuring BCF-related datasets. As this is standardised within the AEC industry, we will maintain this way of data organisation in this work.

4.4.3 Federated BCF projects

In this section, we sketch the outline for the setup of federated BCF projects. Therefore, we combine domain-agnostic Web specifications from the Solid

[25] Web Access Control, https://solid.github.io/web-access-control-spec/, accessed 31.01.2022

[26] The LBDserver vocabulary, https://w3id.org/lbdserver#, accessed 08/02/2022

ecosystem with domain-specific concepts proposed in the bcfOWL and LBDserver initiatives. Section 3.1 describes an organisational structure for discovering and managing federated BCF data. The project becomes a federated graph in such a setup: the union of contributions stakeholders make on their own "office server", shared using WebIDs. Similarities with the existing BCF API and BCF XML will be drawn where relevant. Based on Solid's existing Web Access Control specifications, an access-control layer is devised upon this organisational structure.

Project Discovery

To make federated BCF projects easily discoverable, we base upon the aggregation structures proposed in the LBDserver. This means that an office can maintain its projects in a root *ldp:Container* on its Pod, i.e., the Project Repository. The URL of this registry is referenced in the office's WebID (*lbds:hasProjectRegistry)*. A project registry has sub-containers for each project the Pod owner participates in ('project access points'). These sub-containers, in turn, have pointers to the contributions ('partial projects') of each stakeholder, including the contribution of the owner of the Pod, which we identify by a </local/> sub-container (Werbrouck et al., 2022). The first type is "virtually contained", the second type is effectively hosted by this container on the Solid Pod. Thus, by dereferencing the project access point, a client discovers a list of federated partial projects. (see Figure 4.15)

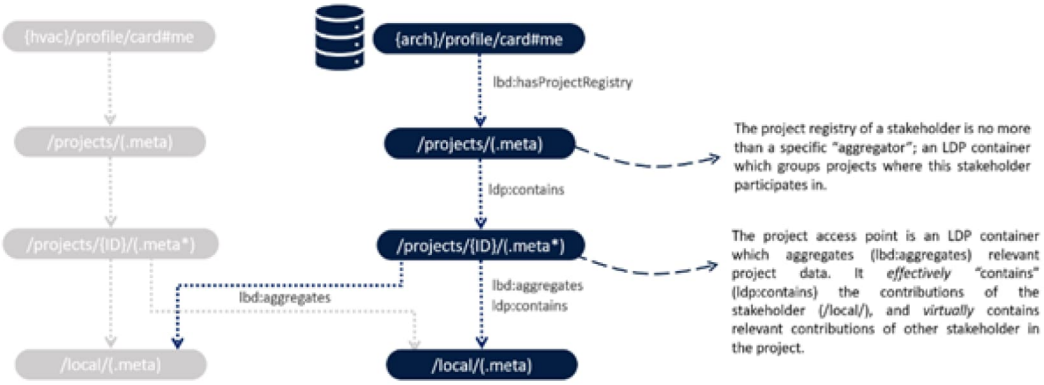


Fig. 4.15. LBDserver patterns for project discovery. The project access point allows finding contributions in the federated project network.

This paper focuses on arranging the BCF data (i.e. *Projects*, *Topics*, *Viewpoints* and *Comments*) inside a Pod, leaving the storage of auxiliary resources (e.g. PDF documents, IFC models) out of scope. In this regard, we suggest a tree-like structure that mimics the routes in the BCF API and the folder structure in BCF XML (Figure 4.16), in contrast to the flattened approach in the LBDServer.

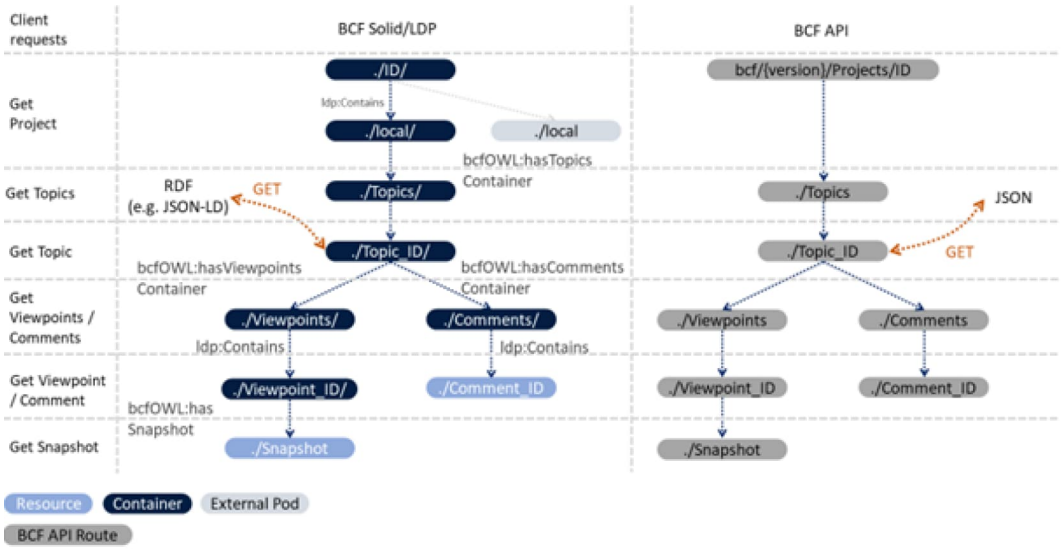


Fig. 4.16. Comparison between the structure of the BCF API and the container-based approach using bcfOWL.

The metadata resource corresponding with the </local/> folder (i.e. at the top level) contains general information about the *Project*, such as its name, what BCF *Extensions* it contains and how they are defined. Furthermore, it links to one or many sub-containers that contain information about the *Topics* belonging to the *Project*, using dedicated sub-properties of "*ldp:contains*". These properties are described in detail in the following section.

Project organisation

Sub-containers containing BCF *Topics* are identified with the RDF predicate "*bcfOWL:hasTopicsContainer*". Each *Topic* is itself located in a sub-container in this container, described with a metafile that contains the *Topics* information in bcfOWL and links to its *Viewpoints* and *Comments* by using the predicates "*bcfOWL:hasViewpointsContainer*" and "*bcfOWL:hasCommentsContainer*". These sub-containers equal in their structure the *Topics* route as defined in the BCF API by including *Viewpoints* and *Comments*. Each *Viewpoint* is a container in itself. The *Comment* is represented as a resource (Figure 4.16). The *Viewpoint*, in turn, contains more concepts, such as a *Snapshot* (e.g. as .png or .jpeg) or a *Perspective Camera*.

If we take a closer look at this hierarchy (Figure 4.16), we can see that there are many similarities to the server routes of the BCF API. Whereas in BCF XML, each

issue is located in its folder (identified by the GUID of the *Topic*). The *Topic* is summarised together with the *Comments* in a Markup file. The BCF API provides all this information as hierarchically organised REST API URI patterns that are accessed sequentially. The BCI APIs sequence is mainly preserved in our proposed structure and allows finding data using standardised endpoints. Apart from the fact that every partial project acts as a "partial BCF server" on its own, which means they need to be queried using multiple HTTP requests, a client will not experience a difference between a Solid-based *Topics* container and a centralised one. This means that a single Pod can be used as a complete BCF server without referencing external partial projects. However, the same infrastructure can be used to federate the project's information.

However, some benefits arise with the RDF-based organisational approach compared with the mere implementation of standardised API routes. Although in an LDP (Solid) environment, similar endpoint patterns to the BCF API are used, these endpoints are also semantically described using the metadata files. Instead of just receiving BCF JSON responses from the server, the server describes what each container entails and what its metafiles depict. The information is easily discoverable in these containers on the Pod. Because resources are stored as files but served in a REST API, it combines the file-based BCF XML, and the service-oriented BCF API approaches.

Furthermore, the links to the sub-containers do not have to be restricted to local resources and can point to any number of other Pods from other stakeholders. Thus, a distributed communication of issues is achieved, in which each participant can store their remarks and additions in their own Pod.

Lastly, this approach allows the dynamic discovery of data. Although the proposed tree structure mimics the BCF API and the BCF XML structure, it is not the only possible configuration. The property-based discovery of containers and sub-containers allows a client to discover how a project is organised. However, this resource could have been stored in a completely different location. An external service may then present this data 'as-if' it is compliant to a specific standard such as BCF.

Access Control and Groups

In a decentral project, each office may maintain access control to the resources they contribute. The WAC ontology supports the use of *acl:AgentGroup*-s, which point to a *vcard:Group* instance, referencing its members (v*card:hasMember*) via their WebID. For each project, an office can publish one or more groups containing

the employees' WebIDs (e.g., 'localEmployees.ttl'). The ACL that governs the resources in the local project folder can then simply grant these groups specific access rights. For instance, the responsible project manager in the office gets *acl:Control* rights and the *acl:Read* and *acl:Write* rights of the other employees working on this project. Agent groups defined by other project stakeholders (i.e., hosted in their own project container) are granted only acl:Read permissions.

Although the above describes a project-specific approach, this works similarly for resource-specific access rights.

Project interaction

This paper does not yet tackle the challenges of a complete workflow, where people create, comment, and update new and existing Issues. When we take the update of an issue as an example, the question must be asked, where this update will be stored in the distributed system. A possible option is that the person who creates the update is storing the new issue - or just the updated content – on their Pod and notifies the original Pod of the issue of its existence. Thereby, when the issue gets queried, it returns a reference to its new version. This option allows tracing the complete history of the *Topic* without ever really deleting or changing existing data. However, it also creates much redundancy that could, in the long run, influence the performance of the queries. Another option that is more in line with the current implementation of the BCF API is to implement a logging system that keeps track of the changes by the users.

4.4.4 Proof of Concept

A proof of concept was created to demonstrate the feasibility of the proposed framework for federated management of BCF data. This includes usage of the Solid Community Server[27], an open-source implementation of the Solid specifications. The data used in this demonstration is based on the BIM models of the DC chair at RWTH Aachen University[28].

To emulate the federated environment, three Pods were set up, containing a total of six issues:

[27] Solid Community Server, https://github.com/solid/community-server, accessed 28.01.2022

[28] Demo dataset: https://github.com/Design-Computation-RWTH/EG-ICE_2022_BCFdemo accessed 28.02.2022

- The Pod owned by the project architect office contains four topics
- The Pod owned by the HVAC engineer contains one topic
- The Pod owned by the structural engineer contains one topic

In this demonstrative scenario, Oliver, working at the engineering office, wants an overview of the current issues registered for the DC chair project. He is registered at the Architect Office's Pod in the list of employees assigned to this project (Listing 1).

Listing 1: the employee group (http://pod.myoffice.org/Projects/8b71315b-7db2-4b35-a1e9-fcddaf8556f5/groups#committed)

```
<#committed> a vcard:Group ;
    dc:created "2022-02-22T22:22:22Z"^^xsd:dateTime ;
    dc:modified "2022-02-22T22:22:22Z"^^xsd:dateTime ;
    vcard:hasMember <http://localhost:3000/oliver/profile/card#me> ,
            <https://localhost:3000/jeroen/profile/card#me>
```

Every stakeholder's </local/> partial projects contain an .acl file that references this group and assigns to acl:Read rights to its members (Listing 2). Of course, editing rights (acl:Write, acl:Append) can be granted in the partial project provided by the architect's office itself.

Listing 2: access rights for employee group in Listing 1

```
<#us>
    a acl:Authorization;
        acl:agentGroup <http://localhost:3000/office/Projects/8b71315b-
7db2-4b35-a1e9-fcddaf8556f5/groups#commited> ;
    acl:accessTo <./>;
    acl:default <./>;
    acl:mode acl:Read, acl:Write, acl:Append .
```

First, the available partial projects need to be discovered. This is covered by the SPARQL query:

SELECT ?partial WHERE {<> lbds:aggregates ?partial}

As indicated in Figure 4.16, the organisation of partial projects corresponds with the BCF API specification data patterns. Because these are standardised routes,

further discovery is not necessary, and the Web client can send the following (authenticated) requests to each of the partial projects:

GET {partial project}/topics

This will yield an LDP container with pointers to the contained *Topics*, which can now be easily retrieved and presented in a GUI. This series of HTTP requests is not identical to those required to access the BCF API. However, they can be easily implemented under the hood by a BCF server to expose federated information to conform to the standard. In that case, the server acts as a middleware to regulate access to information on the stakeholder Pods.

4.4.5 Discussion and Conclusion

The translation of BCF into a Solid environment introduces its main concepts in a federated setup, using the Solid specifications and the Linked Data Platform. This way, issue communication can be spread over multiple Pods that belong to different stakeholders. *Project* stakeholders are no longer solely identified by their e-mail addresses, as implied by the BCF user concept. Instead, they use of semantically rich WebIDs for offices and employees allows the dynamic creation of user groups (e.g. defined by role or participation) and the re-use of credentials in multiple federated *Projects*. A project team from an architectural stakeholder can define its members in their Pod and manage who has access to the *Project* data and who does not. The resulting group (vcard:Group) is then linked to BCF *Project* data to control access. Hence, it becomes possible to define more granular access rights for the different roles on the *Project* pod. This corresponds to the structures in the federated construction industry.

The decentral authentication mechanism, as defined in the Solid specifications, does not require the server to store the login credentials for every user. The service is just responsible for verifying if the access rights associated with a given WebID allow a user to interact with a specific resource in a specific way. The need for the user to create an account for every service is thereby removed by using WebIDs.

Future developments in this area include an investigation of guaranteed availability of the federated data, so that loss of project information (either accidentially or intentially) is prevented. Furthermore, a notification system between Pods in the network will enable automated synchronisation (e.g. when someone updates a Topic or creates a new Comment).

In this paper, we proposed a tree-like structure that combines the traits of BCF

XML and the BCF API. The URL routes of the BCF API thereby serve as a means to access the individual containers. The information in these containers is stored in a .meta file. The different containers holding the BCF data are linked via bcfOWLs properties. We have shown that BCF projects information can be queried using a one-stop access point implemented using distributed Solid architecture and offered as a single source of truth. In the platform, authentication can be self-hosted, and the stakeholders can manage their data using a tree-like structure that mimics the hierarchical pattern of the BCF API endpoints. It was also shown that the data authorisation enforcement can be set using the container architecture of the Solid specification.

Since BCF is a member of the OpenCDE API family, we hope this research can serve as a template for further research on aligning AEC standards with future-proof concepts for federation on the Web.

4.4.6 Acknowledgements

This research is funded by the Research Foundation Flanders (FWO) as a Strategic Basic Research grant (grant no. 1S99020N) and by the EU through the H2020 project BIM4REN grant (grant no. 820773).

4.4.7 References

Bizer, C., Heath, T., Berners-Lee, T., 2009. Linked Data - The Story So Far. Int. J. Semantic Web Inf. Syst. 5, 122. https://doi.org/10.4018/jswis.2009081901

DIN SPEC 91391-1 Common Data Environments (CDE) for BIM projects - Function sets and open data exchange between platforms of different vendors - Part 1: Components and function sets of a CDE; with digital attachment, 2019.

Hendler, J., Gandon, F., Allemang, D., 2020. Semantic Web for the Working Ontologist: Effective Modeling for Linked Data, RDFS, and OWL. Morgan & Claypool.

ISO 19650-1 Organization and digitization of information about buildings and civil engineering works, including building information modelling (BIM) - Information management using building information modelling - Part 1: Concepts and principles, 2018.

Mansour, E., Sambra, A.V., Hawke, S., Zereba, M., Capadisli, S., Ghanem, A., Aboulnaga, A., Berners-Lee, T., 2016. A Demonstration of the Solid Platform for Social Web Applications, in: Proceedings of the 25th International Conference Companion on World Wide Web, WWW 16 Companion. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, pp. 223226. https://doi.org/10.1145/2872518.2890529

Preidel, C., Borrmann, A., Mattern, H., König, M., Schapke, S.-E., 2018. Common Data Environment, in: Borrmann, A., König, M., Koch, C., Beetz, J. (Eds.), Building Information Modeling: Technology Foundations and Industry Practice. Springer International Publishing, Cham, pp. 279291. https://doi.org/10.1007/978-3-319-92862-3_15

Sambra, A., Mansour, E., Hawke, S., Zereba, M., Greco, N., Ghanem, A., Zagidulin, D., Aboulnaga, A., Berners-Lee, T., 2016. Solid: A Platform for Decentralized Social Applications Based on Linked Data.

Schulz, O., Beetz, J., 2021. Image-documentation of existing buildings using a server-based BIM Collaboration Format workflow., in: EG-ICE 2021 Workshop on Intelligent Computing in Engineering. Presented at the EG-ICE, Berlin.

Schulz, O., Oraskari, J., Beetz, J., 2021. bcfOWL: A BIM collaboration ontology, in: Proceedings of the 9th Linked Data in Architecture and Construction Workshop. Presented at the LDAC2021, Luxembourg.

Tao, X., Das, M., Liu, Y., Cheng, J.C.P., 2021. Distributed common data environment using blockchain and Interplanetary File System for secure BIM-based collaborative design. Autom. Constr. 130, 103851. https://doi.org/10.1016/j.autcon.2021.103851

van Berlo, L., Krijnen, T., 2014. Using the BIM Collaboration Format in a Server Based Workflow. Procedia Environ. Sci., 12th International Conference on Design and Decision Support Systems in Architecture and Urban Planning, DDSS 2014 22, 325332. https://doi.org/10.1016/j.proenv.2014.11.031

Werbrouck, J., Pauwels, P., Beetz, J., Mannens, E., 2021. Data Patterns for the Organisation of Federated Linked Building Data, in: Proceedings of the 9th Linked Data in Architecture and Construction Workshop. Presented at the LDAC2021, Luxembourg.

Werbrouck, J., Pauwels, P., Beetz, J., van Berlo, L., 2019. Towards a Decentralised Common Data Environment using Linked Building Data and the Solid Ecosystem.

Werbrouck, J., Pauwels, P., Beetz, J., Mannens, E., 2022. LBDserver - a Federated Ecosystem for Heterogeneous Linked Building Data. (Under review).

4.5 A Generic Framework for Federated CDEs applied to Issue Management (AEI 2023)

Image a scenario where a facility manager keeps all issue data in a BCF server and then gets an updated set of as-built model data of the associated building packaged as an Information Container for Document Delivery (ICDD) file. The facility manager wants all the given information to be integrated with existing knowledge so that she has a consistent view of the current status of all assets of the supported premise. The new facility management system implements buildingSMART OpenBIM standards. So, she uploads the file by using the buildingSMART OpenCDE Document API. The system automatically connects relevant information with the existing issue descriptions, and when she later accesses the issue descriptions, are given updated information about the building.

Hypothesis 4.2 is shared with Section 4.5 as the work is extension of the paper.

Table 4.7 summarizes the use of the design science research method in this research.

My contribution to the paper: The work was shared between me, Oliver Schulz, and Jeroen Werbrouck. In the research work, I organized us to fill out Ben Ellways Academic Toolkit, Article Design Canvas[29], focused on the iterated work on clarifying the research questions, the hypotheses, the consistent story, and writing the first version of the conclusions. I summarized the differences to the earlier conference paper [Oraskari et al., 2022b]. I also took part in writing the Related Work section about federations, related initiatives, and software architectures.

Table 4.7. The work of Werbrouck et al. in the Hevner et al. [2004] design science research framework

[29] https://www.researchgate.net/project/Research-Design-Canvas-and-Article-Design-Canvas

Guideline 1: Design as an artifact

Example construction for BCF data was implemented and federated queries.

Guideline 2: Problem relevance

The business problem is that, in the longer term, AEC design firms may be involved in collaborative alliances that use different CDE vendors. A solution is needed to make efficient use of the accumulated knowledge. A federated and linked data approach is promising from this perspective.

The constituent user community of the artefact is the researchers of the Linked Building Data community and the software architects of a Common Data Environment. The solution belongs to the Common Data Environment problem class.
This work is an enabler for the Common Data Environment problem class, which explores ways to house construction project information to collect, manage, collaborate and share information for the organisation of the artefact.

Guideline 3: Design evaluation

Our design was assessed using the use case of Is- sue Management. *Hevner et al. Design Research Evaluation Method:* Analytical Static Analysis.

Guideline 4: Research contributions

The paper illustrated how managing interoperable building data in a federated environment could be achieved using existing open industry standards.
The main contribution of this research is a generic, federated framework for project-related, interdisciplinary collaboration for CDEs.

Guideline 5: Research rigor

The work was peer reviewed and the results are verifiable: our source code available for inspection.

Guideline 6: Design as a search process

We listed requirements for a desirable solution and have proposed a federated framework that reflects them.

Guideline 7: Communication of research

Accepted 7 augustus 2023. Advanced Engineering Informatics, 2023.

The content presented here is based on our final draft of the article published originally in [Werbrouck et al., 2023]. The content presented here is based on our final draft of the article.

Jeroen Werbrouck Faculty of Engineering and Architecture, Ghent University, Faculty of Architecture, Design Computation, RWTH Aachen University, jeroen.werbrouck@ugent.be
Oliver Schulz Faculty of Architecture, Design Computation, RWTH Aachen University, schulz@dc.rwth-aachen.de
Jyrki Oraskari Faculty of Architecture, Design Computation, RWTH Aachen University, Jyrki.Oraskari@dc.rwth-aachen.de
Erik Mannens Faculty of Engineering and Architecture, Ghent University
Pieter Pauwels Faculty of Engineering and Architecture, Ghent University, Faculty of the Built Environment, TU Eindhoven
Jakob Beetz Department of Design Computation, RWTH Aachen University, Aachen, Germany, j.beetz@caad.arch.rwth-aachen.de

Abstract: This paper analyses the requirements for managing interoperable building data in a federated Common Data Environment (CDE). We discuss the need for generic (meta)data storage patterns, semantic query interfaces, decentral authentication, data aggregation, and adaptation and prove that their combination is feasible with current-day technologies. We illustrate the mechanisms of such federated CDE by considering the topic of digital Issue Management, one of the primary functions of a CDE. In an exemplary data flow process, we show how generic (federated, Semantic Web-based) data patterns for Issue Management can be aggregated and restructured to match existing industry standards like buildingSMARTs BIM Collaboration Format (BCF) API. Finally, we show the methodology is compatible with current-day practice by implementing this process in a proof of concept. The main contribution of this research is a generic, federated framework for project-related, interdisciplinary collaboration for CDEs.

keywords: Common Data Environment, Federation, Issue Management, BIM Collaboration Format, Solid

4.5.1 Introduction

Similar to almost every other economic sector, the Architecture, Engineering, Construction and Operations (AECO) industry has known significant digitalization efforts over the past decades. However, this adoption is taking place at a slower rate than most other industries [Agarwal et al., 2016]. Although

it shares some challenges with other domains, such as product manufacturing and heritage management, the AECO industry is unique in that it must integrate multiple of these complexities simultaneously. For example, it is one of the most decentralised industries around [Box, Inc., 2014]: every project has multiple small- and medium-sized offices contributing to it with their intellectual properties, and every office may work on multiple projects simultaneously. Hence, the industry can be described as a double-layered patchwork with a project- and stakeholder-centric layer. Furthermore, most of its end products will be unique, so the potential to set up standardised workflows or production chains is limited. Finally, built assets require high maintenance efforts long past their handover, which requires data retention policies spanning several decades, likely involving multiple handovers between stakeholders. However, this need for data to be long-term available also means that there are many opportunities for interdisciplinary information reuse. For example, for registration of damage records, managing access and integration with infrastructure, or for AEC-external domains such as history and heritage management and room adaptation based on medical requirements or personal preferences. However, a highly interdisciplinary data management strategy is required to allow such extensibility.

In recent years, multiple international standards and specifications have been agreed upon to facilitate collect, manage, and exchange data between these stakeholders. According to ISO 19650 [International Organization for Standardization, 2018b], a Common Data Environment (CDE) is defined as a Single Source of Information (SSoI) about any built asset. The data is collected in a managed workflow in named and persistent datasets, i.e., information containers. Current CDE platforms primarily contribute to the project-centric layer, offering services to synchronise and interact with project data and data storage space through API interfaces. In this way, data duplication is avoided, and conflicts can often be identified before they cause harm.

Contrasting with the federated nature of the industry, current-day CDEs are typically organised as centralised, commercial platforms developed by companies that provide common BIM (Building Information Modelling) authoring tools (e.g., Autodesk BIM360, Graphisoft BIMcloud, Trimble Connect, and Nemetschek BIMPLUS). While this enables tight integration with the other services in a vendors ecosystem, it also entails the risk of being caught in a vendor lock-in. This is related to the providers business model, guiding them to favour proprietary data formats (Closed BIM) over open exchange formats (Open BIM), which enables a more streamlined data exchange between services within the ecosystem. However, it generally complicates interoperability, long-term archival and compatibility with external services, which may be developed on top of a different (open or

proprietary) data model [Borrmann et al., 2018].

When building data is to be reused extensively throughout the asset's life cycle, the discovery of related information and the flow of data between disparate CDEs and external services needs to be as effortless as possible. The German standard DIN SPEC 91391 [Deutsches Institut für Normung, 2019a] extends on the ideas of ISO 19650 and emphasises that the CDE for a building can change during the phases of that building's life. Moreover, information may be dispersed over multiple CDEs even in a single phase of the building life cycle - as stated in the DIN SPEC 91391 - especially in the case of large or complex assets or widely dispersed teams. Hence, a communication protocol between CDEs from different vendors is necessary. The specification's authors suggest that standardised RESTful (REpresentational State Transfer) [Fielding, 2000] application programming interfaces (APIs) should be used to ensure the exchange of information between different CDEs. However, even with a standardised RESTful API, challenges regarding account management, data ownership and data reusability persist. As mentioned, an AECO office or a contractor can be part of multiple projects. At least in theory, for every project, a different CDE provider can be chosen. The situation occurs that their employees not only need to master and manage subscriptions for different environments for different projects but also that their intellectual property gets fragmented on the servers of multiple commercial platforms. This is not only a theoretical issue but an actual problem that can lead to legal disputes [HH JUDGE EYRE QC, 2021].

In this paper, we balance the benefits and barriers of a *federated* CDE from the perspective of the building sector in terms of interoperability and data reusability. In this approach, data is managed by the entity that created it, and connected to other data on the Web, hence becoming part of a larger, federated project. We discuss the requirements and technological basis for such an ecosystem to address the above-mentioned challenges and incorporate both the project-centric and the stakeholder-centric layers of the patchwork. One of the core functionalities of a CDE is Issue Management, which is currently only possible in centralised environments. Because Issue Management requires intensive interactions between project partners, it is suited to illustrate the feasibility of interactions in a federated environment. Hence, we will use Issue Management to demonstrate the organisational patterns of the ecosystem using existing W3C standards. Furthermore, we will illustrate the compatibility of the ecosystem with the existing industry standard BCF (BIM Collaboration Format) [van Berlo and Krijnen, 2014] using its Web-based flavour BCF API. The implementation of the BCF API is often based on centralised data storage patterns. However, the specification does not enforce this.

We can identify the following hypotheses, which will be analysed in the paper:

Hypothesis 4.3 *Decentralised data management reflects the nature of a fragmented industry, and better guarantees ownership of data compared to centralised solutions.*

Hypothesis 4.4 *Considering the entire life cycle of a digital built asset, a federated approach can reinforce the single source of information compared with common centralised solutions.*

Hypothesis 4.5 *A reusable single source of information is discipline-agnostic at its core. Separation of functionality and storage allows to re-organise data in a multitude of ways, providing compatibility with not one but many specific purposes in current AECO practice. Consequently, such layered system allows parties with lower maturity levels to communicate with more digitally mature partners.*

After an introduction of related work and some background technologies that will be used further in the paper (Section 4.5.2), we take a technology-independent approach in Section 4.5.3, outlining the requirements for the identified ecosystem to function as intended. Then, we show that this is achievable with current-day technologies (Section 4.5.5), in a combination of domain-agnostic and -specific technologies and standards (Section 4.5.4). We illustrate the design of the ecosystem using Issue Management as an exemplary part of a CDE in Section 4.5.5, and indicate the ecosystem's general applicability beyond Issue Management by outlining how other aspects of a CDE may follow the same architectural principles. The advantages and disadvantages of the approach as well as future work will be discussed in Section 4.5.6. We conclude the paper in Section 4.5.7.

This paper builds on previous research presented at the conference paper of [Oraskari et al., 2022b]. In [Oraskari et al., 2022b], an alignment was created between the API endpoints of the BCF API and the structuring patterns of the Linked Data Platform [Speicher et al., 2015] as implemented in the Solid ecosystem [Mansour et al., 2016]. However, in this paper, functionality and storage are decoupled, resulting in a more abstract service architecture that may not only be applied to the BCF API but also to other (custom) functionalities of CDEs. Therefore, the main contributions of this paper include an analysis of the requirements for time-lasting and technology-independent organisational patterns of a federated CDE and a mapping of these principles to current-day technologies. To achieve this, we use selected hypotheses and a requirement analysis of a federated CDE. A high-level overview of technologies for a federated CDE is given by Werbrouck et al. [Werbrouck et al., 2019], which bases upon the specifications of the Solid project [Mansour et al., 2016]. In such a setup, data remains under the original creator's control, connected to other project data through Web technologies and accessible using a single Web

identity. This was extended to a domain-agnostic data federation ecosystem in [Werbrouck et al., 2023]. However, a concrete connection with current-day AECO industry standards and CDE functionalities remained absent until now.

```
@prefix bcfowl: <http://lbd.arch.rwth-aachen.de/bcfOWL#> .
@prefix dcat: <http://www.w3.org/ns/dcat#> .
@prefix consolid: <https://w3id.org/consolid#> .
@prefix dcterms: <http://purl.org/dc/terms/> .
@prefix void: <http://rdfs.org/ns/void#> .
```

Listing 4.1. Prefixes used in the paper.

4.5.2 Related Work

Federation and Linked Data

ISO 19650 defines a federated model as a composition of information containers delivered by a task team around a project or an asset. In this case, federation means that there may exist multiple disciplinary 'partial' models, which can be spatially integrated into one 'coordination model' [Preidel et al., 2018]. Federated models in the context of the BIM Maturity Ramp of the UK BIM Task Group refer to an approach where the sub-models are loosely coupled [Borrmann et al., 2018]. However, the notion of discipline-specific sub-models is not directly related to the concept of knowledge federation in a Web context: using the ISO 19650 definition, a centralised CDE can host a federated coordination model (as the union of partial models). Knowledge federation in the Web context relates to the aggregation and harmonisation of information from different datasets on the Web.

Parallel with developments towards more streamlined data management in the AECO sector, a global tendency exists towards publishing data according to the FAIR principles (Findable, Accessible, Interoperable and Reusable) [Wilkinson et al., 2016]. For example, in 2018, the European Commission outlined an action plan towards FAIR data in the EU [Commission et al., 2018]. Although the FAIR principles are often associated with Open (Science) Data, they do not require data to be publicly available - as access restrictions are often necessary.

The Semantic Web [Berners-Lee et al., 2001] technology stack is often identified as one of the few that fulfils the FAIR requirements, although its usage is not a prerequisite [Mons et al., 2017]. In 2001, Tim Berners-Lee defined the Semantic Web [Berners-Lee et al., 2001], including many technologies of the current Semantic Web technology stack. The term Linked Data was only coined in

2006 [Berners-Lee, 2006], explicitly considering the publication of data on the Semantic Web and the goal of linking it to outside data. This resulted in a 5-star deployment scheme for open data: the more stars, the higher the potential to link to other data, with the 5th star eventually forming '*data in the Web, linked to other data*' [Berners-Lee, 2012], a federated knowledge graph. The cornerstone of this knowledge graph is the Resource Description Framework (RDF) [Klyne et al., 2014], which relates atomic statements in the form of basic sentences or triples. Using HTTP URIs (Uniform Resource Identifiers), a Web-wide context is used, making the federation of interconnected data possible. The SPARQL Protocol And RDF Query Language (SPARQL) [Harris et al., 2013] is the international standard for querying data structured in RDF. For an in-depth introduction to RDF, SPARQL and other Semantic Web technologies, the reader is referred to [Allemang et al., 2020]. For an overview of the challenges that lie ahead in this research domain, the reader is referred to [Verborgh and Vander Sande, 2020].

As for the AECO sector, the potential of Semantic Web technologies has been documented multiple times, in terms of logical inferencing and proofs [Pauwels et al., 2017], interdisciplinary collaboration [Pauwels et al., 2017, Curry et al., 2013, Corry et al., 2014] and interoperability on data level [Beetz, 2009]. Multiple domain-specific data schemas (ontologies) have been developed throughout the years, ranging from the Semantic Web representation of IFC (ifcOWL [Pauwels and Terkaj, 2016]) to the more modular ontologies developed by members of the W3C Linked Building Data (LBD) Community Group [Rasmussen et al., 2021]. Moreover, the use of Semantic Web technologies allows the creation of a federated knowledge graph, where information on one server can be semantically related to information on another one, contrasting with project information management on one centralised CDE. In this context, a federated CDE thus allows for the creation of partial models in a decentral fashion: stakeholders manage their work on a server of their choice but can refer on a granular level to information on other servers. A prerequisite for this is the usage of commonly agreed standards and protocols. Using client-side federated queries (i.e., joining information from different datasets), a federated SSoI can be created, querying subsets of the Web for project-specific and contextual information for a particular purpose. In this paper, we use the term federation to discuss such alignment: federated management (computer science) of federated partial models (AECO).

This paper uses the Turtle [Becket et al., 2014] syntax for RDF graphs, using the prefixes in Listing 4.1. The same prefixes are used for SPARQL queries throughout the paper.

Identity Management in a Federated Context In its early days, Linked Data was commonly associated with open data, i.e. data that is reusable by anyone for any purpose. By following links in the cloud of Linked Open Data (LOD), information can be aggregated and combined for a specific scenario. However, there are many situations where data is not openly accessible, and only authorised agents can interact with this data. Technologies that allow separation of identity and data access - a prerequisite for authentication in a federated context - have been standardised in recent years, such as OpenID Connect (OIDC) [Sakimura et al., 2014], which has known wide adoption. Supporting OIDC, a service can outsource the authentication protocols to specific identity providers (IDP), often big tech companies. In the Solid Project [Mansour et al., 2016], this dependency on large companies is resolved: Solid is a set of Web standards and protocols of which the key idea is to enable individuals to manage their data. The specifications described in the Solid Protocol [Capadisli et al., 2022] extend the OIDC protocol with the notion of a dereferenceable WebID: a URL that acts as a username on the Web, hosted by an identity provider of choice. Consequently, this means that AECO offices may as well choose to self-host their IDP and store their produced intellectual property without the need for an external CDE provider [Werbrouck et al., 2019]. Essentially, Solid adds an authentication layer to the existing Linked Data Platform (LDP) [Speicher et al., 2015] which allows read/write interactions with heterogeneous (i.e. RDF and non-RDF) resources through basic HTTP requests. Identified by their WebID, authenticated agents can be granted certain access rights based on the Web Access Control (WAC) [Solid, 2022] vocabulary. A WebID is linked to one or more Solid Pods, i.e. personal data vaults that implement the Solid protocol.

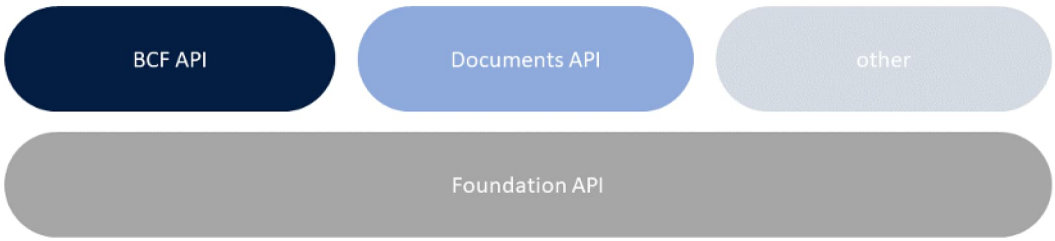


Fig. 4.17. Abstraction of buildingSMART's OpenCDE APIs, originally presented by Kulbak and Paasiala in October 2019. The BCF API is part of buildingSMARTs OpenCDE APIs.

Image source: [Oraskari et al., 2022b])

Image source for Image 4.17 is [Oraskari et al., 2022b].

Common Data Environments

Common Data Environments in the AECO industry are agreed-upon platforms that serve as a Single Source of Information about any asset. They are used to exchange plans, (BIM) models and communication and provide tools for cloud-based collaboration, both in multi-disciplinary context [Sacks et al., 2022] as for domain-specific purposes such as HVAC design [Seidenschnur et al., 2022]. ISO 19650 defines workflows for CDEs while not going too much into the specifics of what components define a CDE. On the other hand, the DIN SPEC 91391 [Deutsches Institut für Normung, 2019a] provides a detailed description of the requirements and functionalities needed in a CDE. Next to the requirements, the DIN SPEC features a CDE API in its second part [Deutsches Institut für Normung, 2019b], which serves as a suggestion for standardised communication with a CDE. It is mainly focused on accessing information containers in a CDE via an API.

In contrast to the DIN SPEC 91391 proposal, the buildingSMART OpenCDE proposal does not consist only of a single interface but an aggregated set of APIs that together form the OpenCDE APIs. The current set of APIs is the OpenCDE Foundations API[30], serving a common authentication approach for all the following interfaces, the BCF API[31] for communicating `Issues`, and the Documents API[32], which manages the interaction with CDEs and authoring tools on a file level. An abstraction of buildingSMART's OpenCDE APIs, originally presented by Kulbak and Paasiala in October 2019, is illustrated in Figure 4.17. Other APIs to follow the current set are envisioned by buildingSMART [buildingSMART International Ltd., 2020].

The core part of a CDE can be divided into two concepts. In terms of infrastructure, a CDE relies on an access-controlled data storage system to host the data generated and gathered over time while multiple people work on one common project. This data can include, for example, plans, (BIM) models and communication such as e-mails. In functional terms, we have the features described in the DIN SPEC 91391, including messaging, model viewers, approval processes, etc. The CDE standards do not describe whether a CDE has to be centralised or federated as long as the functionalities are provided and the data is available. Nevertheless, CDE providers nowadays mainly provide a centralised point of access for CDEs where all information is accumulated. Since the well-established CDEs are proprietary, we cannot state with certainty if the back-end infrastructure is federated or centralised. However, when a single company is

[30] OpenCDE Foundations API: https://github.com/buildingSMART/foundation-API (Accessed: 17.11.2022)

[31] buildingSMARTs BCF API: https://github.com/buildingSMART/BCF-API (Accessed 17.11.2022)

[32] buildingSMARTs Documents API: https://github.com/buildingSMART/documents-API (Accessed 17.11.2022)

controlling access, it makes the CDEs behave like a centralised platform.

Preidel et al. [Preidel et al., 2018] provide an overview of CDEs in the AECO industry by referencing standards and products, claiming that '*the centralisation of data storage within the CDE reduces the risk of data redundancy and ensures the availability of up-to-date data at any time.*' While DIN SPEC 91391-1 makes a similar claim, ISO 19650 does not, which leaves room for federated CDE workflows. Several research approaches deviate from the idea of a central data vault. For example, in Werbrouck et al. [Werbrouck et al., 2019], the authors suggest a decentralised CDE based on the Solid ecosystem, and Tao et al. [Tao et al., 2021] describe a CDE distributed via a blockchain.

BIM Levels and Stages The BIM maturity levels and stages are commonly associated with the concept of CDEs. The levels were introduced in the PAS 1192 [BSI, 2013] and describe the evolution from 2D CAD plans (Level 0) towards 3D models (Level 1) and the concept of BIM (Level 2) and the standardisation of BIM (Level 3), e.g., via IFC (Industry Foundation Classes) - the main international open standard for BIM by buildingSMART (ISO 16739-1:2018, [International Organization for Standardization, 2018b]). PAS 1192 is nowadays superseded by ISO 19650, which replaces the levels introduced in the PAS 1192 and describes the maturity of information management in the form of maturity stages, each stage consisting of different layers. While ISO 19650 describes all three stages, the standard mainly focuses on Stage 2 while having minor overlaps with Stage 1 and Stage 3. The standard highlights that Stage 3 is still under development and is not yet wholly achieved [International Organization for Standardization, 2018b]. A summary of ISO 19650 and a scenario for Stage 3 can be found in [Godager et al., 2021, 2022]. DIN SPEC 91391 is based on the concept of PAS 1192 as well as ISO 19650 and references both, although the related concepts of levels and stages are mixed. While the DIN SPEC mainly focuses on level/stage 2, it provides further insight into the different levels. It describes CDE Level 2 as container-based, where the metadata can be used to query the containers and their possible sub-containers, while CDE Level 3 is described as database-oriented, with query language access to interact with the individual components of the containers. Developments in the Linked Building Data Community, which among others, designs standard, machine-readable vocabularies (ontologies) for concepts in the AECO Industry (Section 4.5.2), are generally seen as a step towards BIM Level 3. The description of Level 3 as database-oriented raises the question of a common query language (QL). Approaches towards a common (domain-specific) query language for BIM can be seen in BIMQL [Mazairac and Beetz, 2013], and BIMSPARQL [Zhang et al., 2018], which focus on making the IFC content queryable. However, linking data with adjacent domains will require compatibility with common domain-agnostic QLs - such as SPARQL, SQL, or GraphQL.

Single Source of Information In Section 4.5.1, it is claimed that a federated approach reinforces the Single Source of Information (SSoI) in comparison with CDE solutions that are based upon a single access point for the data (Hypothesis 4.4). When taking a closed BIM approach, the boundaries of the SSoI are well-defined and can thus be supported by a specific CDE and its associated toolset. In a big open BIM situation, however, the main idea is the reusability of data for unforeseen purposes during the life cycle of the asset, which aligns with the FAIR principles mentioned in Section 4.5.1. In the most general sense, the SSoI is then a combination of project data and contextual data (e.g., users, location or government), all federated on the Web. The providers for such contextual information will publish their data independently from each other and from the project (hosted by the central CDE). Whenever either a synchronised copy of such contextual data is created in the central CDE, or hyperlink pointers are established, the project essentially becomes federated, but its centre of gravity still lies in the central CDE, which forms the main access point for discovering and using project data. However, when project data resides in multiple CDEs, this hierarchy is less clear: multiple access points exist, but the SSoI still needs to be valid. Moreover, when these access points are exposed through a proprietary API or the underlying project data is stored in proprietary file formats, exchange and synchronisation of data becomes more cumbersome and sometimes plainly impossible, as multiple conversion processes will be needed to relocate data from one proprietary CDE to another one. We exclude the situation where the project network is formed by multiple (federated) instances from the same CDE vendor, as the single access point is then governed by a single 3rd party - de facto again centralising the decision-making on data access. Therefore, a generally federated CDE (i.e. as a peer-to-peer network of data) should base upon open data standards. In the ideal case, these standards are discipline-agnostic, so connections with external domains using the same structures can be easily established, and a multi-domain SSoI becomes technically possible. As mentioned in Section 4.5.2, the Semantic Web technology stack has been identified as an apt solution to address such interdisciplinary, federated SSoI.

Issue Management In the context of the ongoing establishment of open BIM processes, the need for a communication interface arose to be able to transmit information and issues within the models in a software-independent manner. Therefore, the BIM Collaboration Format (BCF) [buildingSMART Internationa, 2023] was developed by buildingSMART to provide a software- and vendor-neutral exchange format for model-based issue communication and management. Together, three main parts - `Topics`, `Comments` and `Viewpoints` - form the main structure behind BCF and are all connected to a `Project` concept:

1. The `Topic` carries general information about an issue. It uses properties like a current status, a definition of the type of `Topic`, the person who created it and who is responsible for fixing it.
2. `Viewpoints` connect the format to BIM by providing a virtual camera located inside the model that looks at the scene as part of a current discussion. They can also provide links to specific building elements by listing their GUID.
3. A `Comment` concept provides the textual information and an author in a discussion. It is linked to the `Topic` and can reference a `Viewpoint` as well.

The BCF data can either be exchanged in a file-based format using BCF XML[33] or by using a server [van Berlo and Krijnen, 2014] via a REST API (called BCF API) that returns its information in a JSON format. Even though the data can be serialised in two different ways, the overall concept behind both formats is the same, and the structures of BCF API and BCF XML differ only slightly. For example, BCF Servers - using the BCF API - often allow archiving and downloading of the BCF `Project` and its issues as a file in the BCF XML format. Since the communication in an open BIM process is usually an integral part of many workflows, it is often integrated directly into a CDE [Preidel et al., 2018]. Research regarding the decentralisation of these CDEs and BCF can also be observed. Tao et al. [Tao et al., 2021] describe a CDE distributed via a blockchain. This example combines the principles of a CDE with BCF by distributing them as BCF XML over the blockchain. In Gürtler et al. [Gürtler et al., 2015], BCF is implemented in a multi-model-based environment and combined with workflows based on Information Delivery Manuel (IDM) (ISO 29481-1:2010, [International Organization for Standardization, 2010]) and Model View Definitions based on XML (mvdXML) [Beetz et al., 2018] to further enhance the collaboration between different stakeholders.

Since the current BCF serialisations lack the general contextual information and shared meta-models that RDF solutions have, the bcfOWL ontology [Schulz et al., 2021] was created to bring together the worlds of Issue Management and Linked Data. In the proposed linked data model, BCF issues are stored as RDF triples and can be queried using SPARQL. Additionally, the different concepts for the `Topics` defined in the BCF `Extensions` can be enriched with further semantics. Therefore, the `Extensions` in bcfOWL can be used as a gateway to other ontologies in the context of LBD. The ontology is not introducing new concepts to BCF and has semantic interoperability with BCF XML and BCF API. A converter can be created to serialise bcfOWL data into the standard BCF JSON and XML formats. Thus the ontology can serve as a shared foundation for both formats. By using SPARQL to query the data stored as bcfOWL, it is also possible to overcome the accessibility limitations of BCF caused by its hierarchical structure, as described in [Schulz and Beetz, 2021].

[33] buildingSMARTs BCF XML: https://github.com/buildingSMART/BCF-XML (Accessed 12.12.2022)

Related initiatives

As opposed to monolithic software applications, the concept of microservices has known a steep adoption curve over the last decade. Microservices differ from monolithic software so that instead of being a single unit of an application, a combination of small services is formed, keeping the central management to a bare minimum. The aim is for the services to be as loosely coupled as possible. As outlined by Fowler et al. [Lewis and Fowler, 2014], the microservices architecture is based on a general programming principle, where systems to be created can be put together from off-the-shelf components. Unlike libraries, which are coupled with the running software, in the microservices architecture, the functions are offered as separate services that can be called using messaging over the Internet. Amongst their benefits are granular scaling, enabling heterogeneous systems, and more flexible deployment. In the following paragraphs, we discuss several AECO-related initiatives which base upon a microservice architecture. The focus will lie on research-oriented CDEs that are open-source, which allows to identify features corresponding with federated data management (Section 4.5.3).

One of the earliest attempts (within the AECO industry) to establish a microservice-oriented framework for interoperable, integrated building product model for the entire life cycle of the asset was the BLIS-SABLE project [Kiviniemi et al., 2005]. BLIS-SABLE was established as a combination of the BLIS (Building Lifecycle Interoperable Software group) and SABLE (Simple Access to Building Lifecycle Exchange) projects. A set of domain-specific APIs was developed, where each API facilitates the interaction with a potentially federated model, consisting of IFC sub-models. The project finished in 2002.

Another example of early service-based infrastructures for AEC is the concept of BIM bots, interacting with data on an online model server. BIM Bots form an online system of autonomous agents, each bot having a minimal task in managing information in a digital asset model. They form a decentralised network of services where data is exchanged as IFC/STEP or JSON over HTTP, the same way the industry collaborates. BIM Bots are triggered by events, and each one covers a business case implemented as open source or proprietary. For example, a BIM bot can automate interactions between online tools, perform analyses or simulations on BIM data, enrich a BIM model by detailing, and apply a Model View Definition filter for a specific scenario. The concept of BIM Bots in an open standard context is often related to the IFC-based model server BIMserver [Beetz et al., 2010, van Berlo and de Laat, 2018]. However, the concept can be broadened to proprietary ecosystems as well, as illustrated by the recent upcoming of APIs for creating

microservices or browser applications on top of CDEs (e.g. Autodesk Platform Services [Autodesk Inc., 2023] or Trimble Connect API [Trimble Inc., 2021]).

More recently, several research projects have focused on the usage of Semantic Web technologies for data-based information exchange for AECO purposes. The DRUMBEAT platform [Vu Hoang and Törmä, 2016], which finished in 2017, is an implementation of the Linked Data concept for building information with a RESTful API to access the objects. The platform is open and decentralised, meaning that any party can install it and use it to publish its data. Objects published at different hosts and installations of the platform can refer to each other, and the platform ensures that the remote inverse links get updated appropriately. The ifcOWL ontology forms the core data model of the DRUMBEAT project.

Another initiative which uses micro-services powered by Linked Building Data is the SCOPE (Semantic Construction Project Engineering) project [Huyeng et al., 2020], which was finalised in 2021. In SCOPE, a system architecture is proposed that allows micro-services to interact with full RDF-based BIM models stored in a triple store. These micro-services (e.g., a Revit-to-RDF exporter, a rendering service) are then bundled in a Docker network and exposed via a dedicated API gateway, so users can access them in a unified way, e.g., to be used for structural analysis in other tools. SCOPE focuses on the connection of a project CDE with external product datasets, using semantic Web technologies.

The RUB-ICDD platform [Hagedorn et al., 2023] is a CDE based on the Information Container for linked Document Delivery (ICDD) (ISO 21597, [International Organization for Standardization, 2020]). ICDD is an industry-standard to containerise and link individual project-related datasets into a larger 'multi-model' [Gürtler et al., 2015], relying on RDF as the semantic glue that organises the multi-model and sub-document identifiers. While the intended purpose of ICDD was to be a means for archival as a zipped container, it has been identified as a core data model for active CDEs as well [Senthilvel et al., 2020, Karlapudi et al., 2021]. RUB-ICDD is a centralised platform that offers a number of Semantic Web tools to interact with an ICDD-based multi-model.

Finally, the ConSolid project [Werbrouck et al., 2019, 2023] is an ongoing research project investigating the use and potential extension of the Solid specifications to organise a federated CDE that is compliant with the FAIR principles. Based on metadata patterns and nested catalogues of federated project data, query-based discovery, filtering and sub-document linking of project resources are made possible in a federated environment. Conform with the reasoning behind Solid, applications (client-side and headless microservices) are decoupled from

data storage. Data pluripotency (i.e. the potential for reusability), established using open Web and industry standards, is the ecosystem's core enabler of interdisciplinary semantic enrichment of federated data. The high-level data patterns and principles of the ConSolid infrastructure were outlined in earlier work. However, the core infrastructure purposefully does not consider the content of files, so a demonstration of its use as a CDE which supports existing industry workflows and requirements has not been covered yet.

4.5.3 Towards a federated CDE

Requirements

A technology-agnostic overview of the components of a federated CDE needs to be laid out, which is the main topic of this section. The requirements of such a system are outlined below and derived from the challenges that were listed in the Introduction section.

We identify the following high-level requirements:

- **R1** **Decentral storage**: access controlled data vaults, based on decentral authentication (R2). Data storage should ensure that the data is tampering-proof (data integrity).
- **R2** **Decentral authentication**: verify a visitor's identity based on protocols for decentral identity providers. After authentication, this identity can be used for authorisation.
- **R3** **Reusable data organisation patterns**: storage patterns that are as machine-readable and structured as possible [Berners-Lee, 2012]. In other words, rich data with linked context (e.g. bcfOWL) can be used to generate other data structures for a specific purpose (e.g. BCF) - the inverse is more difficult and thus less reusable.
- **R4** **Data heterogeneity**: a building project will consist of structured (e.g. semantics), semi-structured (e.g. geometry) and unstructured datasets (e.g. imagery). The ecosystem must acknowledge this heterogeneity and make no assumptions about the formats and schemas it will host.
- **R5** **Metadata description**: although datasets may differ in schema or data type (R4), they should have uniform metadata description patterns as to allow general discoverability and filtering of datasets. The requirements for metadata to allow this are covered by the FAIR principles [Mons et al., 2017].
- **R6** **Query interfaces**: supporting access to the (meta)data through existing general-purpose query languages.

R7 **Adapters**: fulfilling the task of reconfiguring generic (reusable) data patterns to subsets in a format readable by external applications. These may be application-specific or provide domain-specific standardised views on the data, useful by multiple applications. Adapters can be microservice-based APIs or directly embedded in client application code.

R8 **Aggregators**: provide access to federated datasets. Project policies may allow it to cache project data in order to buffer the potential disconnection of agents in the network.

The first requirement (R1) serves the need for a decentral storage environment. As the Web itself is a decentral storage system by design, this requirement is easily fulfilled when considered standalone in the context of openly published knowledge. In the case of access-restricted information, information will be hosted by one or more secure data vaults. A high-trust environment should guarantee that the data has not been tampered with. Hashing and encryption technologies may provide such a guarantee but are generally considered out of scope for this paper.

The second requirement (R2) imposes additional complexity: decentral authentication technologies avoid dependency on central identity providers and allow any actor to manage a verifiable identity on the Web, either by themselves or by outsourcing to a trusted provider. Effectively, a standardised identity management protocol is essential to verify identities between disparate identity providers. Note that the case of trusted third-party identity providers is not eliminated. However, concerning authentication, a communication layer between them needs to be in place so that identity providers can verify identities hosted by other providers.

Independent from the previous requirements is (R3) the notion that data should be useful for multiple purposes. Data reusability increases according to the 5-star deployment scheme for open data proposed by Berners-Lee [Berners-Lee, 2012], and is one of the cornerstones of the FAIR data principles. The more structured the data becomes, the less difficult it becomes to extract the knowledge it describes; consequently, it is also easier to reconfigure or adapt this information to data structures suitable for a particular purpose or conforming to an existing data standard. Note that the fact that data in the ecosystem might (and will likely be) access-restricted rather than open has little impact on the degree of structuredness of the data and hence its potential for reusability.

Although resources contributing to the setup of a digital asset model are becoming increasingly machine-readable, some resources will remain semantically ambiguous because of their inherent unstructuredness or because a less structured form is preferred over the highest possible degree of machine-

readability (e.g., because more standardised support, better tooling). This relates to (R4), i.e. data heterogeneity. Examples of the former are binary data formats for storing imagery or point clouds; the latter could refer to geometric serialisations that existing geometry kernels can read immediately.

Metadata records (R5), i.e., data about the datasets, are necessary to align seemingly incompatible requirements R3 and R4, in other words, ensure that data pluripotency is not an all-or-nothing requirement but rather a sliding scale. For example, indicating the media type of a resource allows the automatic selection of a fit piece of software for interpreting the data or aligning its content with outside datasets. As noted in the costs and benefits of open data with only one star, '*other than writing a custom scraper, it is hard to get the data out of the document*' [Berners-Lee, 2012]. For common formats, such custom scraper may, however, be already available (e.g. as a microservice) and automatically addressable. Metadata structures also serve the purpose of levelling the field between heterogeneous resources to a certain degree, as they allow to link sub-document elements in different resources without impacting these resources, hence opening up the silos of unstructured and semi-structured datasets to be more permeable for external references. The presence of structured metadata is one of the cornerstones of the FAIR principles, which outline the requirements for metadata in higher detail [GO FAIR, 2016].

The more resources in a digital project become connected, the better a domain-agnostic query language (R6) is able to filter project data into case-specific subsets. This minimally holds for querying metadata patterns but is more potent if also actual project resources can be queried. However, for this to be used maximally, semantic data needs to be as structured as possible (i.e. as close as possible to a 5-star description). To use this requirement to its full extent, only inherently unstructured data can be stored otherwise. If needed, 5-star data can then be reorganised by external services (see R7) to match less structured data schemas.

Existing tools and standards often expect a specific schema or data format, which seldom aligns with generic structured data. If needed, structured, interconnected data with universal context can then be reorganised by adapter services (R7) to dynamically expose the data in a different form as to comply with a standardised or proprietary API while still relying on the Single Source of Information provided by the graph of structured data. At the same time, these intermediary services may facilitate compatibility between stakeholders who use different maturity levels in their workflows.

Finally, a single data access point will often be expected, either again for compatibility reasons or for the sake of querying performance. This role is fulfilled by aggregators (R8), which funnel federated data and expose it in a specific way.

	decentral storage	decentral authentication	data structures	heterogeneous storage & linking	query interfaces
BLIS-SABLE	yes	no	IFC	no (IFC)	no
BIMserver	no	no	IFC	no (IFC)	custom (JSON-based)
DRUMBEAT	yes	no	ifcOWL (RDF)	technically possible (RDF)	SPARQL
SCOPE	yes	no	LBD (RDF)	technically possible (RDF)	SPARQL
RUB-ICDD	no	no	multi-model (ICDD)	linksets (ICDD)	SPARQL
ConSolid	yes	yes	multi-model (DCAT)	federated linksets (ConSolid ontology)	SPARQL

Table 4.8. Comparison between different collaborative AECO platforms in research

Aggregators can host synchronised copies of project data in order to speed up the retrieval and querying process or fetch and serve the data dynamically. Retention policies applying to the original (federated) data can instruct aggregators what to do if stored data becomes unavailable or outdated. Such policies are to be set on the project level. As such, aggregators might serve as backups for project data as well, if such behaviour is intended by the project consortium.

Dynamic Project Boundaries

Contrary to a central CDE, in a discovery-based, federated project environment, a project does not have well-defined boundaries in terms of what information to include. Rather, these boundaries will be defined by deciding whether or not to follow pointers to relevant datasets. This is a consequence of the combination of the requirements for reusable data and a federated architecture, and thus not a requirement in itself. As outlined in [Werbrouck et al., 2023], this combination makes data aggregation very scalable: in a classic approach, 'project data' is the total of information produced by direct project stakeholders. However, other aggregations are possible as well. For example, for an office seeking to learn lessons from previous assignments, the total amount of 'datasets of interest' may just consist of a higher-level catalogue of their own contributions to past initiatives, i.e. all data on their vaults. Similarly, other tasks may require the aggregation and analysis of asset data from a typological or geospatial perspective. For example, 'all libraries in Flanders' could be a catalogue of hyperlinks to federated sub-projects managed by the government. This catalogue, in turn, can be nested in one or more bigger ones, e.g. 'all public buildings in Belgium'. However, since outgoing hyperlinks can be followed virtually endlessly, it is up to the client to determine the relevant link-traversal depth, and hence the extent of the 'project'.

Comparison of environments

We can now relate the requirements outlined in Section 4.5.3 to the CDE solutions introduced in Section 4.5.2. However, this paper does not aim to provide an in-depth comparison between CDE solutions. The requirements on storage (R1) and authentication (R2) are directly applicable. An RDF- or multi-model-based project data organisation implies the possibility of highly structured datasets according to the 5-star principles (R3), including heterogeneous storage (R4). In this case, multi-model-based approaches will be more generic as they do not require a core RDF project model to relate information to - cross-document linking will happen on the metadata level. For cross-document linking, ICDD-based platforms such as RUB-ICDD will rely on ICDD linksets. The ConSolid platform introduces a variant of this to allow asynchronous linkset creation and linking, which is required in a federated context. Uniform metadata descriptions (R5) with a common query interface (R6) are only necessary if the project files are heterogeneous - otherwise, a custom query language can be used. The requirements on adapters (R7) and aggregators (R8) are of lesser use to such comparison, as they exist as a consequence of, respectively, highly structured data and decentral storage. An overview of these platforms and their features in the context of federated, heterogeneous CDEs is given in Table 4.8.

Commercial platforms are excluded from the comparison - as they are proprietary, an evaluation of their internal structures is not possible. Generally, we can note that the business model of commercial platforms is oriented towards central data storage (conflicting with R1) of proprietary models (conflicting with R3 and R4). Authentication on these platforms is often based on OIDC, where the platforms act as Identity Providers. Although the internal ecosystem of these CDEs benefits from such authentication, it is not considered fully decentral, as cross-platform authentication is not possible.

Of course, most of these infrastructures, ConSolid excluded, were designed with different purposes than combining data federation and heterogeneity, which drives the logic behind the requirements in Section 4.5.3. While ConSolid is one implementation that matches the requirements, other approaches are certainly possible. However, since it combines the requirements and has been validated against the FAIR principles [Werbrouck et al., 2023], in the following sections we will base upon the ConSolid infrastructure and data patterns for devising a layered infrastructure that allows communication between clients and federated project vaults using standard industry workflows.

4.5.4 High-level Architecture

This section discusses how the requirements of Section 4.5.3 can be used as engineering concepts for a generic infrastructure (Section 4.5.4) that can be technologically implemented (Section 4.5.4). The interaction patterns between the different components will be reusable across data usage scenarios.

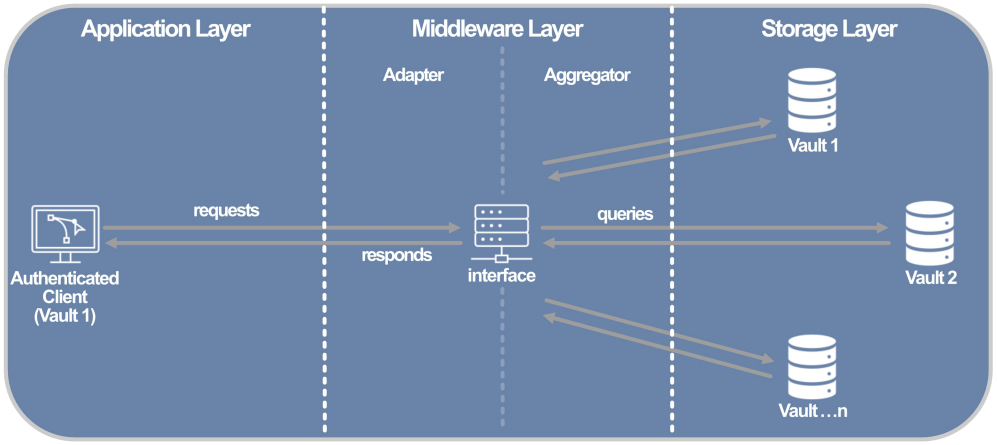


Fig. 4.18. Generic CDE Framework

High-Level Components

The federated framework consists of three layers (Figure 4.18):

1. A client, which is authenticated via a decentral identity (R2). Both GUI-based applications and independent services are considered clients in the network.
2. Middleware, generally consisting of aggregators (R8) and adapters (R7) using standardised APIs or query languages (R6). A chain of middleware layers is possible. The number of middleware functions is not limited, hence it is up to the consortium to define which functionality is required for which phases of the project.
3. The storage layer, storing all information in the respective vaults (R1) of its creators. It is recommended to store project data as highly-structured as possible (R3). In general, unstructured, heterogeneous data (R4), such as imagery and point clouds (unstructured) or geometry (semi-structured) should also follow standardised, open formats, to be useful for as many client applications as possible. All datasets are described using uniform metadata descriptions (R5), allowing their discovery and filtering using a common query language.

As discussed in 4.5.3, the endpoints of a federated project can be discovered via recursive catalogues. Any identified stakeholder in the project needs to instantiate such catalogue, which serves as a gateway to discover and filter datasets in the entire project. Figure 4.19 illustrates this principle as implemented in [Werbrouck et al., 2023], using the Data Catalog (DCAT) vocabulary [Albertoni et al., 2023] to traverse the project graph.

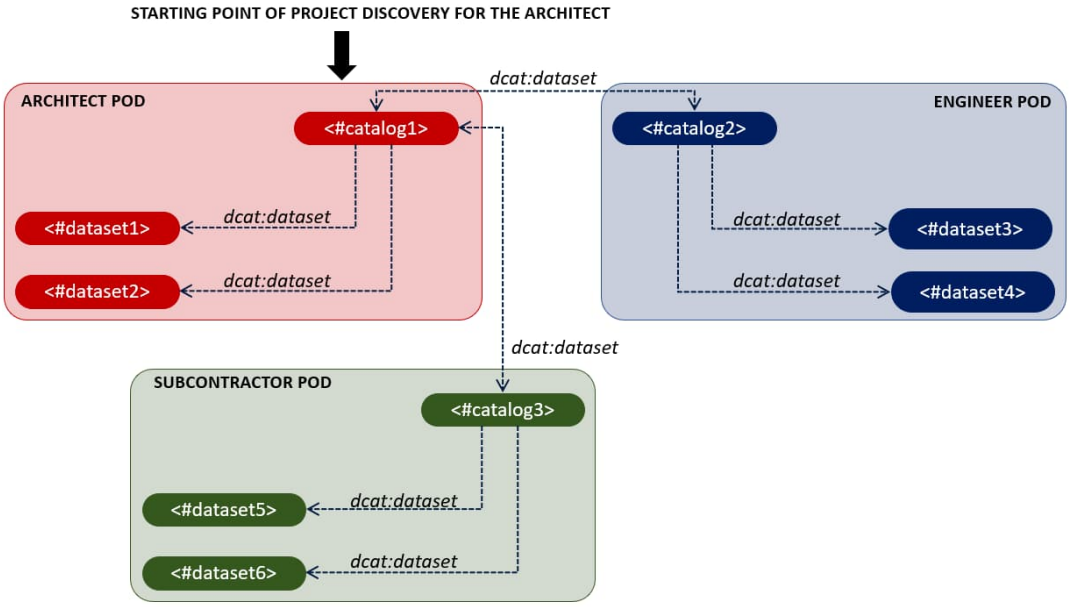


Fig. 4.19. Graphical discovery patterns of other vaults in the project.

Image source for Image 4.19 is [Werbrouck et al., 2023].

We now consider the basic CRUD (Create, Read, Update, Delete) interaction processes with federated project data. For the writing process (Create), the client sends the payload with the newly created resources to the appropriate middleware service. Since there is no single server to host the data, services in the middleware layer need to decide which vault the data needs to be stored in. The storage policies of the ecosystem should be to host all newly created resources in the creators vault. I.e., while it is technically possible to upload resources to the vaults of other stakeholders (if they grant the required access rights), it is advised to apply such restriction to avoid ambiguities about data ownership. Saving data to a different vault should be prevented by the access rights of the other stakeholders vaults. Every stakeholder should only have reading and no creating, updating or deleting rights to vaults they do not own. The middleware service should be able to infer the location of the data vault from the authorisation parameters. A possible protocol to do this is described in the WebID-OIDC [Coburn et al., 2022] specification of the Solid ecosystem.

```
SELECT * WHERE {
  ?catalog a consolid:Project ;
  dcterms:identifier "46db9652-9d6b-4bd7-894b-462d3c807661"
  dcat:dataset ?accessPoint .

  # keep only those access points that are located on other vaults
  FILTER regex(str(?accessPoint), '^((?!https://pod.architect-office.de/architect/).)*$')
}
```

Listing 4.2. SPARQL query to identify all project endpoints.

When information is requested (Read) from the client, the functional layer first queries the project gateway for the endpoints. When the participating data vaults are known, the middleware service can now execute the original query in a federated way. After collecting the information, it is converted into the required data structure and sent back to the client.

The updating process is conceptually more complicated because the topic being updated does not have to be owned by the person doing the update. In the context of this paper, an update refers to the process of someone in the consortium creating a new version of a specific project resource rather than a combination of deleting data and inserting the changes (a combination of delete and write operations). The question arises where the data should be stored so that ownership is evident. The original data is not touched, and all updated properties are stored in the vault of their creator. To obtain a complete view of a resource, a middleware service assembles this view from the data in the vaults. The way the data is assembled can differ depending on the middleware service. The approach chosen is intended to ensure that original data is never changed and that the history of the respective properties can be traced back at any time.

Any interaction to delete resources would be similar to creating new resources. However, creating newer versions or archival of deprecated resources is preferred over deleting resources, which should only be done with great care. Project consortia can implement retention policies which govern the archival and deletion of resources according to unambiguous parameters.

Technology Stack

Until this point, the concepts and requirements have remained largely technology-agnostic. However, a mapping to existing technologies is necessary to demonstrate the general feasibility of such infrastructure. In this section, we will identify existing technologies which allow a practical set-up of the ecosystem.

```
SELECT * WHERE {
  ?catalog a consolid:Project ;
  dcterms:identifier "46db9652-9d6b-4bd7-894b-462d3c807661"
  dcat:dataset+ ?dataset .

  ?dataset dcat:distribution ?dist .
  ?dist void:vocabulary <http://lbd.arch.rwth-aachen.de/bcfOWL#> ;
  dcat:downloadURL ?url .
}
```

Listing 4.3. SPARQL query to identify all project resources structured using bcfOWL. Catalogues on other vaults are also retrieved via dcat:datasets. Optional filters on metadata records may be present to identify a subset of project resources.

Data Storage and Identity Management For the Data Storage and Identity Management layer, we base upon the Solid specifications and their implementation as the Node.js-based Community Solid Server (CSS) [Van Herreweghe et al., 2023], using a version of this Community Server that was extended in the ConSolid project. In this extension, data is not only served via the HTTP routes, following the Linked Data Platform (LDP) specification but in addition, a SPARQL interface is provided that mirrors all RDF resources on the Pod as named graphs. The default graph is queryable as the union of all permitted named graphs. This corresponds with the upcoming vision to consider a Pod a 'hybrid knowledge graph', outlined in [Dedecker et al., 2022], yet is not included in the current specifications yet. Furthermore, the strict containerisation of the LDP specification is dropped and replaced by a concept of virtual views, which can be generated dynamically by filtering the metadata [Werbrouck et al., 2022]. Consequently, URLs do not reflect any resource hierarchy anymore, in contrast with the tree-based structure offered in the standard CSS. Middleware services as identified in section 4.5.4 can provide such virtual views. Authentication in Solid is based on the Web Access Control (WAC) [Capadisli and Berners-Lee, 2022] and the Access Control Policy [Bosquet, 2022] RDF vocabularies.

Reusable Data In previous sections, we introduced the requirement of data reusability, so it can be queried and re-organised to fit multiple purposes. We identify RDF as the currently most suitable, natively decentral data structure for data interoperability and reusability, in line with domain-agnostic research and recommendations [Wilkinson et al., 2016, Mons et al., 2017]. As mentioned, a unified structure is specifically required for metadata. The project data itself is expected (and allowed) to be heterogeneous, so RDF-based knowledge modelling for domain-specific data cannot be enforced, although it will have multiple advantages as well [Beetz, 2009, Pauwels et al., 2017, Rasmussen et al., 2021] - corresponding with RDF being the basis for 5-star Linked Open Data [Berners-Lee, 2012]. Heterogeneous data, such as imagery and models, are thus saved

in their respective formats and can be annotated with metadata via RDF. In line with metadata practices adopted by the European Union [Kirstein et al., 2019], the DCAT (Data Catalog)[34] vocabulary is used as a basis for structuring RDF-based metadata in the ConSolid framework. Listing 4.2 shows how DCAT is used to recursively discover the other registered endpoints in a project access point. Any interaction of project data with a microservice is initiated by this request, in order to know which endpoints to query for the subsequent requests. Figure 4.19 illustrates how a project catalogue from one stakeholder contains the catalogue from the other stakeholders, so bi-directional discovery of datasets is possible. Listing 4.3 provides a query to discover the project datasets available on a given data vault. Project resources are `dcat:Distributions` of metadata descriptions (`dcat:Dataset`). A project is then any catalogue (`dcat:Catalog`) containing relevant datasets (`dcat:Dataset`) or other `dcat:Catalogs`. The project is identifiable with the property `dcterms:identifier`, referring to a unique project ID. When the project ID is known, a recursive query pattern can be used with which this single catalogue resource can be used to identify all other catalogues in the federated project network (Listing 4.2).

Next, the retrieved access points can be queried themselves for the project datasets on that particular vault, along with their distributions and where to download them (Listing 4.3). Additional filters can be added, e.g. to solely include resources conforming to a specific media type or specification (e.g. bcfOWL, see Section 4.5.5).

Query Interface The SPARQL Protocol and RDF Query Language (SPARQL) is used as the query interface (R6) for all DCAT metadata residing on the Pods. The SPARQL interface on top of ConSolid Pods can be used by microservices as an input for specialised APIs, or be queried directly by authorised clients. Applying SPARQL on DCAT metadata allows project-wide filtering of datasets with specific metadata properties. After this filtering, dedicated query languages can then be used to query project datasets of compatible media types, although project data structured using RDF ontologies can be queried with SPARQL as well.

Aggregators and Adapters Aggregators and Adapters are web services which can be programmed in any suitable programming language. In this capacity, they should conform to the HTTP protocol [Fielding et al., 1999], and ideally adhere to a RESTful architecture.

[34] Data Catalog Vocabulary (DCAT): https://www.w3.org/TR/vocab-dcat-3/ (Accessed 12.12.2022)

4.5.5 Proof of Concept

In this section, we demonstrate the envisaged ecosystem and its compatibility with existing workflows. First, we describe how the ecosystem facilitates federated Issue Management, based on the BCF API specification and the bcfOWL ontology. The workflow is illustrated using the benchmark dataset 'Schependomlaan' provided by buildingSMART[35], and a sample workflow targetting the 'Openings and Recess Management' [van Berlo et al., 2017, Verein Deutscher Ingenieure, 2022], which is considered a subset of Issue Management that is common during the BIM-based design phase of a building project.

Then, we illustrate the general applicability of the framework by describing how another specification, the Documents API, can be implemented by following the same interaction patterns. The Proof of Concept development is mainly based on Node.js applications that are programmed using JavaScript and TypeScript, available on Github[36].

Issue Management Workflow

A regular Issue Management Workflow contains of a chain of communications between different participants in a project. In our example workflow, a request is submitted by the MEP engineer whether an opening can be created in a certain area of the building so that technical equipment (for example, a pipe) can be routed through it (Figure 4.20). Depending on the defined workflow in the project, this request is forwarded to the appropriate entity and checked. Both the structural analysis and the architectural design must be considered for checking. Once checked, the results are sent back to the MEP engineer. The open BIM format BCF is often used for this process since stakeholders use different BIM software in many projects. An example workflow following the BCF specification is available on Github[37]. In our scenario, we consider an execution via the BCF API, which allows communication with the ecosystem to take place as follows:

1. The MEP engineer creates a new `Topic` and assigns it to a reviewer. The `Topic` contains information such as what kind of request it is (`Topic Type`), the `Topic Status` (usually 'Open' when created), and more. A `Viewpoint` concept

[35] Schependomlaan dataset: https://github.com/buildingSMART/Sample-Test-Files/tree/master/IFC2x3/Schependomlaan (Accessed 02.06.2023)

[36] Proof of concept codebase: https://github.com/Design-Computation-RWTH/federated-issue-management (Accessed 06.06.2023)

[37] Example BCF Topic: https://github.com/Design-Computation-RWTH/federated-issue-management/tree/main/sample_data (Accessed 05.06.2023)

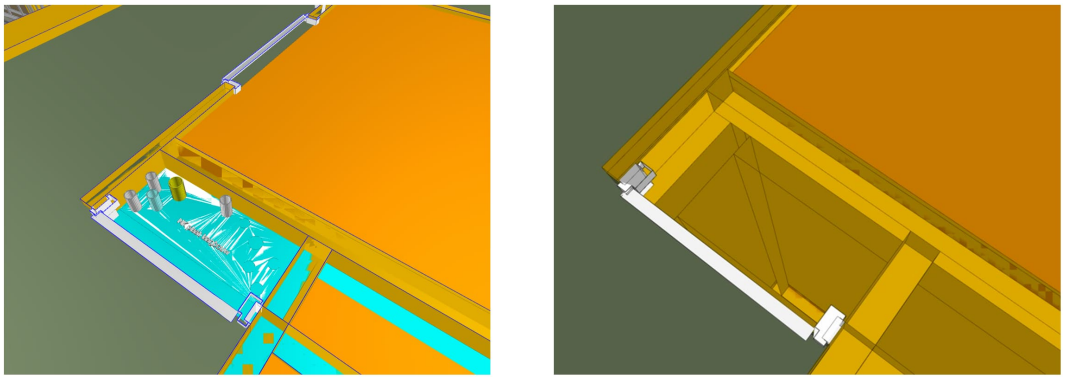


Fig. 4.20. Left-hand side: geometric clash related to 'Openings and Recesses' present in the Schependomlaan dataset. Right-hand side: geometry after the issue is resolved.

links to the `Topic` and defines a camera position pointing to the scene and, in an optimal case, references the corresponding building elements of the BIM. An optional `Comment` concept can be attached to the `Topic` if a discussion is needed.
2. After the assignment, the reviewer can filter all `Topics` assigned to them and review them.
3. After the check, the `Topic` is updated, and the `Topic Status` is updated according to the result. There may be an assignment to another reviewing entity or an assignment back to the MEP engineer.
4. The MEP engineer checks the `Topic's` status to continue with the design or propose an alternative opening according to the result.

To facilitate these interactions, we use the generic framework described in Section 4.5.4. Therefore, the first layer (the client) and the middleware layer, which combines the role of aggregator and adapter, use the buildingSMART communication standard BCF API. The user uses BIM authoring software with BCF communication capabilities. The federated identity associated with the user's vault is used for communication with the middleware and data layers.

To discover all projects the client is participating in, the middleware (aggregator role) service queries the user's data vault. For each project, the vault responds with the location of a project catalogue containing metadata about the project and hyperlinks to similar access points on the vaults of the other stakeholders in the project. Then, the aggregator sends a federated query to all stakeholder vaults to assemble the requested information. This query includes a filter to query for project datasets relating to Issue Management, which needs to be included in the dataset's metadata. The discovered datasets are then retrieved by the middleware service and converted (adapter role) to the desired schema of the BCF API. Finally,

the middleware service sends the information back to the client as an HTTP response.

In our scenario, the MEP engineer wants to create a new BCF issue to request approval for an opening in a wall so that pipes can go through that wall. The newly created resource contains all the essential information for the approval request, such as the author, the creation date, the type, the corresponding building elements, and a camera perspective looking at the scenery, as well as a responsible partner. The BCF API (middleware) can now be used to query for all issues that need approval by a specific agent: the middleware layer collects the data from the project vaults and exposes it in BCF format, so the responsible agent (here the architect) can now approve or reject the issue. In the BCF context, approving or rejecting involves an updating operation on the `Topic Status`. As discussed in Section 4.5.4 the updated properties are stored in the vault of its creator. As soon as the `Topic` gets queried, the adapter is responsible for constructing a compliant (most recent) view of it and sending it to the client, who can now approve or reject the request. After this step, the issue can either be assigned to a further reviewer or, depending on the project's agreed-upon workflows, be assigned back to the MEP engineer. The engineer can repeat the discovery process by querying and filtering all `Topics`.

Practical Setup

In the flattened approach of ConSolid, data is not managed in a specific container hierarchy. Hence, contrary to Solid's implementation of LDP containers, URLs do not contain their parent containers: a resource is 'contained' in a catalogue only because this catalogue references it as such. Therefore, a resource (which may be a container itself) could theoretically have multiple parent containers. However, in Issue Management, more specifically in the BCF API, resources are structured in a hierarchical tree. A microservice querying the project can filter (meta)data to restructure information according to a derived structure. In this case, this structure will be dictated by the BCF API specification. A BCF Project and its constituent resources are considered a subset of all resources belonging to a federated project. This includes the following:

1. One or more resources are identified as BCF `Extensions`.
2. One or more resources are identified as BCF `Topics`.
3. One or more resources are identified as BCF `Comments`, internally linked to a BCF `Topic`.
4. One or more resources are identified as BCF `Viewpoints`, internally linked to a BCF

`Topic`.

As the ConSolid ecosystem does not make any assumptions on the nature of the project resources, these can range from IFC files over pure geometry, imagery or RDF-based semantics. In our example, the resources contain Issue Management data structured using the bcfOWL ontology [Schulz et al., 2021], in line with the identification of RDF as the most suitable approach towards interoperable and reusable data. In the following section, the specific interaction patterns between the federated project and a microservice that implements the BCF API specification are illustrated, following the scenario outlined in Section 4.5.5.

Figure 4.21 illustrates the communication between the components necessary for the scenario in Section 4.5.5, where the MEP engineer creates a new `Topic` for reviewing an opening in the building model and assigns it to the architect for reviewing it. Listing 4.4 contains an example of how this `Topic` could be stored in the vault using RDF and bcfOWL. Simultaneously, a metadata record for this `Topic` resource is created to make it discoverable from the SPARQL interface to the vault. For updating and describing the history of bcfOWL resources, we refer to the approach of [Oraskari et al., 2022a], which discusses the use of resource states. After the architect (i.e. the assigned party) has reviewed the `Topic`, they create a new `Topic State` for it. All updated properties are added to this `bcfowl:TopicState`, including a pointer back to the original `Topic` on the vault of the MEP engineer (`bcfowl:hasTopic`). Listing 4.5 provides an overview of how this resource would now be stored in the architect's vault.

To get a current view of the entire `Topic` via the BCF API, the middleware service (adapter and aggregator) needs to query all `bcfowl:TopicStates` referring to the original `Topic`. To construct its most recent view, the property `bcfowl:hasCreationDate` can be used. An example query is given in Listing 4.6. The results of this query are included in Table 4.9 which shows the results from the different vaults after aggregation by the middleware service. Similarly, the provenance of data ownership can be derived from the property `bcfowl:hasCreationAuthor`.

?topic	?cdate	?author
<https://pod.mep-engineer.eu/Topic1>	"2022-12-12T13:45:30"	<https://pod.mep-engineer.eu/profile#me>
<https://pod.mep-engineer.eu/Topic1>	"2022-12-13T12:14:15"	<https://pod.architect.eu/profile#me>

Table 4.9. Results of the query in Listing 4.6, based on the data in Listings 4.4 and 4.5. Rows in the table represent found Topic States.

The same discovery patterns can be used for the other BCF concepts.

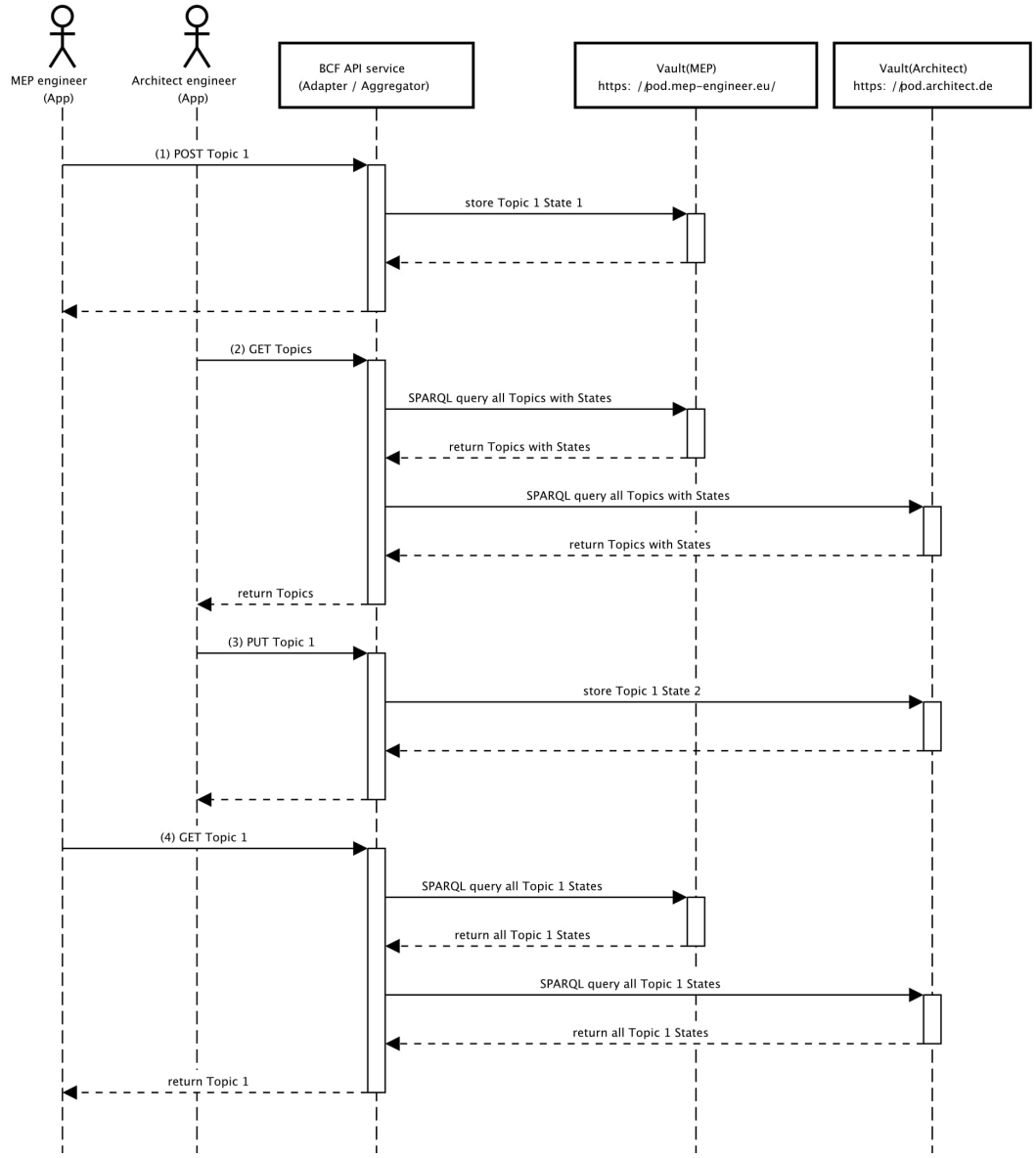


Fig. 4.21. Sequence Diagram of the requests a client application (Application Layer) makes to a BCF API implementation (Middleware Layer), which aggregates data from the vaults in the project (Storage Layer) and restructures ('adapts') this information to match the requirements of the BCF specification. The sequence corresponds with the scenario described in 4.5.5.

BCF-external Issue Management

The framework described in this paper is not exclusively aimed towards BCF-compatible Issue Management. Instead, the separation of storage and

```
<https://pod.mep-engineer.eu/Topic1> a bcfowl:Topic ;
bcfowl:hasProject <https://pod.project-manager.eu/OpeningsAndRecessProject> .

<https://pod.mep-engineer.eu/Topic1State1> a bcfowl:TopicState ;
bcfowl:hasTopic <https://pod.mep-engineer.eu/Topic1> ;
bcfowl:hasTitle "Please check this opening" ;
bcfowl:hasTopicType <https://pod.project-manager.eu/OpeningsAndRecesses> ;
bcfowl:hasCreationDate "2022-12-12T13:45:30" ;
bcfowl:hasCreationAuthor <https://pod.mep-engineer.eu/profile#me> ;
bcfowl:hasTopicStatus <https://pod.project-manager.eu/ExtensionStatusOpen> ;
bcfowl:hasAssignedTo <https://pod.architect.eu/profile/card#me> .
```

Listing 4.4. The original `Topic` and its `bcfowl:TopicState` are stored on the vault of the MEP engineer.

```
<https://pod.architect.de/Topic1State2> a bcfowl:TopicState ;
bcfowl:hasTopic <https://pod.mep-office.eu/Topic1> ;
bcfowl:hasCreationDate "2022-12-13T12:14:15" ;
bcfowl:hasCreationAuthor <https://pod.architect.eu/profile#me> ;
bcfowl:hasStatus <https://pod.project-manager.eu/ExtensionStatusClosed> ;
bcfowl:hasAssignedTo <https://pod.mep-engineer.eu/profile/card#me> .
```

Listing 4.5. The updated `bcfowl:TopicState` is closed by the architect and reassigned to the MEP engineer. The new state is stored on the vault of the architect

```
SELECT ?topic ?cdate ?author WHERE {
  ?catalog a consolid:Project ;
  consolid:projectId "46db9652-9d6b-4bd7-894b-462d3c807661" ;
  dcat:dataset+ ?dataset .

  ?dataset dcat:distribution/dcat:downloadURL ?topic .

  ?topicstate a bcfowl:TopicState ;
  bcfowl:hasTopic ?topic ;
  bcfowl:hasCreationDate ?cdate ;
  bcfowl:hasCreationAuthor ?author .
}
```

Listing 4.6. An exemplary query to retrieve all `Topics` of a project from the vaults.

functionality and the presence of query interfaces to the data vaults allow to create divergent APIs, co-existing in the middleware layer. For example, issues related to the 'Openings and Recess Management' that was used in Section 4.5.5 and 4.5.5 can easily include several hundreds of requests for openings.There In [van Berlo et al., 2017], this process was mapped to the BCF file format, and it was found that a spatial change of an opening can quickly lead to several new requests, and it is not straightforward to filter out outdated requests. This problem persists using the BCF API instead of the BCF file format. Furthermore, [Schulz and Beetz, 2021] found that in specific cases, the BCF API is unsuitable for generating queries for affected building elements. Thus, if the MEP engineer wants to query the issues attached to a specific building element (Step 4 in Section 4.5.5), this cannot

```
POST /select-documents
Body:
{
  "server_context": "46db9652-9d6b-4bd7-894b-462d3c807661",
  "callback": {
    "url": "http://example.org/cde-callback-example",
    "expires_in": 3600
  },
  "supported_file_extensions": [
  ".ifc",
  ".ifczip"
  ]
```

Listing 4.7. Example request for requesting resources using the Documents API [buildingSMART International, 2022].

```
SELECT * WHERE {
  VALUES ?ext { ".ifc" ".ifczip" }

  ?catalog a consolid:Project ;
  consolid:projectId "46db9652-9d6b-4bd7-894b-462d3c807661" ;
  dcat:dataset+ ?dataset .
  ?dataset dcat:distribution ?dist .

  ?dist dcat:downloadURL ?document ;
  dbpedia-owl:fileExtension ?ext .
}
```

Listing 4.8. SPARQL translation of the request in Listing 4.7, querying DCAT metadata on ConSolid vaults.

happen using the BCF API without first requesting all BCF resources from the server and then searching locally for the building elements, resulting in a very high number of requests to the server. An architecture adhering to the requirements discussed in Section 4.5.3 allows two alternative options to address this:

1. directly query the project and its resources using a common query language cf. BIM Maturity Level 3.
2. use a parallel API that uses the same resources as the BCF API but serves the data in a different hierarchy - cf. BIM Maturity Level 2.

In both options, however, the same data from the vaults should be used, corresponding to the data reusability requirement (R3). Custom services and APIs can thus easily co-exist with standard ones.

Document Management Workflow

In this section, we illustrate this with an API endpoint mapping of the Documents API [buildingSMART International, 2022], which is also part of the OpenCDE API family (buildingSMART). The Documents API facilitates the exchange of documents between a client application and a CDE. The case of document synchronisation between two CDEs is identified but not yet supported or documented in the specification. The Documents API is media-type-agnostic and not influenced by the content of the documents to be exchanged. Largely the specification describes three possible interactions:

- Document Download
- Document Upload
- Document Query

Each of these interactions is initiated with a request to the standardised API endpoints `/select-documents`, `/upload-documents` and `/document-versions`, which respond with the CDE-specific URL dedicated to this specific action. In the specification, this is set to an interactive browser URL, yet in this case, we focus on workflows that can be automated - hence we consider a dedicated API endpoint instead of a browser URL which requires further user interaction. An example request for document download is given in Listing 4.7 [buildingSMART International, 2022]. The other interactions can be mapped in a similar way.

The 'server_context' key included in the Document API request is an identifier for a project or a directory on the CDE. The middleware service can directly map this to a `consolid:projectId`. The callback is intended for browser redirects and will thus be omitted here. The metadata descriptions of the ConSolid ecosystem should thus be enough for an aggregator-adapter middleware service to expose a Documents API on top of the federated project. Compliant with recommendations of the FAIR principles, the Web standard for metadata catalogues DCAT is used. The under-the-hood translation of the request in Listing 4.7 to a generic query to the SPARQL interfaces of the project vaults is given in Listing 4.8. The supported file extensions are included in the SPARQL query as extensions (`dbpedia-owl:fileExtension`) of resource distributions `dcat:distribution`.

4.5.6 Discussion

The main contributions of this research are the identification of a set of requirements, which leads to a high-level design for federated frameworks for interdisciplinary collaboration for CDEs. This design was mapped to existing concepts and methods for Web Engineering, such as microservice architectures, the Semantic Web technology stack, WebID-OIDC and the Solid Ecosystem. As the framework is based on a generic architecture, it is not limited to strict AEC-related purposes. Finally, by demonstrating the alignment of a well-known industry standard (BCF API) to the framework, it was shown that the adoption of a federated infrastructure can take place gradually and is not, per definition, disruptive - on the contrary, it is a more truthful reflection of the nature of the industry than centralised solutions. By discussing how disparate APIs can co-exist, we illustrated that the infrastructure is generally applicable and not only useful for Issue Management communication.

After identification of the requirements for a federated infrastructure, the existing (research) platforms introduced in 4.5.2 were compared according to these requirements. In contrast to these existing approaches in the industry, the framework separates functionality from data storage.The prototype also implements the requirement for decentral authentication, based on the WebID-OIDC specification. This eliminates reliance on multiple identity providers. The protocol does not favour one IDP over another, so it is an apt candidate for a cross-CDE authentication layer.

The presented framework is a distributed approach to sharing information where cooperating parties have agreed on standard ways to work together as independent parties. This goes hand in hand with data ownership. Each stakeholder owns and manages their data, which aligns with how the AECO industry is organised. Federated data storage thereby addresses both layers of the patchwork: stakeholder-centric storage guarantees the protection of intellectual property, while data aggregation on the service level allows a project-centric lens on such federated project data. Other stakeholders are not granted the right to change data on other vaults. Access to their data cannot be revoked - as we have seen in [HH JUDGE EYRE QC, 2021] - because it is hosted in their vaults. Thereby, a stronger guarantee of ownership is provided in comparison to central approaches, which is in line with Hypothesis 4.3.

Contrasting with the current situation of hosting all project-related information on a centralised CDE, the federated approach acknowledges that the boundaries of project information are, in fact, volatile depending on the (project-specific or contextual) data required for a particular activity, as is the list of people and

organisations contributing to the overall knowledge base. When information on the Web is pointed at instead of copied into a centralised CDE, duplication is avoided, reinforcing the Single Source of Information (Hypothesis 4.4).

The separation of data and functionality and the use of federation technologies allow information from different sources (R1) to be combined by a specific service to address a well-defined purpose. This holds for AECO-related topics and adjacent disciplines. To make the data as reusable (R3) as possible, domain-agnostic data storage patterns (e.g. based on Semantic Web technologies) provide access to asset data to authorised agents (R2) for the entire life cycle of the building. The existence of general-purpose query languages allows higher-level components (services) in the ecosystem to request and aggregate (R8) the data they need and possibly adapt (R7) it to be compatible with discipline-specific or generic standards and activities (Hypothesis 4.5).

A combination of these requirements gives the end users freedom of choice to host their data on federated data vaults without limiting them in the tools to interact with this project data. The levels on which adapters and applications can communicate can range from standardised APIs - such as buildingSMARTs openCDE APIs - to more granular access via query languages while working with the same pluripotent data and allowing flexible project-dependent workflows. Thus, this framework could also show a possible way towards the BIM Maturity Stage 3 as defined in ISO 19650.

The DIN SPEC 91390 implies that all CDE features need to be accumulated on one platform. With the federated framework, it is possible to outsource the CDE functionalities to specialised applications that still comply with the requirements of the DIN SPEC 91391 and the ISO 19650.

To test the generic approach, we focused on addressing challenges related to the specific case of Issue Management. Although Issue Management is only one application of a CDE, it faces the same problems and challenges as other CDE processes, such as communication between stakeholders, resource creation and updating and cross-resource referencing. Especially in communication - the main focus of Issue Management - there is often the need for certainty about who said what. It was shown how self-created communication can be kept in one's hands by storing it in personal vaults. In addition, we demonstrated how standardised well-accepted interfaces such as the BCF API could be used in a federated manner without having to adjust the standard. This confirmed that these communication protocols have no bias regarding whether the infrastructure is centralised or federated, so consistency with current-day workflows and existing industry applications is kept.

The selected functionalities of the proposed framework - the use of the BCF API for the interface layer and Solid Pods for the data vaults - have been documented in the proof of concept. It proves the feasibility of implementing the discussed requirements using current technologies and the concept's compatibility with current-day workflows in the industry.

While the results are promising, the federated approach also has its limitations which are to be investigated in future research. Even if external users cannot modify the data of another vault, theoretically, nothing (except technical know-how) prevents the owner of a vault from modifying data in his or her favour. To prevent this, a relationship of trust between the stakeholders must be established, which can be solved technologically, for example, by digitally signing the data. Solutions for this already exist [Carroll, 2003, Sayers and Karp, 2004, Tummarello et al., 2005] and must be combined with the approach presented here. In many cases, there is an additional need to ensure that data from a single vault can be kept accessible or recoverable in the event of a failure - intentional or unintentional - of a single member of the federated chain. On the one hand, this issue is conceptually covered by the aggregator (R8), although this feature was not technically implemented in the Proof of Concept. On the other hand, the various stakeholders must be contractually obligated to guarantee the backup of data in the event of such a failure. It should be noted, however, that such failures can also occur in a centralised system, where such an incident could be even more severe since only individual members would likely be affected in a federated ecosystem. Furthermore, there is the general question of archiving data for the life cycle of a building. It is unlikely that all individual stakeholders - often small offices - can provide the vaults for the projects over several years to decades - which is the case with centralised CDEs as well. Thus, a data handover to an entity such as the owner, the government or specialised data storage companies would likely have to happen at some point for the data to remain available in subsequent phases of the building life cycle.

4.5.7 Conclusion

In this paper, we have analysed the criteria for implementing federated Issue Management, motivated by three initial hypotheses. These criteria were initially presented in a technology-agnostic way. A proof of concept then related these requirements to current-day technologies and applied it to an exemplary functionality of CDEs, namely Issue Management.

Our research has followed the guidelines of design science research [Hevner et al., 2004]. We started analysing what problems are associated with tools used in the

AECO industry today, why the centralised approach is not optimal, and what the challenges are when creating a federated CDE. A relevant question is also how to concretise the vision of the new ISO 19650 Maturity Stage 3. We listed requirements for a desirable solution and have proposed a federated framework that reflects them. The solution uses open standards and can be queried to provide an aggregated view, i.e., a Single Source of Information. The intention of this work is not to replace current standards - such as the openCDE APIs - but to complement them. We analysed the design in comparison with the central approaches. In addition, our design was assessed using the activity of Issue Management, and the results are verifiable: our source code is available for inspection. It was shown that the implementation of the system is viable, the lessons learned were documented, and we discussed the findings.

4.5.8 Acknowledgements

This research is funded by the Research Foundation Flanders (FWO) as a Strategic Basic Research grant (grant no. 1S99020N) and by the EU through the H2020 project BIM4REN grant (grant no. 820773).

References

Rajat Agarwal, Shankar Chandrasekaran, and Mukund Sridhar. Imagining construction's digital future. *McKinsey & Company*, 24, 2016. https://www.mckinsey.com/capabilities/operations/our-insights/imagining-constructions-digital-future, Online; accessed Oct. 15. 2023.

Riccardo Albertoni, David Browning, Simon Cox, Alejandra Gonzales Beltran, Andrea Perego, and Peter Winstanley. Data catalog vocabulary (DCAT) - version 3. W3C recommendation, W3C, 2023. https://www.w3.org/TR/vocab-dcat-3, Online; accessed Oct. 15. 2023.

Dean Allemang, Jim Hendler, and Fabien Gandon. *Semantic Web for the Working Ontologist: Effective Modeling for Linked Data, RDFS, and OWL*, volume 33. Association for Computing Machinery, New York, NY, USA, 3 edition, 2020. ISBN 9781450376174. https://doi.org/10.1145/3382097.

Autodesk Inc. Autodesk platform services, 2023. https://aps.autodesk.com/, Online; accessed Oct. 15. 2023.

David Becket, Tim Berners-Lee, Prud'hommeaux Erix, and Gavin Carothers. RDF 1.1 Turtle - Terse RDF Triple Language. W3C recommendation, W3C, 2014. https://www.w3.org/TR/turtle/, Online; accessed Oct. 15. 2023.

Jakob Beetz. *Facilitating distributed collaboration in the AEC/FM sector using Semantic Web Technologies*. PhD thesis, Built Environment, 2009. https://doi.org/10.6100/IR652808.

Jakob Beetz, Léon van Berlo, Ruben de Laat, and Pim van den Helm. Bimserver.org–an open source ifc model server. In *Proceedings of the CIP W78 conference*, page 8, 2010.

Jakob Beetz, André Borrmann, and Matthias Weise. Process-based definition of model content. In André Borrmann, Markus König, Christian Koch, and Jakob Beetz, editors, *Building Information Modeling: Technology Foundations and Industry Practice*, pages 127–138. Springer International Publishing, 2018.

ISBN 978-3-319-92862-3. doi: 10.1007/978-3-319-92862-3_6. https://doi.org/10.1007/978-3-319-92862-3_6.

Tim Berners-Lee. Linked data, 2006. URL https://www.w3.org/DesignIssues/LinkedData. Online; accessed Oct. 15. 2023.

Tim Berners-Lee. 5-star deployment scheme for open data. http://5stardata.info/, 2012. Accessed: 2022-11-17.

Tim Berners-Lee, James Hendler, Ora Lassila, et al. The Semantic Web. *Scientific American, Springer Nature*, 284(5):28–37, 2001. http://doi.org/10.1038/scientificamerican0501-34.

Andre Borrmann, Markus König, Christian Koch, and Jakob Beetz. Building information modeling: Why? what? how? In *Building Information Modeling - Technology Foundations and Industry Practice*, pages 1–24. Springer International Publishing, September 2018. ISBN 978-3-319-92861-6. http://doi.org/10.1007/978-3-319-92862-3_15.

Matthieu Bosquet. Access Control Policy. Protocol, W3C Solid Community Group, 2022. https://solid.github.io/authorization-panel/acp-specification/, Online; accessed Oct. 15. 2023.

Box, Inc. The information economy: A study of five industries. Technical report, Box, Inc., 2014. https://img.forconstructionpros.com/files/base/acbm/fcp/document/2014/06/box-cloud-study_11535206.pdf, Online; accessed Oct. 15. 2023.

BSI. Pas 1192-2: 2013: Specification for information management for the capital/delivery phase of construction projects using building information modelling, 2013.

buildingSMART Internationa. BIM Collaboration Format (BCF). https://technical.buildingsmart.org/standards/bcf/, 2023. Online; accessed Oct. 15. 2023.

buildingSMART International. Documents api, 2022. https://github.com/buildingSMART/documents-API, Online; accessed Oct. 15. 2023.

buildingSMART International Ltd. Technical roadmap buildingSMART getting ready for the future. 2022-12-07, 2020. URL https://www.buildingsmart.org/about/technical-roadmap/. Online; accessed Oct. 15. 2023.

Sarven Capadisli and Tim Berners-Lee. Web Access Control. Protocol, W3C Solid Community Group, 2022. https://solid.github.io/web-access-control-spec/, Online; accessed Oct. 15. 2023.

Sarven Capadisli, Tim Berners-Lee, Ruben Verborgh, and Kjetil Kjernsmo. Solid Protocol. Protocol, W3C Solid Community Group, 2022. https://solidproject.org/TR/protocol, Online; accessed Oct. 15. 2023.

Jeremy J Carroll. Signing rdf graphs. In *International Semantic Web Conference*, pages 369–384. Springer, 2003. https://doi.org/10.1007/978-3-540-39718-2_24.

Aaron Coburn, Elf Pavlik, and Dmitri Zagidulin. Solid-oidc. Specification, Solid, August 2022. https://solidproject.org/TR/oidc, Online; accessed Oct. 15. 2023.

European Commission, Directorate-General for Research, and Innovation. *Turning FAIR into reality: final report and action plan from the European Commission expert group on FAIR data*. Publications Office, 2018. https://data.europa.eu/doi/10.2777/1524.

Edward Corry, James ODonnell, Edward Curry, Daniel Coakley, Pieter Pauwels, and Marcus Keane. Using semantic web technologies to access soft aec data. *Advanced Engineering Informatics*, 28(4):370–380, 2014. https://doi.org/10.1016/j.aei.2014.05.002, Elsevier.

Edward Curry, James ODonnell, Edward Corry, Souleiman Hasan, Marcus Keane, and Seán ORiain. Linking building data in the cloud: Integrating cross-domain building data using linked data. *Advanced Engineering Informatics, Elsevier*, 27 (2):206–219, 2013. https://doi.org/10.1016/j.aei.2012.10.003.

Ruben Dedecker, Wout Slabbinck, Jesse Wright, Patrick Hochstenbach, Pieter Colpaert, and Ruben Verborgh. Whats in a pod?–a knowledge graph interpretation for the solid ecosystem. In *6th Workshop on Storing, Querying and Benchmarking Knowledge Graphs (QuWeDa) at ISWC 2022*, pages 81–96, 2022.

Deutsches Institut für Normung. DIN SPEC 91391-1:2019-04 Common Data Environments (CDE) for BIM projects - Function sets and open data exchange between platforms of different vendors - Part 1: Components and function sets of a CDE; with digital attachment, 2019a. https://dx.doi.org/10.31030/3044838.

Deutsches Institut für Normung. DIN SPEC 91391-1:2019-04 Common Data Environments (CDE) for BIM projects - Function sets and open data exchange between platforms of different vendors - part 2: Open data exchange with common data environmentst, 2019b. https://dx.doi.org/10.31030/3044839.

Roy Fielding, Jim Gettys, Jeffrey Mogul, Henrik Frystyk, Larry Masinter, Paul Leach, and Tim Berners-Lee. Hypertext transfer protocol–http/1.1. https://www.w3.org/Protocols/rfc2616/rfc2616.html, 1999. Online; accessed Oct. 17. 2023.

Roy Thomas Fielding. *Architectural styles and the design of network-based software architectures*. University of California, Irvine, 2000.

GO FAIR. Fair principles, 2016. https://www.go-fair.org/fair-principles/, Online; accessed Oct. 15. 2023.

Bjoern Godager, Erling Onstein, and Lizhen Huang. The concept of enterprise BIM: Current research practice and future trends. *IEEE Access*, Volume 9, 2021. http://dx.doi.org/10.1109/ACCESS.2021.3065116.

Bjoern Godager, Knud Mohn, Christoph Merschbrock, Ole Jonny Klakegg, and Lizhen Huang. Towards an improved framework for enterprise BIM: the role of ISO 19650. *Journal of Information Technology in Construction*, Volume 27: 1075–1103, 2022. ISSN 1874-4753. URL https://www.itcon.org/paper/2022/53. http://dx.doi.org/10.36680/j.itcon.2022.053, Online; accessed Oct. 15. 2023.

Mario Gürtler, Ken Baumgärtel, and Raimar J. Scherer. Towards a Workflow-Driven Multi-model BIM Collaboration Platform. In Luis M. Camarinha-Matos, Frédérick Bénaben, and Willy Picard, editors, *Risks and Resilience of Collaborative Networks*, IFIP Advances in Information and Communication Technology, pages 235–242, Cham, 2015. Springer International Publishing. ISBN 978-3-319-24141-8. http://doi.org/10.1007/978-3-319-24141-8_21.

Philipp Hagedorn, Liu Liu, Markus König, Rade Hajdin, Tim Blumenfeld, Markus Stöckner, Maximilian Billmaier, Karl Grossauer, and Kenneth Gavin. Bim-enabled infrastructure asset management using information containers and semantic web. *Journal of Computing in Civil Engineering*, 37(1):04022041, 2023.

Steve Harris, Andy Seaborne, and Eric Prud'hommeaux. SPARQL 1.1 Query Language. W3C recommendation, W3C, 2013. https://www.w3.org/TR/sparql11-query/, Online; accessed Oct. 15. 2023.

HH JUDGE EYRE QC. Mott macdonald ltd v trant engineering ltd [2021] EWHC 754 (TCC). https://www.bailii.org/ew/cases/EWHC/TCC/2021/754.html, 2021. Online; accessed Oct. 15. 2023.

Tim-Jonathan Huyeng, Christian-Dominik Thiele, Anna Wagner, Meiling Shi, André Hoffmann, Wendelin Sprenger, and Uwe Rüppel. An approach to process geometric and semantic information as open graph-based description using a microservice architecture on the example of structural data. In *EG-ICE 2020 Workshop on Intelligent Computing in Engineering*, Berlin, June 2020. Universitätsverlag der TU Berlin. URL http://tubiblio.ulb.tu-darmstadt.de/121623/. http://dx.doi.org/10.14279/depositonce-9977, Online; accessed Oct. 15. 2023.

International Organization for Standardization. ISO 29481-1: 2010 (E): Building information modelingInformation Delivery Manual. Standard, International Organization for Standardization, 2010.

International Organization for Standardization. ISO 19650-1:2018: Organization and digitization of information about buildings and civil engineering works, including building information modelling (BIM) Information management using building information modelling Part 1: Concepts and principles. Standard, International Organization for Standardization, Geneva, CH, 12 2018a. URL https://www.iso.org/cms/render/live/en/sites/isoorg/contents/data/standard/06/80/68078.html. Online; accessed Oct. 15. 2023.

International Organization for Standardization. ISO 16739-1:2018: Industry Foundation Classes (IFC) for data sharing in the construction and facility management industries. Standard, International Organization for Standardization, 2018b.

International Organization for Standardization. Information container for linked document delivery . Standard, International Organization for Standardization, 2020.

Janakiram Karlapudi, Prathap Valluru, and Karsten 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*, pages 76–86. Universitätsverlag der TU Berlin, 2021.

Fabian Kirstein, Benjamin Dittwald, Simon Dutkowski, Yury Glikman, Sonja Schimmler, and Manfred Hauswirth. Linked data in the european data portal: a comprehensive platform for applying dcat-ap. In *International Conference on Electronic Government*, pages 192–204. Springer, 2019. http://dx.doi.org/10.1007/978-3-030-27325-5_15.

Arto Kiviniemi, Martin Fischer, and Vladimir Bazjanac. Integration of multiple product models: Ifc model servers as a potential solution. In *Proc. of the 22nd CIB-W78 Conference on Information Technology in Construction*, 2005.

Graham Klyne, Jeremy J. Carroll, and Brian McBride. RDF 1.1 Concepts and Abstract Syntax. W3C recommendation, W3C, 2014. https://www.w3.org/TR/rdf11-concepts/, Online; accessed Oct. 15. 2023.

James Lewis and Martin Fowler. Microservices: a definition of this new architectural term, 2014. URL http://martinfowler.com/articles/microservices.html. Online; accessed Oct. 15. 2023.

Essam Mansour, Andrei Vlad Sambra, Sandro Hawke, Maged Zereba, Sarven Capadisli, Abdurrahman Ghanem, Ashraf Aboulnaga, and Tim Berners-Lee. A

demonstration of the solid platform for social web applications. In *Proceedings of the 25th International Conference Companion on World Wide Web*, pages 223–226, 2016. http://doi.org/10.1145/2872518.2890529.

Wiet Mazairac and Jakob Beetz. BIMQL an open query language for building information models. *Advanced Engineering Informatics*, Volume 27(4):444–456, 2013. ISSN 1474-0346. URL https://www.sciencedirect.com/science/article/pii/S1474034613000657. http://doi.org/10.1016/j.aei.2013.06.001, Online; accessed Oct. 15. 2023.

Barend Mons, Cameron Neylon, Jan Velterop, Michel Dumontier, Luiz Olavo Bonino da Silva Santos, and Mark D Wilkinson. Cloudy, increasingly fair; revisiting the fair data guiding principles for the european open science cloud. *Information services & use*, 37(1):49–56, 2017. http://doi.org/10.3233/ISU-170824, IOS Press.

Jyrki Oraskari, Oliver Schulz, and Jakob Beetz. Towards describing version history of BCF data in the semantic web. In *Proceedings of the 10th Linked Data in Architecture and Construction Workshop co-located with 19th European Semantic Web Conference (ESWC 2022). CEUR-WS Team, Redaktion Sun SITE, Informatik V, RWTH Aachen, Germany*. CEUR-WS, 2022a. https://ceur-ws.org/Vol-3213/paper08.pdf. Accessed November 8, 2024.

Jyrki Oraskari, Oliver Schulz, Jeroen Werbrouck, and Jakob Beetz. Enabling federated interoperable issue management in a building and construction sector. In *Proceedings of the 29th EG-ICE International Workshop on Intelligent Computing in Engineering. Aarhus University Library/Det Kgl. Bibliotek - AU Library Scholarly Publishing Services. Aarhus, Denmark*, pages 92–101. EG-ICE, 2022b. ISBN 978-87-7507-521-8. URL https://ebooks.au.dk/aul/catalog/view/455/312/1848-2. http://doi.org/10.7146/aul.455.c200, Online; accessed Oct. 15. 2023.

Pieter. Pauwels and Walter Terkaj. EXPRESS to OWL for Construction Industry: Towards a Recommendable and Usable ifcOWL Ontology. *Automation in Construction*, 63:100–133, 2016. 10.1016/j.autcon.2015.12.003, Elsevier.

Pieter Pauwels, Sijie Zhang, and Yong-Cheol Lee. Semantic web technologies in AEC industry: A literature overview. *Automation in Construction*, 73:145–165, January 2017. ISSN 09265805. URL https://linkinghub.elsevier.com/retrieve/pii/S0926580516302928. http://doi.org/10.1016/j.autcon.2016.10.003, Online; accessed Oct. 15. 2023.

Cornelius Preidel, Andre Borrmann, Hannah Mattern, Markus König, and Sven-Eric Schapke. Common Data Environment. In *Building Information Modeling - Technology Foundations and Industry Practice*, pages 279–291. Springer

International Publishing, September 2018. ISBN 978-3-319-92861-6. URL http://doi.org/10.1007/978-3-319-92862-3_15.

Mads Holten Rasmussen, Maxime Lefrançois, Georg Ferdinand Schneider, and Pieter Pauwels. Bot: the building topology ontology of the w3c linked building data group. *Semantic Web*, 12(1):143–161, 2021. http://doi.org/10.3233/SW-200385,IOS Press.

Rafael Sacks, Zijian Wang, Boyuan Ouyang, Duygu Utkucu, and Siyu Chen. Toward artificially intelligent cloud-based building information modelling for collaborative multidisciplinary design. *Advanced Engineering Informatics*, 53: 101711, 2022. https://doi.org/10.1016/j.aei.2022.101711, Elsevier.

Natsuhiko Sakimura, John Bradley, Mike Jones, Breno De Medeiros, and Chuck Mortimore. Openid connect core 1.0, 2014. URL https://openid.net/specs/openid-connect-core-1_0.html. Online; accessed Oct. 15. 2023.

Craig Sayers and Alan H Karp. Computing the digest of an rdf graph. *Mobile and Media Systems Laboratory, HP Laboratories, Palo Alto, USA, Tech. Rep. HPL-2003-235*, 1, 2004.

Oliver Schulz and Jakob Beetz. Image-documentation of existing buildings using a server-based BIM collaboration format workflow. In *EG-ICE 2021 Workshop on Intelligent Computing in Engineering*, 2021. https://doi.org/10.14279/depositonce-12021.

Oliver Schulz, Jyrki Oraskari, and Jakob Beetz. bcfOWL: A BIM collaboration ontology. In *Proceedings of the 9th Linked Data in Architecture and Construction Workshop*, Luxembourg, October 2021. URL https://linkedbuildingdata.net/ldac2021/files/papers/CIB_W78_2021_paper_122.pdf. Online; accessed Oct. 15. 2023.

Mikki Seidenschnur, Ali Kücükavci, Esben Visby Fjerbæk, Kevin Michael Smith, Pieter Pauwels, and Christian Anker Hviid. A common data environment for hvac design and engineering. *Automation in Construction, Elsevier*, 142:104500, 2022. https://doi.org/10.1016/j.autcon.2022.104500.

Madhumitha Senthilvel, Jyrki Oraskari, and Jakob Beetz. Common data environments for the information container for linked document delivery. In *Proceedings of the 8th Linked Data in Architecture and Construction Workshop*. CEUR-WS. org, 2020. URL https://ceur-ws.org/Vol-2636/10paper.pdf. Online. Accessed Oct. 20. 2023.

Solid. Web access control spec. Specification, Solid, july 2022. https://solidproject.org/TR/wac,Online; accessed Oct. 15. 2023.

Steve Speicher, John Arwe, and Ashok Malhotra. Linked Data Platform 1.0. W3C recommendation, W3C, 2015. https://www.w3.org/TR/ldp/, Online; accessed Oct. 15. 2023.

Xingyu Tao, Moumita Das, Yuhan Liu, and Jack C. P. Cheng. Distributed common data environment using blockchain and Interplanetary File System for secure BIM-based collaborative design. *Automation in Construction*, 130, October 2021. ISSN 0926-5805. URL https://www.sciencedirect.com/science/article/pii/S0926580521003022. http://doi.org/10.1016/j.autcon.2021.103851, Online; accessed Oct. 15. 2023.

Trimble Inc. Trimble connect api, 2021. https://connect.trimble.com/, Online; accessed Oct. 15. 2023.

Giovanni Tummarello, Christian Morbidoni, Paolo Puliti, and Francesco Piazza. Signing individual fragments of an rdf graph. In *Special interest tracks and posters of the 14th international conference on World Wide Web*, pages 1020–1021, 2005. https://doi.org/10.1145/1062745.1062848.

Léon van Berlo and Ruben de Laat. The bimbots protocol, Feb 2018. URL https://github.com/opensourceBIM/BIM-Bot-services/wiki. Online; accessed Oct. 15. 2023.

Léon van Berlo and Thomas Krijnen. Using the BIM Collaboration Format in a Server Based Workflow. *Procedia Environmental Sciences*, 22:325–332, January 2014. ISSN 1878-0296. URL http://www.sciencedirect.com/science/article/pii/S1878029614001789. http://doi.org/10.1016/j.proenv.2014.11.031, Online; accessed Oct. 15. 2023.

Léon van Berlo, Mathijs Natrop, and Gerwin Korpershoek. Collaborative engineering with IFC and BCF: Managing openings and recesses. In *Lean and Computing in Construction Congress (LC3): Volume I: Proceedings of the Joint Conference on Computing in Construction (JC3), July 4-7, 2017, Heraklion, Greece*, pages 877–884, 2017. http://doi.org/10.24928/JC3-2017/0194.

Joachim Van Herreweghe, Ruben Verborgh, Ruben Taelman, and et al. Community solid server, 2023. https://github.com/CommunitySolidServer/CommunitySolidServer, Online; accessed Oct. 15. 2023.

Ruben Verborgh and Miel Vander Sande. The Semantic Web identity crisis: in search of the trivialities that never were. *Semantic Web Journal*, 11(1): 19–27, 1 2020. URL https://ruben.verborgh.org/articles/the-semantic-web-identity-crisis/. http://doi.org/10.3233/SW-190372, IOS Press, Online; accessed Oct. 15. 2023.

buildingSMART Verein Deutscher Ingenieure. VDI 2552 blatt 11.2- building information modeling - exchange requirements - slots and openings, 2022.

Nam Vu Hoang and Seppo Törmä. DRUMBEAT Platform A Web of Building Data Implementation with Backlinking. In Symeon Christodoulou and Raimar Scherer, editors, *eWork and eBusiness in Architecture, Engineering and Construction: ECPPM 2016*. CRC Press, 2016. https://doi.org/10.1201/9781315386904.

Jeroen Werbrouck, Pieter Pauwels, Jakob Beetz, and Léon van Berlo. Towards a decentralised common data environment using linked building data and the solid ecosystem. In *Advances in ICT in Design, Construction and Management in Architecture, Engineering, Construction and Operations (AECO) : Proceedings of the 36th CIB W78 2019 Conference*, pages 113–123, 2019. URL http://hdl.handle.net/1854/LU-8633673. isbn: 978-1-86135-487-7 978-1-86135-486-0, Online; accessed Oct. 15. 2023.

Jeroen Werbrouck, Pieter Pauwels, Jakob Beetz, and Erik Mannens. Mapping federated aec projects to industry standards using dynamic views. In *Proceedings of the 10th Linked Data in Architecture and Construction Workshop co-located with 19th European Semantic Web Conference (ESWC 2022)*. CEUR-WS, 2022.

Jeroen Werbrouck, Pieter Pauwels, Jakob Beetz, and Erik Mannens. Consolid: a federated ecosystem for heterogeneous multi-stakeholder projects, 2023. URL http://www.semantic-web-journal.net/content/consolid-federated-ecosystem-heterogeneous-multi-stakeholder-projects. Online; accessed Oct. 15. 2023.

Mark D Wilkinson, Michel Dumontier, IJsbrand Jan Aalbersberg, Gabrielle Appleton, Myles Axton, Arie Baak, Niklas Blomberg, Jan-Willem Boiten, Luiz Bonino da Silva Santos, Philip E Bourne, et al. The fair guiding principles for scientific data management and stewardship. *Scientific data*, 3(1):1–9, 2016. https://doi.org/10.1038/sdata.2016.18, Nature Publishing Group.

Chi Zhang, Jakob Beetz, and de Vries. BimSPARQL: Domain-specific functional SPARQL extensions for querying RDF building data. *Semantic Web*, pages 1–27, August 2018. http://doi.org/10.3233/SW-180297.

4.6 Discussion and Conclusions

Common Data Environments (CDEs) play an important role in the management of construction data. In this chapter I have focused on how Linked Building Data can be published in a CDE. I have chosen two topics to do this. Firstly, how can Information Container for Document Delivery (ICDD) data be published in an open standard CDE? The second theme focuses on exploring how BCF data, which buildingSMART has positioned as one of its OpenCDE standards, can be implemented in a federated way.

The research question focused on understanding the benefits of using ICDD containers in open Common Data Environments(CDE) to facilitate the overall architecture of data storage, access and retrieval. In particular, the focus was on structuring and linking heterogeneous data at meta and sub-document levels. The research compared CDE implementation methods such as DIN SPEC 91391-2, OpenCDE-API, LDP frameworks and automated metadata import. In addition, the research looked at achieving a single source of truth for project data in a distributed setup, allowing decentralised authentication and stakeholder control of their data. A proposed solution was a tree-like structure combining BCF XML and BCF API features with a single point of access implemented using the distributed Solid architecture.

In the paper presented in section 4.2, we showed how ICDD metadata can be aligned with three open standards: the DIN SPEC 91391-2, the buildingSMART OpenCDE API, and the Linked Data Platform (LDP). To ensure that ICDD containers are not just files in the CDE, we created the metadata alignment to make the data more CDE native. The research in section 4.3 has shown that mapping and publishing are technically feasible. To address the limitation that the non-interactive part of the standard was still missing, an extension to the standard was proposed to reduce manual effort by automating metadata copying. The implementation decompresses the ICDD package and makes the files available online after publication. This is

the first step towards realising the principles of linked data, the ability to link and access data content from the web.

In our paper, we have discussed how ICDD can be published as a CDE and how a datapod-based architecture can provide a framework for a distributed LBD architecture. We have demonstrated the feasibility of the ideas with the artefacts. The higher level visions will allow the ideas to be implemented later as implementation techniques evolve and are replaced by new ones.

Sections 4.4 and 4.5 discuss, from an architectural point of view, how BCF data should be organised and published so that the data remains in the hands of its owner. The owner, by default, would be the AEC design office, but could also be a facilities manager after data transfer. There are two related issues: responsibility for the design and the CDE vision of a single source of truth (SSoT), which also means avoiding the need to enter the same information repeatedly, which is believed to reduce errors and bring more information to the table. In this article we have imagined how a personal secure web server where data is stored (data pod) could be the architectural solution sought. The idea could be generalised to CDE data in general. This is partly a continuation of the earlier LDP research in sections 4.2 and 4.3. While CDE and data pods are content agnostic, the proposed architecture provides a way to decentralise data and user authentication, data authorisation and, as we have shown, federated search. The latter is what enables SSoT in Linked Data environments. It is consistent with Gartner's view of data management as providing consistent access to data (see section 1.5).

Chapter 5

Information Security Aspects: Confidentiality

Data confidentiality is data protection that ensures that it is not disclosed without authorization or accessed by unauthorized parties [Alghamdi, 2017]. User authentication, data authorization, and secure data handling are the basis of data confidentiality.

In this chapter, I address the research question: How information security aspects of data management related to confidentiality can be handled.

Hypothesis 5.1 *When managing Linked Building Data, the data's availability, and confidentiality can be better secured than in the common file-sharing approach. The claim is addressed in the Sections 5.1.*

I will introduce what support I have for the claims and discuss arguments that are against them.

Figure 5.1 shows my article in this chapter at a glance.

Publication	Objectives	Method	Key findings
Oraskari & Törmä, Access control for web of building data: Challenges and directions., 2016	• to analyse the access control challenges of practical use cases • find access control criteria • explore a possible decentralised access control	The Design Science Research Method: Descriptive Scenarios	• the analysis of access control challenges of practical use cases • objectives for access control • possible decentralized access control solutions

Fig. 5.1. My article related to information security aspects of data management related to confidentiality at a glance

5.1 Access control for the web of building data: Challenges and directions (ECPPM 2016)

Image a situation in which a new air terminal has been installed. The facility manager wants to link the associated database element to the external product data. Access rights are needed.

The paper is two-sided. It starts with the requirements and roles in a secure Linked Building Data environment. Then, it ends with a review of the relevant authentication technologies.

Hypothesis 5.1 claimed that when managing Linked Building Data, the data's availability, confidentiality, and integrity can be better secured than the common file-sharing approach. First, the strongest argument about the security itself is that Linked Data is accessed granularly, allowing finer-grained access policies than the file-based approach. However, Common Data Environment (CDE) services are prevalent, in some cases data can be accessed in a granular manner (like in the Autodesk could), and usually also contain authorisation services (e.g., in the OpenCDE specification, the OAuth2 as a part of OpenCDE Foundation API[1]).

[1] https://github.com/buildingSMART/foundation-API

Although we have presented WebID and its TLS extension as a potential time-lasting solution for an independent project or organisation-wide authorisation, OAuth2 is the industry-standard protocol to allow selected sites to access resources on behalf of the user. While OAuth2 (like in a Facebook login) allows the user to select the authenticator, the services are not usually possible to be self-hosted as in the WebID context. Furthermore, when using Linked Data, one weak argument is that non-WebID solutions are not RDF-based.

In the use-case, the facility manager could use a WebId (See Sections 5.1.5, and 4.4.2) as the user identification and use his or her own server as an identity provider hosting the WebId profile. The *Web Access Control system*[2] could be used to express access rights for the product data (the *Solid platform specification*[3] replaces the Web Access Control system and is now the preferred solution). Then, a Solid server[4] installation could enforce the authorisation. The facility manager needs rights assigned by the product catalogue owner so that he can access the product data when following the links or writing federated SPARQL queries.

Table 5.1 summarises the use of the design science research method in this research.

My contribution to the paper: I wrote a major part of the article. My focus was on analysing the objectives of access control and the solutions. D.Sc. (Tech) Seppo Törmä supervised the work.

Table 5.1. The work of Oraskari and Törmä [2016a] in the Hevner et al. [2004] design science research framework

[2] https://www.w3.org/wiki/WebAccessControl
[3] https://github.com/solid/web-access-control-spec
[4] https://solidproject.org/

Guideline 1: Design as an artifact

This paper introduces the DRUMBEAT platform, the stakeholders and to be protected datasets connected with use cases that were studied in the project, objectives for the access control, and solution approaches. To understand how WebID works, I implemented a WebID X.509 and profile generator[5].

Guideline 2: Problem relevance

Companies that use BIM may have data that should be protected for copyrights, and company secrets, are under a non-disclosure agreement or should be protected against misuse. The same applies when using Linked Data. The associated business problem is finding clear data governance policies and techniques to enforce the rules.

The constituent user community of the artefact is the researchers of the Linked Building Data Community and software security architects of a Common Data Environment. The solution belongs to the Common Data Environment and access control problem classes. This work is an enabler for the Common Data Environment problem class, which explores ways to house construction project information to collect, manage, collaborate and share information on the organization of the artefacts.

Guideline 3: Design evaluation

Hevner et al. Design Research Evaluation Method: Descriptive Scenarios were used to evaluate the security needs.

[5] https://github.com/jyrkioraskari/weibd

Guideline 4: Research contributions

The works main contributions are the set of use cases, related actors, and the needed resources to protect. This allows for consideration of what to protect and to whom to give rights. If following the general security principle of least privilege [Saltzer and Schroeder, 1975] so that only necessary rights that are necessary for a legitimate purpose are granted to the users, the knowledge helps to design systems that do not allow extra rights for the users and so, the security of the services can be improved.

Guideline 5: Research rigor

The work was peer-reviewed.

Guideline 6: Design as a search process

The listed requirements at the beginning of the article were used to guide the design process.

Guideline 7: Communication of research

Proceedings of the 11th European Conference on Product and Process Modelling (ECPPM 2016), Limassol, Cyprus, 7-9 September 2016

The Author has retained the right to republish the Contribution in any printed collection consisting solely of the Authors own Works without charge and subject only to notifying the Publisher of the intent to do so and ensuring that the publication by the Publisher is properly credited and that the relevant copyright notice is repeated verbatim. The publisher Taylor Francis Groups has been notified and they have affirmed the use as long as the following text is added:

This is an Accepted Manuscript of an article published by Taylor & Francis in eWork and eBusiness in Architecture, Engineering and Construction: ECPPM 2016 on 28 March 2017, available online: https://www.taylorfrancis.com/chapters/edit/10.1201/9781315386904-10/access-control-web-building-data-challenges-directions-oraskari-t%C3%B6rm%C3%A4.

The work was originally published in [Oraskari and Törmä, 2016a]. The content presented here is based on our final draft of the article.

Jyrki Oraskari School of Science, Aalto University, Espoo, Finland,
Seppo Törmä School of Science, Aalto University, Espoo, Finland,

Abstract: Web of Building Data enables open publication of building-related data on the Web. However, a large part of building data requires access control for reasons of facility security, resident privacy, competition, and IPR protection. The linking across multiple models at different hosts creates challenges to implement an access control scheme that could avoid repeated and tedious registrations and authentications by users, and enable simple definition of the access control rules at different hosts. In this study the access control challenges of practical use cases are analyzed, objectives for access control are summarized, and possible decentralized access control solutions are explored. The focus is on WebId+TLS and OpenId Connect. The importance of common ontologies for role or property based access control is discussed

5.1.1 Introduction

Over the lifecycle of a building, a large set of parties produce information related to the building and its environment. The amount and variety of this information is growing due to the ongoing digitalization of society and the construction industry. The potentially extremely valuable information remains underutilized: it is fragmented and managed in a disconnected manner, which makes it difficult to access and combine with other relevant information. (Törmä 2014)

Web of Building Data is a set of technologies for the publication of building-

related data on the Web. It enables online access to an up-to-date version of data in a granular fashion. Each building element has a Web address and can be accessed individually. Using their addresses, building elements and other objects related to them can also be linked with each other across different hosts and models. There are many use cases where these capabilities are valuable, and the increasing openness of data enables efficiency improvements and new innovative applications.

The publication of building data online on the Web creates a very different security environment than the traditional way of exchanging building models between parties, either point-to-point or through a centralized project repository. From a security perspective, Web of Building Data has both advantages and disadvantages. A major advantage is that since building objects can be accessed in a granular manner – one object at a time – it is not necessary in many use cases to transfer complete models to other parties. Transferring a complete model is risky: it requires trust that the receiving party does not misuse the model, will keep it confidential, and store and manage it scrupulously to avoid security breaches. In contrast, the granular transfer of data enabled by Web of Building Data can be authenticated, controlled, and monitored to alleviate confidentiality problems. Its disadvantage from the security perspective is the open nature of the Web: all published data is by default accessible to all Web users. If confidentiality is required, access to data needs to be controlled by proper authentication and authorization mechanisms.

Probably only a minor portion of all building data can be completely open for everyone to access; most of it can only be published to a restricted audience. Confidentiality requirements spring from a variety of sources: the physical security of a building, the privacy of building users, the protection of the intellectual property of designers, the wish to hide operational data from competitors, and the desire to avoid the embarrassment due to the poor quality of data.

The focus of this paper is to study the challenges of access control for Web of Building Data. What are the categories of stakeholders, what are their roles in publishing and accessing data, how sensitive is the published data, and what kinds of threats would the platform be faced with? From what directions could the solutions to the challenges be found?

To answer the above-mentioned questions, we will first analyze a set of use cases for Web of Building Data in the areas of a simple model publication, linking of product data, status sharing, and cross-model linking. The stakeholders, their roles in publishing, managing and accessing data, and types and sensitivity of data contents in each case are described, and relevant challenges are identified.

The paper discusses possible approaches to address the security challenges. There is an obvious need to balance the safety concerns with the flexibility and simplicity of the overall security design. As an example, the nature of the platform would suggest properties such as distributed identity management and single sign-on; the combination of these two requirements will constrain the options for the platform implementation.

In the remainder of the paper, we will first review the concept of Web of Building Data and its implementation in the DRUMBEAT platform, analyze the security challenges with use cases, and then explain the solution approaches. After the discussion, we end with concluding remarks.

5.1.2 Web of Building Data and DRUMBEAT platform

Web of Building Data (WoBD) is a set of technologies for *publication* of building-related data on the Web, instead of *exchanging* it as files between project partners (Törmä 2014). Like the ordinary Web, it is a decentralized and open framework, which can be used by any party as it chooses. WoBD can support information sharing among an extended set of stakeholders over the whole lifecycle of a building.

Web of Data

Web of Data is a term that refers to the publication of data on the Web in a way that data objects can be first-class citizens: they have a Web address, can be individually accessed, and can be linked with each other. Web of Data consists of basic Web technologies (URI, HTTP), Semantic Web representations and technologies (RDF, Turtle, JSON-LD, RDFS, OWL, etc.), and Linked Data principles (Bizer et al. 2009) that cover the publication and access of data.

The Web of Data approach for data publication can be summarized as following:

- *Identifiers:* URIs (Uniform Resource Identifier) (Berners-Lee 1998) are used as object identifiers.
- *Data:* Data is represented in RDF (Resource Description Framework) (Lassila & Swick 1999), a graph-based representation framework.
- *Concepts:* Conceptual models are ontologies defined typically in OWL (Web Ontology Language) (Dean et al. 2004) or in simple cases with RDFS (Brickley & Guha 2014) or SKOS (Miles & Bechhofer 2009). The same dataset can be

partially described by several different ontologies.
- *Serializations:* Data can be serialized in standard formats (e.g., rdfXML, Turtle, JSON-LD) to allow tool chain interoperability and to avoid vendor lock-in (Carroll & Stickler 2004).
- *Access:* Objects are accessed with their URIs over the HTTP protocol (Fielding & Reschke 2014). The description of the immediate RDF graph neighbourhood of an object is returned in a suitable RDF serialization. The description can contain further links to other related objects.

Web of Building Data

All major BIM authoring tools can export BIM models in the standard IFC format (Industry Foundation Classes) (ISO 2013). In WoBD approach the exported IFC files can be converted into RDF graphs, so-called ifcRDF, which uses the concepts, properties types defined in the ifcOWL ontology (Pauwels et al. 2016).

The ifcRDF models are then published on the Web so that the descriptions of objects are retrievable using the HTTP protocol. The returned descriptions are serializations of the immediate neighbourhood of an object in the RDF graph.

ifcOWL is just one ontology in which building-related data can be published. Other ontologies based on existing standards could be used – for instance, COBie, CityGML, or InfraGML – and new ontologies can be created for specific use cases, such as status sharing or cross-model linking.

DRUMBEAT platform

DRUMBEAT platform is an implementation of the Web of Building Data concept, developed at Aalto University. It contains an IFC to RDF converter called IFC2LD (Hoang & Törmä 2015), and allows the publication of models on the Web.

In order to make Web of Building Data a practical approach, the DRUMBEAT platform contains several additional mechanisms:

- *Organization of data*: The published data is organized into collections, models, and datasets. These concepts are defined in a metadata ontology LBDHO (Linked Building Data Host Ontology).
- *REST interface:* The creation of the metadata objects (collections, models, datasets) are done through a REST API (Fielding 2000). IFC models that are

uploaded into datasets are automatically converted into RDF.

- *Backlinking:* To make the published building data linked into two directions, the platform relies on backlinking functionality: whenever a link to a remote object is created a notification to the remote server is sent, and the remote server can create a backlink, that is, a remote inverse link.
- *Hosting and accessing links:* Each party hosts the *outgoing* links of all the objects that it hosts. When an object is accessed, the outgoing links are included in the returned description. There is thus no need for separate access to links.

It should be noted that backlinking is an essential function in some of the use cases described below.

Deployment

Web of Building Data is meant to be deployed in a completely decentralized manner. The basic architecture does not require any centralized components, apart from common ontologies. Each party that produces building data, publishes it on its own and manages the data independently of others. The outgoing links from data are published in the same way as other data. The links can be either embedded in models or published as separate link models, depending on whether the lifecycle of links is the same or different from that of the data.

Using the DRUMBEAT platform software, the options to publish building data are similar to the publication of Web pages. A company can have its own Web servers and run the DRUMBEAT software to publish data. If that is too complicated, it can rent a server capacity from a cloud to run the platform, or there could be external service providers that run the platform and provide domains for users to publish their building data.

Applications can access the data on the DRUMBEAT platform through a REST API. They can follow the links from one server to another, in a similar manner to Web browsers do with HTML pages. They can also combine data from several different servers according to the needs of a use case.

One advantage of decentralized publishing is to maintain clear ownership and control of data. Each party can decide the access control rules for the data as it sees proper.

5.1.3 Use cases

In this section, a set of use cases studied in the DRUMBEAT project will be reviewed. Different use cases are related to various stages of the building lifecycle. Common to all examples is the distributed nature of information production and utilization.

Publication of a model

This is a simple use case where one party publishes a BIM model to allow other parties to access the objects in it or establish links to them. In the DRUMBEAT platform this means that an IFC file is uploaded and the platform converts it into ifcRDF (in which the concepts that property types refer to the ifcOWL ontology), creates URIs for all entities with a GUID, and makes these objects retrievable over the HTTP.

Table 5.2. Table 1 - Model publication use case

Stakeholders	Models/datasets
BIM model author	BIM model
BIM model users	

Some obvious requirements arise from this simple scenario: the model author needs to have the rights to add and modify content on the platform and no authorized parties should not have those rights. To ensure the integrity of data, modification rights cannot be given awarded lightly; the model author can be expected to have the expertise and tools to produce coherent data and be able to take responsibility of it. In the worst case, the data would be destroyed by a malicious Internet bot.

The access to a published model is typically given to many users in addition to the publisher. Most of these users cannot have the rights to modify the content but they need to have the right to read it. There will thus be users with different access rights.

Most of the access to building data is instrumental. The need to access data about a building is more based on the role and properties of a user and her relation to

the building, than on the preferences or identity of a user. Access rights can be associated to users with the following type of expressions:

- “The site manager of project-12“
- “All employees of the subcontractors of company-23”
- “All employees of company-x working in project-12”
- “The property manager of facility-34”
- “All residents of building-45”

The access control rules, therefore, refer to the relative roles of the user (architect, site manager, property manager, employee, resident) or companies (contractor, subcontractor), and certain individuals (project-12, company-23, facility-34, building-45). The identifiers of individuals can naturally be URIs.

In summary, the challenges arising from the model publication use case are:

- The producers and users of data need to be authenticated.
- Assignment of different access rights for different users must be supported.
- The access control rules should be based on the roles or properties of users with respect to certain individuals.

Cross-model linking

An object-level linking can be established between two BIM models. There are several examples:

1. *Structural-Architectural:* The precast elements in a structural model are linked to the walls and slabs in an architectural model.
2. *MEP-Architectural:* The terminals in a MEP model are linked to the spaces they serve in an architectural model.
3. *MEP-Structural:* The MEP objects colliding with structural objects are linked to each other in the form of clashes to be resolved.

Link generation could take place in BIM authoring tool during the reference model-based design (between the objects in the reference model and the authored model), or after the design work in a cross-model checking system such as Solibri Model Checker.

Table 5.3. Table 2 – Cross-model linking use case

Stakeholders	Models/datasets
BIM model author	BIM models
Contractors	Link models
BIM model users	

Once the BIM models have been created and the linking between them generated, links can be used as channels to propagate information across models. For instance, the architect could get information about the status of rooms indirectly through the structural model or MEP model. Status information is important when the possibility of further changes is evaluated.

In a change situation, linking can be used to provide notifications to other parties to indicate how the change would affect them. When an object has changed – when its properties or relations have been modified – and it has a link to an object in another model, the latter object needs to be checked. A notification to the other model will give focused information about the likely impact of the change.

The challenges arising from the access control use case are the following:

- When following links between models, the user may need to be re-authenticated.
- The user should not be burdened by re-authentication requests.
- When following links between models, it should be possible to evaluate the access control rules in a coherent manner.
- When following links between models, if a user does not have read rights to the target object of a link, the link should not be presented as valid.
- It must be possible to publish link models without modification rights to the interlinked models.

Linking to product data

IFC models contain only rudimentary information about the actual products selected as implementations of particular building components. Most of the product information is represented in online product catalogues or databases, for instance, at product vendors' sites. Product data is a natural example of enriching IFC models with more detailed data. The building objects in IFC models can be

linked to appropriate product data, as long as that product data is accessible on the Web and identified with a URI. This is possible, for instance, with bsDD Data Dictionary and MagiCloud Product Library.

If needed, the linking to product data can be done without touching the original IFC model at all. In the Web of Data approach, the link can be embedded in the model but it can equally well be published in an external link model.

Some product data catalogues are commercial services that can be accessed only by subscription. Also, many freely accessible catalogues prefer a user registration. In any sufficiently complex model, there will be links to several different product libraries.

Table 5.4. Table 3 – Product data use case

Stakeholders	Models/datasets
BIM model author	BIM models
Product selector	Link models
Product library operators	Product libraries
Product vendors	Product vendors data
BIM model users	

BIM model authors create the models. During the design work, they can select generic or specific products as the implementation of components in the model. They associate the product information with components in their model, which means that a linking is created between the objects in the BIM model and the products in the product library. A product selector, typically the contractor, can later change the tentatively selected products, which means the creation of another linking that specializes or overrides the linking created by a designer. The earlier choices of the designer should not necessarily be accessible to all users, but for some users, they need to be preserved for later analyses.

The model users may need to access the BIM model together with detailed product information in a variety of use scenarios: to set up, operate, diagnose, fix, replace, or reuse a product. This presumes that the BIM models will be available to the model users, which naturally is a goal of BIM. However, if the product information is only linked to the BIM model and the information itself must be accessed from a commercial service, the access right should be delegated, or otherwise all model users need to be customers of that service, either through a subscription or an

access fee.

In summary, the additional access control challenges are:

- The access behaviour of a user may need to be monitored in order to avoid data hoarding.
- Depending on the business model, it should be possible to delegate access rights to product data to the users of a model.

Status sharing

During the execution of a construction project, all building objects go through a sequence of status changes as a result of activities performed on them by different parties. For example, a precast element can have the following sequence of states: designed, detailed, accepted to casting, casted, finalized, transported to the site, in site storage, installed, inspected.

The different activities causing status changes are generally carried out by different companies. The production of status information is therefore genuinely distributed among all actors participating in the inter-organizational workflow of a building object.

Status information is not very complex or voluminous. However, since status updates are produced by many different companies and the ICT systems differ from party to party, it has been difficult to organize the status sharing in a manner that all parties interested in the status of the work would be able to see it. The key problem is to configure the complex information flows in an easy and flexible manner.

In the WoBD approach, each party publishes status events in its own domain, and the event contains a link to a building object in a BIM model published elsewhere. The same party can publish multiple status events concerning one object, and different parties can publish status events about the same objects.

Through backlinking mechanisms, whenever a new status event is created referring to a building object, a notification is sent, and a corresponding inverse link back to the status event is created at the server hosting the BIM model. Each building object will therefore contain links to all the status events referring to it. That is, it contains its whole status history as links, but the status events themselves remain in the domains of their original publishers.

Table 5.5. Table 4 – Status sharing use case

Stakeholders	Models/datasets
BIM model author	BIM models
Status event provider (detailer, contractor, fabricator, logistics company)	Status datasets
BIM model users	

It would be beneficial to promote the open sharing of status information since it can create visibility and transparency to the execution of a project, which again provides opportunities to improve the controllability and efficiency of the project. However, there are also many specific uses for status information. In the workflow of a component, the downstream parties can predict and understand when they need to be ready for their activities, the upstream parties (such as designers) need status information to understand when changes to components are still possible, and the project management wants to monitor the progress of the work and react to deviations.

This use case creates the following challenges to access control:

- If someone wants to access the status history of a construction object, she needs to visit several other sites to access the status events themselves. To make this practical, there needs to be a proper single sign-on solution (Shaer 1995) in place.
- There can be competitors in the same project that produce status events about different construction objects (e.g., two precast concrete fabricators). They may both be ready to publish status events to other parties in a project but not to each other, as they prefer to hide details about their operational efficiency from each other.
- The server that publishes status events referring to an object in a remote server needs to notify that remote server about a new link. The former server needs to authenticate to the latter server.

5.1.4 Objectives for access control

Common to the use cases is the need to authenticate the user in an effortless way that lessens the burden of memorizing passwords, but there is also the need to

identify the communication parties to prevent any bogus activity and to be able to filter the presented content by the business relation that they have.

We presume that cooperation takes place between parties that are equal. They are companies and their employees working for them. To make the solution neutral, if possible, there should not be a central registry or coordinator for the user data or sessions. User management should be decentralized, which requires the standardization of security conventions. In this context, a common ontology for security roles would be essential.

There may also be the need to be able to refer to projects or facilities in the same way across parties, e.g. an architect could be given read rights for all entities in a certain identified project. All partner firms could have their internal projects but have a way to refer to the common grounds of the cooperation.

Autonomous security management can be interpreted so that every data producer owns and manages their data, which would solidify the security arrangements. No external user needs write access to the system, but she can make change requests or publish her enhanced version of the dataset if some added property, a status event, or an issue would be preferred to be included. The new set could still be linked to the existing data.

The CIA triad (Perrin 2008; Greene 2006) is one of the security models used to assess the safety of systems: C stands for Confidentiality, I for Integrity, and A for Availability. Our focus is on the access control of building data. When that is reflected on the triad, most of the objectives map on the confidentiality. As mentioned in (Rouse et al. 2014), one of the key targets for companies is to assess and classify the severity of the possible damage that could be affected when the confidentiality of a piece of data is broken. The planned access control should enable expressing of the shielded entities and make it possible to protect them from any malicious use.

Trespassing is not prohibited on the Internet but a criminal intent is an essential part of any crime (University of Minnesota Libraries 2015). In legal terms, both cracking and burglary involve unauthorized breaking into something for the purpose of a felony (Thomson Reuters 2016a; Thomson Reuters 2016b). The goal of access control is to limit the means the users have to avoid known or unknown threats come true, but if they do, the protections should be valid also in court. That is, all valuable pieces of data should be shielded.

In plain terms, Single Sign-On (Shaer 1995) refers to methods that allow end users to authenticate many services from a single point. The authentication is done once and propagated to the associated systems (Parker 1995). In a

RESTful implementation, that can be implemented using authorization tokens. However, as RESTful services are stateless by default (Pautasso & Cesare 2013), in a distributed and decentralized network the tokens need to be self-contained in a way that the server can know where to check the validity of the authorization. Instead of using a random number, the address of the authentication server needs to be included.

The solution should reflect the agreements between the companies. That is, in the optimal case, it would suffice that parties agree on the cooperation in a user role level and the responsibility of allocating a specific employer would be delegated to the employer. This increases the flexibility of arranging the authorization but implies that the actual users may not be introduced to the company before the usage of the authentication. Furthermore, if complete logging is preferred, the login has to contain enough data to identify the user.

In the case of an emergency, it may be urgent to give the right information to the right people in the right roles as soon as possible. For example, we may not be able to name the firefighters, paramedics, or police officers beforehand, but the roles may be given certain predefined access to the system.

In summary, the main objectives for an access control solution to WoBD are the following:

1. Decentralized implementation.
2. Support for single sign-o.
3. Access rights assigned to roles or properties, instead of individual users.
4. Users should have accessible profile information that determines their roles/properties.
5. Common ontologies for roles/properties.

5.1.5 Solution approaches

In the light of the preceding discussion, we review suitable decentralized authentication protocols and Web-oriented authorization ontologies.

Authentication

OpenID Connect OpenID Connect (Sakimura et al. 2014) is a flexible single sign-on authentication protocol for the Internet. It is built on top of OAuth 2.0

(Hardt 2012) as a layer to provide user information in a standardized way. In that, the identification token is used in the role of an ID card. It contains the URL of the issuer, but the fields are very limited. However, in principle, the sub-claim (Sakimura et al. 2013) of the ID token could contain an address to a WebID profile or the separately fetched OpenID Connect UserInfo profile. The URL could point to a description of the user's role in the cooperation. The solution is only partial since both profile pages would be needed to be protected separately, and it is left to the content provider to apply limitations.

WebID-TLS The WebID-TLS authentication consists of:

1. A public web-accessible user profile that describes the user identity (Sambra et al. 2016),
2. A self-signed X.509 certificate (Housley et al. 2002) that refers to the profile (Story et al. 2016),
3. TLS network protocol (Allen & Dierks 1999) and
4. the protocol that authenticates the end-user when the associated client certificate is introduced (Story et al. 2016).

TLS ensures all communication is encrypted, but it is also used to verify the host of the WebID profile.

The above-described protocol can limit access to the protected resources, but it lacks ways to shield the profile data from malicious data snooping, or attacks from inside like tampering in an untrustworthy server. As proposed in (Wild et al. 2014; Wild et al. 2015), the WebID-TLS should be extended to include signing the WebID profiles and validating the signatures during the login phase, and filtering the access to the profile data.

Instead of testing knowledge like a password, WebID-TLS is based on ownership, i.e. if a user can show that she owns a WebID certificate that gives her rights to a protected resource, she can proceed without a login prompt. It lightens the burden of remembering passwords and thus can serve as a single sign-on system, but care needs to be taken to ensure that nobody can steal the private store. For example, good protection against computer trojans is recommended. A malicious program that attacked a private key store was described in (Doherty 2011).

In general, WebID gives the users all freedom to create the identification data themselves. That is possible, but in the Web of Data context, added assurance is needed to prove that the identity owner can represent the company that she is working for. A solution is to let companies publish employee identities as secured WebID profiles. That ensures also that uniform security measures are in use to

protect the profiles. Still, at the content provider end, extra care should be taken to check that the associated domains belong to the cooperating companies.

Authorization The ownership-based discretionary access control (DAC) is familiar to most of folder-based system users (Ubale & Apte 2014). In that, creators control access rights to their data. These are in line with the objectives of the system, but, in contrast to groups that are typical in a DAC system, we do not have a list of members, but roles assigned by partner organizations. On the other hand, role-based access control (RBAC) (Gritzalis et al. 1997) systems support user roles. Therefore, if both models can be merged as proposed in (Sandhu & Munawer 1998), the system would fulfil our objectives.

As shown in (Sacco & Passant 2011), WebID profiles can be in unison with additional ontologies like Privacy Preference Ontology (PPO), which enable us to express more precisely not only who can access the data, but also what is shown. This is useful when trying to express restrictions to avoid showing some details to a competitor.

Web Access Control vocabulary (WAC) can describe the access rights for URI referenceable objects in a distributed fashion (Villata et al. 2011). WAC is a discretionary access control schema for the web. That is, access control is given for sets of users that have a WebID identifier. It does not use user profiles, but they can be emulated using web-accessible lists of groups. If access control lists or parts of them are network-accessible, they need special care in security considerations.

5.1.6 Discussion

Web of Building Data creates a challenging environment for user authentication and access control. Only a small part of building-related data can be published openly. There are obvious needs - security, privacy, competition, and IPRs - for controlling access to much of the data. In addition, access control should be implemented so that the decentralized nature of WoBD is respected, and no unnecessary centralized mechanisms are adopted.

The web of Building Data approach has the advantage that - as a participant in a decentralized system - every data producer is free to determine the access control policies for the data it owns. However, from the perspective of the user, this can be a disadvantage if the access control systems differ among hosts that contain interlinked data. When accessing data from interlinked datasets, a user may need separate authentication to each of the different servers, and for the

publishers there can be an additional burden to specify access control rules for a heterogeneous set of intermittent users.

The challenges arise as a result of linking. Without linking, the hosts would be separate from each other, and each of them could use traditional access control solutions. The opportunities provided by Web of Building Data to link data across hosts, to follow links and to aggregate data from multiple hosts can create a fragmented environment from the perspective of access control. The data at different ends of a link could have completely different access control policies, and use even different to define those policies. It can become tedious to set up the access rights to each user at each different host, and to users to access interlinked data if new authentication is required in each host.

A harmonious access control scheme needs the employ some form of a single sign-on solution, and common ontologies for defining access control policies. At the technical level there are existing approaches - such as WebId-TLS or OpenId Connect - that are designed to genuinely distributed environments and that can be adopted in the Web of Building Data platforms. Common ontologies for a description of roles and properties of users can be developed, as well as common policies to refer to instances central in access control (projects, facilities, sites). The research on the security solutions for the DRUMBEAT platform continues in these directions.

5.1.7 Conclusions AND FUTURE WORK

Although the Web of Building Data creates a challenging framework from the perspective of access control, there is a set of emerging technologies – OpenId Connect, WebId+TLS, WAC, ontologies - that could be used to set up appropriate mechanisms to solve problems in a reasonable manner.

It should be noted that the current state of the access control of building data is far from perfect. Complete models need to be transferred to some parties – revealing unnecessary information – and other parties may not gain any access to data that they would desperately need. The granular access control to building data would already mean progress with respect to the current situation. The technologies to achieve that already exist.

The work on the DRUMBEAT platform continues with the implementation of and practical experiments with distributed access control discusses above. There is a need for a practical solution that gains the acceptance of different stakeholders by balancing the simplicity and understandability of the design with the level of

security achieved.

5.1.8 Acknowledgements

This research has been carried out at Aalto University in DRUMBEAT project ("Web-Enabled Construction Lifecycle", 2014-2017) funded by Tekes, Aalto University, and the participating companies.

5.1.9 References

Allen, C. & Dierks, T., 1999. The TLS protocol version 1.0.

Berners-Lee, T., 1998. Hypertext Style: Cool URIs don't change. W3. org. Available at: https://www.w3.org/Provider/Style/URI.html.

Berners-Lee, T., Fielding, R. & Masinter, L., 2005. RFC 3986. Uniform Resource Identifier (URI): Generic Syntax.

Bizer, C., Heath, T. & Berners-Lee, T., 2009. Linked data-the story so far. Semantic Services, Interoperability and Web Applications: Emerging Concepts, pp.205–227.

Brickley, D. & Guha, R.V., 2014. RDF Schema 1.1. W3C Recommendation, 25, pp.2004–2014.

Carroll, J.J. & Stickler, P., 2004. RDF Triples in XML. In Proceedings of the 13th International World Wide Web Conference on Alternate Track Papers &Amp; Posters. WWW Alt. '04. New York, NY, USA: ACM, pp. 412–413.

Dean, M. et al., 2004. OWL web ontology language reference. W3C Recommendation February, 10.

Doherty, S., 2011. All your Bitcoins are ours.... Symantec Security Response. Available at: http://www.symantec.com/
connect/blogs/all-your-bitcoins-are-ours.

Fielding, R. & Reschke, J., 2014. RFC7231: hypertext transfer protocol (HTTP/1.1): semantics and content. Internet Engineering Task Force (IETF).

Fielding, R.T., 2000. Architectural styles and the design of network-based software

architectures. University of California, Irvine.

Greene, S.S., 2006. Security policies and procedures, New Jersey: Pearson Education.

Gritzalis, S., Stefanos, G. & Diomidis, S., 1997. Addressing Threats and Security Issues in World Wide Web Technology. In IFIP Advances in Information and Communication Technology. pp. 33–46.

Hardt, D., 2012. The OAuth 2.0 Authorization Framework (RFC6749). Available at: https://tools.ietf.org/html/rfc6749.

Hoang, N.V. & Törmä, S., 2015. Implementation and Experiments with an IFC-to-Linked Data Converter. Available at: http://itc.scix.net/data/works/att/w78-2015-paper-029.pdf.

Housley, R. et al., 2002. Internet X. 509 public key infrastructure certificate and certificate revocation list (CRL) profile.

ISO, 2013. Industry Foundation Classes (IFC) for data sharing in the construction and facility management industries, Geneva, Switzerland: International Organization for Standardization.

Kirrane, S. et al., 2013. Secure manipulation of linked data. In The Semantic Web--ISWC 2013. Springer, pp. 248–263.

Lassila, O. & Swick, R.R., 1999. Resource description framework (RDF) model and syntax specification.

Miles, A. & Bechhofer, S., 2009. SKOS simple knowledge organization system reference. W3C recommendation, 18, p.W3C.

Parker, T.A., 1995. Single sign-on systems - the technologies and the products. In European Convention on Security and Detection. Available at: http://dx.doi.org/10.1049/cp:19950488.

Pautasso, C. & Cesare, P., 2013. RESTful Web Services: Principles, Patterns, Emerging Technologies. In Web Services Foundations. pp. 31–51.

Pauwels, P., Pieter, P. & Walter, T., 2016. EXPRESS to OWL for construction industry: Towards a recommendable and usable ifcOWL ontology. Automation in Construction, 63, pp.100–133.

Perrin, C., 2008. The CIA triad. Dostopno na: http://www. techrepublic. com/blog/security/the-cia-triad/488.

Rouse, M., Haughn, M. & Gibilisco, S., 2014. What is confidentiality, integrity, and availability (CIA triad)? - Definition from WhatIs.com. WhatIs.com. Available at: http://whatis.techtarget.com/definition/Confidentiality-integrity-and-availability-CIA [Accessed April 27, 2016].

Sacco, O. & Passant, A., 2011. A Privacy Preference Ontology (PPO) for Linked Data. In LDOW. Citeseer.

Sakimura, D.N. et al., 2013. OpenID Connect Basic Client Implementer's Guide 1.0-draft 37.

Sakimura, N. et al., 2014. Openid connect core 1.0. The OpenID Foundation, p.S3.

Sambra, A. et al., 2016. WebID 1.0-Web Identification and Discovery (Draft) (April 2016). Dvcs. w3. org. Available at: https://dvcs.w3.org/hg/WebID/raw-file/tip/spec/identity-respec.html.

Sandhu, R. & Munawer, Q., 1998. How to Do Discretionary Access Control Using Roles. In Proceedings of the Third ACM Workshop on Role-based Access Control. RBAC '98. New York, NY, USA: ACM, pp. 47–54.

Shaer, C., 1995. Single sign-on. Network Security, 1995(8), pp.11–15.

Story, H. et al., 2016. WebID-TLS, WebID Authentication over TLS, W3C Editor's Draft, 05 March 2014. Dvcs. w3. org. Available at: https://www.w3.org/2005/Incubator/webid/spec/tls/.

Thomson Reuters, 2016a. Burglary Overview. Findlaw. Available at: http://criminal.findlaw.com/criminal-charges/burglary-overview.html.

Thomson Reuters, 2016b. Findlaw Legal Dictionary. Available at: http://dictionary.findlaw.com/definition/cracker.html.

Törmä, S., 2014. Web of building data – integrating IFC with the Web of Data. In ECPPM 2014. pp. 141–147.

Ubale, S.A. & Apte, S.S., 2014. Comparison of ACL Based Security Models for securing resources for Windows operating system.

University of Minnesota Libraries, 2015. CRIMINAL INTENT. Criminal Law.

Available at: http://open.lib.umn.edu/criminallaw/chapter/4-2-criminal-intent/ [Accessed April 26, 2016].

Villata, S. et al., 2011. An access control model for linked data. In *On the Move to Meaningful Internet Systems: OTM 2011 Workshops*. Springer, pp. 454–463.

Wild, S. et al., 2015. ProProtect3: An Approach for Protecting User Profile Data from Disclosure, Tampering, and Improper Use in the Context of WebID. In Transactions on Large-Scale Data-and Knowledge-Centered Systems XIX. Springer, pp. 87–127.

Wild, S. et al., 2014. Tamper-Evident User Profiles for WebID-Based Social Networks. In Web Engineering. Springer, pp. 470–479.

5.2 Discussion and Conclusions

Data security, encompassing confidentiality, integrity, and availability, is pivotal. This chapter focuses on the first part of the information security triad, confidentiality. It is closely related to the Data Management topic of Data Governance (see section 1.5).

The Access Policy aspect is approached through use cases. For the selected use cases, user groups, objects to be protected, and access rights required for the roles have been considered.

The second part of the article, presented in sections 5.1.4, 5.1.5, and 5.1.6 discusses the requirements for access control and how it could be implemented in a Linked Data environment. Similar access control is presented in section 4.4.

The use cases are limited to those that focus on BIM-based models (architectural, structural, MEP), including data publishing, linking to product data, sharing state information, and linking between models. Rather than individual access roles, we also suggested that roles could be based on the user's relative position in the organization. I elaborated on this concept in my Tomcat 9 extension, which I developed and made available on my GitHub ACLWoBD repository[6]. This extension implements access control using the WebID role path concept.

Although the system is decentralized, the proposed access management is based on the assumption that outsiders do not have direct access to the data. This is not always possible. For example, the long life cycle of a building can be a challenge for a server-based solution. Therefore, we have also developed a concept where linked data is encrypted attribute-based so that attributes correspond to user roles (see my GitHub Multi-AuthorityABE4BIM repository[7]).

[6] https://github.com/jyrkioraskari/ACLWoBD
[7] https://github.com/jyrkioraskari/Multi-AuthorityABE4BIM

Creating security for linked building data requires much know-how. Therefore, when creating an artifact, I have chosen an applied approach when applying WebId-TLS, Solid-server container security, and attribute-based encryption. The solution is based on a chain of trust: for example, I assume that the encryption method is trustworthy and works as expected. There is a limit to the security of this approach. For instance, should a quantum computer be capable of breaking TLS encryption in the future, the user management system relying on it would also become vulnerable.

The fact that some BIM-related data has to be protected against criminal actions or to protect commercial interests leads to a distributed setup, where equal partners share data that needs to be accessed, aggregated, and protected. I have shown how this can be done using Linked Data and web identities and how it can be done by extending standard approaches like Open Common Data Environments.

Chapter 6

Information Security Aspects: Integrity

In this chapter, I address the research question of, how information security aspects of data management related to integrity can be handled.

Hypothesis 6.1 *Creating unique labels for the building data elements using checksums from the graph neighbourhood in the BIM model makes tracking changes in the model more accessible than with the unprocessed IFC/STEP formatted [Pratt, 2001] models. This hypothesis is addressed in papers presented in Sections 6.1 and 6.2.*

I will introduce what support I have for the claims and discuss arguments that are against them.

Stable identities have an essential role when managing the information lifecycle of an asset in an organization. When a stable identifier is missing, creating one can help supervise what changes have been made, document versioning, and synchronize data content between systems. Thus, it is also a helper to enable smoother cooperation. Furthermore, checksums have a central role in guaranteeing the integration of the knowledge model: to deduce if data content is changed and to be able to sign documents. On the other hand, signed checksums provide

a way to show accountability and ownership, but they also cannot be denied later. Checksums can also express a 'fingerprint' pattern to track.

I have been studying how to create stable identities for a piece of information, for example, at the element level. The research questions have been: If an element does not have a unique name, how can one be created to track the data lifecycle and the versioning? This is the case in IFC/STEP formatted models, where the scope of index references (i.e. the line numbers) is limited to the generated IFC file. The values are overwritten and replaced with new lined numbers each time an export is made from a native model like Autodesk Revit.

Also, when elements are materialized from a parametrized model in the export phase or edit in the model is implemented by removing and then recreating the same element, the identifiers of the last version are overwritten and replaced by a new one. Furthermore, in an IFC model, large geometrical structures can be in the models without identifiers in the substructures. Still, it may also be essential to see how the geometry has changed, not just which element is affected by the change.

In DRUMBEAT (see the website[1] of the project), our approach was to replace IFC's line numbering-based naming of STEP files (IFCtoRDF [Pauwels et al., 2019]) with globally unique identifiers (GUID)[Leach et al., 2005] identification whenever possible. In my work, I have followed that practice. The idea has been to create an identifier that would persist from one version of an IFC file or export to another. However, the solution is not comprehensive. For example, the GUID is not always preserved because native programs create certain elements during export. I have, therefore, considered ways to create identities based on the context of the IFC element, which allows the creation of immutable identities when the context is unchanged. The challenge is to choose a context that is small enough, stable and selective enough to make checksum computation efficient and ensure that edits in the context area change the identity with minimal

[1] http://www.drumbeat.fi/

probability. Checksums can identify changes, but the change sets should be close to the changes made.

If elements in the models are given identifying or relatively unique names, or attributes and property value combinations can be used to identify the element, they could be used. Also, associated geometry and relative geometrical positioning can be used. In light of my research, a relatively large graph context is needed to create unique identity tags. Another problem with checksums is that if unique identifiers such as GUIDs are not available as part of the checksum, there is a certain probability of collisions. It leads to unstable GUID identifiers in the model. The work shows how this situation can be tackled.

Image a situation where the facility manager expects that a constant problem in the air terminal is related to the last renovation made. He wants to compare the original and the new as-built models.

The work results shown in the following sections allow calculation checksums for nodes using their graph neighbourhood. It is shown that nameless nodes can be generated names and difference sets of added and removed triples calculated. This indicates the areas of the graph that have been affected. In the use scenario, these sets can be used to pinpoint the areas of the BIM model and the properties that have been changed. In that sense, the facility manager could track the changes between the original model and the model that reflects the renovated building. However, if the new as-build model differs significantly, the LBD higher-level model comparison using the LBD ontologies presented in Section 2.3 can be used.

Figure 6.1 shows my articles in this chapter at a glance.

Publication	Objectives	Method	Key findings
Oraskari & Törmä, RDF-based signature algorithms for computing differences of IFC models., 2015	When using the neighbourhood to create names: • to minimize the neighbourhood size • have proper naming for the blank nodes	The Design Science Research Method: Experimental Controlled Experiment	• Improved AlgSign: Incremental AlgSign and Short Paths Crossings Algorithm (SPCA) • significantly smaller difference sets for IFC-derived graphs for RDF change detection
Oraskari, Computationally Efficient IFC File Comparison Using RDF Graph Paths., 2015	• to find an algorithm that gives as near the optimal difference sets as possible in practical memory consumption and CPU usage.	The Design Science Research Method: Experimental Controlled Experiment	• The presented Short Paths algorithm provides smaller changesets than commonly available RDF diff tools: topodiff, Jena and RDFContextTools RDFC14Ner offer. • The memory consumption was better in average when diffing the largest models • The calculation time was suitable for practical

Fig. 6.1. My articles related to information security aspects of data management related to integrity at a glance

6.1 RDF-based signature algorithms for computing differences of IFC models (AC 2015)

In this article, I have studied a case in which an IFC building information model was expressed using the ifcOWL/RDF format [Terkaj and Pauwels, 2014]. As the IFCtoRDF converter converts the IFC/STEP notation files line by line using the EXPRESS definition of the standard [Beetz et al., 2009], there have been two options to create Uniform Resource Identifiers (URIs)[Berners-Lee et al., 2005] for lines with no globally unique identifier associated: either to use the original line numbering or, as

here, to express that we do not have a constant identity for the lines, and therefore use nameless nodes.

Here I have studied the latter approach and hypothesized that we could use the neighbourhood of the nodes to calculate a checksum and use it as a semi-constant identity for the node. In this article, I reason why a semi-constant identity can help calculate version differences. Semi-constant here means that the identity is stable as long as there are no changes in the neighbourhood.

My paper below presents my algorithms for solving the diff calculation problem.

ifcOWL reflects the IFC/STEP standards closely. IFC/STEP files have file line numbers that we can expect to be random for each export from a native model into IFC. Diffing the files directly would lead to almost all file lines being reported as changed. Therefore, the line numbers are here replaced with nameless nodes.

Hypothesis 6.1 claimed that creating unique labels for the building data elements using checksums from the graph neighbourhood makes tracking changes in the model more accessible compared to IFC/STEP when elements do not have a unique identifier or the naming is unstable. First, if a unique checksum is possible to name (i.e., change the blank nodes into URIs, for every unnamed node, then the change set can be calculated just by the set subtraction:

set2-set1 = added triples and
set1-set2 = removed triples.

For example,
set1={a, b}, set2={b, c},
added triples = {c},
removed triples = {a},

where characters represent triples.

However, a change in a node's neighbourhood can also change its checksum. Therefore, the question is how to minimize the neighbourhood size and still have proper

naming for the blank nodes? The articles theme is finding a good algorithm for ifcOWL models (with nameless nodes).

When neighbourhood size is narrowed, inevitably, there is a point where checksums collide, i.e., two or more nodes share the same checksum value. The work shows that it can be handled by just appending the checksums random number that unifies it. It affects the difference set sizes but allows the calculation of the added triples and deleted triple sets.

If we limit the Linked Data representation of building information so that it does not contain any blank nodes and the base URLs of the graphs are unified, comparing graphs is easy, as shown in [Oraskari et al., 2021b]. This approach can be used only when static identifiers for entities in the models are available. The article shows that the new LBD ontologies (see the GitHub repository of W3C Linked Building Data Community Group for ontologies[2]) have opened new possibilities here, as the models contain elements with the IFC global identifiers (GUIDs).

Table 6.1 summarizes the use of the design science research method in this research is summarized in Table ref.tab:OraskariTorma2015.

My contribution to the paper: The content is mine. The introduction was written together D.Sc. (Tech) Seppo

Törmä and he also supervised the work.

Table 6.1. The work of Oraskari and Törmä [2015a] in the Hevner et al. [2004] design science research framework

Guideline 1: Design as an artifact
Methods to calculate checksums for sub-graphs of ifcOWL. I created two new methods and algorithms to create signatures for blank nodes using its surrounding at the graph: an iterative version of the AlgSign called Incremental AlgSign, and the Short Path Crossings Algorithm (SPCA). The algorithms were presented in an implementation-independent pseudocode, and a proof of construct was written in Java using Apache Jena. The high-level pseudocode and associated description allow developers to use any programming framework to reimplement the presented ideas.

Guideline 2: Problem relevance
In ISO 19650 information container models, reviews could be facilitated by an efficient change extractor of the new data revision. One business problem is how the model version differences can be calculated. In addition, when sharing data, it is essential to be able to check if the data content has been tampered with. The checksum is also a pre-step for signing the data, facilitating cooperation where responsibility and the data origin are essential. When handling model versions in IFC/STEP format, it is valuable to calculate what has changed in the model between versions. The difference sets are valuable for supervision and when synchronizing model changes over the net and in version control. Also, comparing the similarities of the two models can give designers some valuable information. In addition, it is also valuable to test a model's integrity: that it is not changed.

[2] https://github.com/w3c-lbd-cg/ontologies

There have been some tools for calculating differences, such as Carroll's [Carroll, 2003] method to calculate checksums for RDF data. The general method that works IFC data presented as RDF did not exist.
The constituent user community of the artefact are the researchers of the Linked Building Data Community, but also developers in the AEC sector that want to use BIM data in their software.
The solution belongs to the Common Data Environment and integrity problem classes.

Guideline 3: Design evaluation

Hevner et al. Design Research Method: In the Experimental Controlled Experiment, random modifications were made to a model, and it was compared to how well the algorithms could find the changes.

Guideline 4: Research contributions

The new algorithms were better in the problem domain (IFC data presented as RDF without line numbers). The presented algorithms show how node neighbourhoods can be used to calculate checksums. The method has many applications: integrity checks, comparing models for versioning, but also from different origins, and pre-step for signing content. My contribution focused on Improving the existing methods and enabling the calculation in the ifcOWL contexts.

Guideline 5: Research rigor

The work was peer-reviewed. The source code for the implementation and test data were made available in my GitHub IFCChangeDetection repository at https://github.com/jyrkioraskari/IFCChangeDetection with the permissive Apache License version 2.0.

Guideline 6: Design as a search process

The improved AlgSign algorithm was based on the earlier work of Tzitzikas, and numerous algorithm versions were created. The paper presents the best one.

Guideline 7: Communication of research

Elsevier Automation in Construction, vol 57, 2015

Elsevier allows the inclusion of the paper in the dissertation for non-commercial purposes[3]. The work is originally published in [Oraskari and Törmä, 2015a]. The content presented here is based on our final draft of the article.

According to Google Scholar, the article had been cited 22 times by January 2023.

Jyrki Oraskari Department of Computer Science, Aalto University, School of Science, P.O. Box 15400, FI-00076 AALTO, Finland, Jyrki.Oraskari@aalto.fi
Seppo Törmä VisuaLynk Oy, Espoo, Finland, Department of Computer Science, Aalto University, School of Science, P.O. Box 15400, FI-00076 AALTO, Finland

Abstract: The capability to accurately detect changes between successive versions of the IFC representation of a BIM model would enable the development of generic change management functionalities for construction projects. Unfortunately, IFC models consist mostly of anonymous objects without stable identities; therefore the computation of differences between versions is complicated. When IFC models are converted into an RDF representation, the uniform graph structure offers new algorithmic opportunities. We study how to assign unique and stable identities to anonymous nodes based on signatures generated from their graph environment, and present the Short Paths Crossings Algorithm (SPCA) that computes sets of paths with limited length from anonymous nodes taking into account their crossings. Empirical tests show that SPCA produces significantly smaller difference sets for IFC-derived graphs than previous algorithms for RDF change detection.

keywords: BIM, RDF, IFC, Diff

6.1.1 Introduction

Large building projects nowadays use *Building Information Modeling* (BIM) to produce models of different aspects of building: architecture, structures, plumbing, heating, air-conditioning, and so on [Eastman et al., 2011]. Multiple interrelated models are produced in individual design tasks by different parties with discipline-specific tools. However, the overall design processes are characterized by frequent changes and the need to manage the impact of the changes on interrelated models. Traditional change management practices require repeated comparisons of models and manual work from designers. To enable more agile and automated change management, information about detailed relations between entities in different models would be useful [Torma, 2013].

[3] https://www.elsevier.com/about/policies/copyright, Online Accessed 20th September, 2023

Solutions for interlinking entities across different models in a standards-based manner using the Web of Data technologies are in development [Törmä, 2014]. BIM models can be exported in Industry Foundation Classes (IFC) format [ISO, 2013] and converted into Resource Description Framework (RDF) [Lassila and Swick, 1999]. Links can then be generated between the URIs of entities across the models to support cross-model information access and change management (Fig. 6.2).

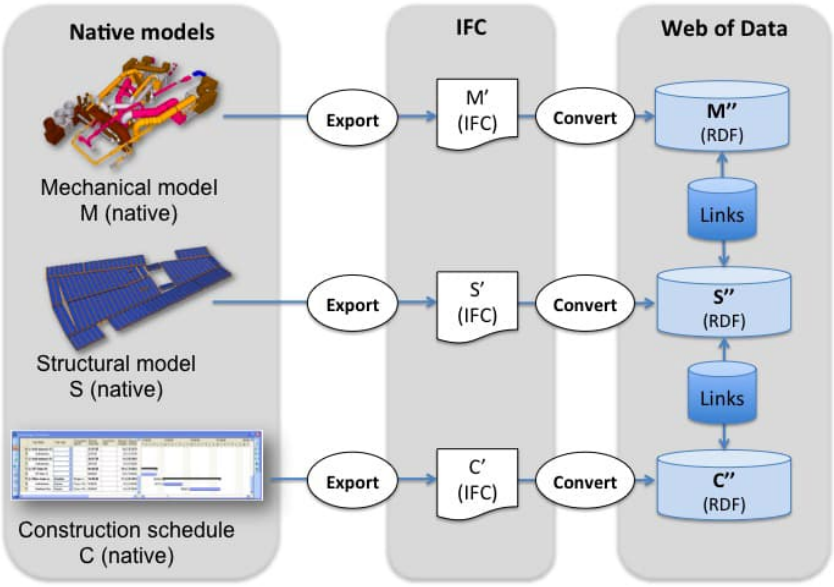


Fig. 6.2. Exporting native models into a common conceptual model and representation

An important enabler to efficiently manage changes between interrelated models is the capability to *accurately detect changes* from one version of a model to the next. These kinds of changes are typically called *diffs* and they are represented as sets of deletions and additions between versions. If correct diffs can be computed, the objects that have changed can be determined; if these are linked to objects in other models, change requests or notifications can be produced [Torma, 2013], possibly in the form of BIM Collaboration Format (BCF) messages. In addition, diffs can be used to manage the change history of a model, for instance, to undo changes or to carry out analyses.

However, computing diffs between IFC model versions is not straightforward. Only a small minority of instances are identified with Globally Unique Identifiers (GUID) [Recommendation, 2005, Leach et al., 2005] and most instances are *anonymous*: they are identified only with line numbers but unfortunately in IFC files the line number of an instance is almost always different in each version. Without stable identifiers, it is difficult to trace objects between versions, and

consequently determine which objects have changed.

The problem of anonymous nodes remains when an IFC model is converted into an RDF graph, where the anonymous objects are represented as *blank nodes*, the RDF equivalent to anonymous objects. However, the uniform graph structure of RDF creates new opportunities to produce unique and stable identities for objects by collecting characteristic information from their unique environment in the graph.

The goal of this research is to *develop efficient algorithms for computing diffs between successive versions of IFC-derived RDF graphs*. The diffs have to be *correct* (contain the whole change) and they should be close to *minimal* (contain only the change). Because diffs of RDF graphs are trivial to compute in the absence of blank nodes, the approach is to compute unique and stable identities for all blank nodes, i.e., to assign the same identities to same objects in different versions. There are currently numerous BIM tools in widespread daily use; the presented algorithms do not require any modifications to those tools but work on the basis of the IFC standard and W3C-specified Web of Data technologies.

As the main result, we present the *Short Paths Crossings Algorithm* (SPCA) which determines the identity of each blank node by computing a set of paths with limited lengths starting from the node. In addition to utilizing the information contained in the paths themselves, the algorithm computes the crossings of the paths since they contain useful information about the structure of the graph in the neighbourhood of the node.

Two test cases were considered: (1) model versions that were manually created with BIM tools and (2) randomly generated removals to models. Compared with the best previous algorithms for RDF change detection, SPCA produces significantly smaller difference sets for IFC-derived graphs. It needs to be emphasized that the results of this research are limited to IFC-derived RDF graphs which are characterized by unusually large numbers of blank nodes. Whether the results apply in other domains is a topic of future research.

The remainder of this paper is organized as follows. Section 2 defines the problem and reviews the related research. Section 3 is devoted to signature-based algorithms: AlgSign, Incremental AlgSign, and SPCA. Section 4 concerns the approximation of the size of the actual change in the graph to enable the evaluation of the algorithms. Finally, the algorithms are empirically tested, showing the superiority of SPCA. In the end, conclusions and ideas for future research are presented.

6.1.2 Problem

The context of the problem is the design phase of a typical construction project in which a number of parties create different, interrelated BIM models with discipline-specific BIM tools. In this study, no special assumptions about design practices or environments have been made. As usual, the master data is maintained in the native models of the BIM design tools which makes the data ownership clear and is good for rapid change management. In particular, no centralized repositories are assumed nor any modifications to existing BIM tools.

BIM models need to be exchanged with other designers to keep interrelated models in agreement and to enable subsequent design work. However, since models usually change many times, they need to be exchanged repeatedly, and other designers must study them all over again to understand the changes. The goal of this research is to reduce the manual work of designers by automatically and accurately detecting the specific changes so that focused notifications can be produced.

In a heterogeneous tool environment the models must be shared in a standardized format. As shown in Fig. 6.2, each model is translated into:

- a *standard ontology* which in this case is IFC, an ISO/PAS 16739 standard for building models, and
- a *standard representation* which is RDF, a graph-based format for sharing structural data over the Web.

Basically, each IFC entity is translated into a set of RDF *triples* that are tuples with three fields called *subject*, *predicate*, and *object*. In each of the triples the IFC entity maps to the subject field, one attribute of the entity to the predicate field, and the attribute value to the object field [Beetz et al., 2014]. The identifiers need to be mapped from GUIDs into IRIs (Internationalized Resource Identifiers) [Duerst and Suignard, 2005], anonymous objects into blank nodes, the values into suitable XSD datatypes [Biron et al., 2004] used by RDF, and the names of types and attributes into corresponding ontology classes and properties.

RDF has a number of advantages in distributed publication of data. It uses Web compatible identifiers, it has constructs for shared vocabularies, synonyms, and concept matching, it enables the merging of graphs, and it is based on conceptual models that can be extended incrementally [Bergman, 2009].

However, it turns out that models translated from IFC into RDF contain a substantial number of blank nodes. There are several sources for them:

1. *Anonymous IFC objects* having no stable identity in the first place. In the public example models in IfcWiki[4], over 85% of the entities are anonymous, with the average being over 95%. When converted to RDF, the resulting graphs have on average 60-70% of blank nodes.
2. *Conceptual gaps* between a native model and standard formats. To fill the gaps, structures of (often mostly anonymous) objects need to be generated. For instance, the structural model of a building (such as S in Fig. 6.2) does not necessarily contain any geometric information – just types, locations and dimensions of components. When it is exported to IFC, various geometrical structures (points, polylines, and polygons) must be generated to conform to the IFC ontology.
3. *Collections and n-ary predicates* that need to be represented in RDF as graph fragments that contain blank nodes [Chen et al., 2012]. A particularly important source of blank nodes are linked list structures of RDF that are used to represent lists and arrays of IFC.

IFC models are typically exchanged in the IFC-SPF format (IFC STEP Physical File) [Pratt, 2001] in which every line of a file creates an object and has a line number that can be used as a local reference (within the scope of one file) to that object. In the IFC-to-RDF conversion of [Beetz et al., 2014] and [Pauwels et al., 2011], the line numbers of IFC-SPF are used to derive IRIs for each IFC object. This approach avoids blank nodes in a "fake" manner by producing non-identifying identifiers. It works especially poorly across the different versions of a same model: when derived from line numbers, the IRIs of the same objects are generally different in each version, and even worse, the same IRI points to a different object in each different version (e.g., a polyline in version 1, a window in version 2, and a property set in version 3) since every file has the same line numbers but a different object on each line. This practice does not eliminate any of the real difficulties that blank nodes pose to external linking or change management.

Therefore, this work is based on the use of a blank node for each line of an IFC-SPF file when no GUID is available [Törmä et al., 2012], and a GUID-based IRI for those lines that contain a GUID. The GUID-based IRIs are generally stable across model versions. Unfortunately, since there are so few GUIDs in an IFC file, the resulting RDF file will mostly contain blank nodes.

To be able to compare two labeled graphs (such as RDF graphs), there has to be a mapping between common vertices (or nodes) [Champin and Solnon, 2003]. If there are no blank nodes in the graph and simple entailment is used, the diff of two versions can be computed as a simple set difference [Zeginis et al., 2007, Volkel et al., 2005]. If there are blank nodes, the problem is reduced to the general graph matching problem [Dickinson et al., 2003] with a high computational complexity.

[4] http://www.ifcwiki.org/index.php/Examples

Our approach is to make diff computation easy: to eliminate blank nodes by assigning them IRIs that are stable across versions [Mallea et al., 2011].

A diff between two RDF graph versions is represented as a *pair of triples sets*: a set of triples that must be *deleted*, and a set that must be *added* when moving from the previous version to the next.

Since there is a unique owner for each model, and the models are shared with others in a read-only manner, it is sufficient to use only *weak diffs* [Berners-Lee and Connolly, 2004]. They are diffs that only apply to exactly the same model versions from which they were derived, in contrast to *strong diffs* that include more context information to make them applicable to a wide variety of model versions.

6.1.3 Related research

There have been many proposals for handling blank nodes to alleviate the problems they cause. Blank nodes can be grounded by starting from a uniquely named node and using owl:FunctionalProperty and its inverse relations recursively to give unique identities to blank nodes [Berners-Lee and Connolly, 2004]. It is a partial solution since many nodes do not have any such grounding paths. Moreover, some nodes can have many paths which raises the questions whether to use all paths to determine the name or only one of them, and which one, and what happens if paths are added or removed. If these problems can be solved, the groundings would produce unique and relatively stable identities as the probability of added or removed grounding paths is likely to be low. It would be a useful preprocessing step for other methods if nothing else.

If a *primary key* for an object was available, it could be easily mapped into an IRI. In some cases, a set of properties none of which is unique separately – for instance, the name, age and address of a person – can together act as a key that uniquely identifies an object. Ding et al. [Ding et al., 2005] call this method *heuristic grounding*. In our context, this is a relatively rare special case with limited use; often these kind of nodes have identities already.

Lee et al. [Lee et al., 2007] presented a method of naming blank nodes using *predicate paths from existing IRIs* [Berners-Lee et al., 2005]. It assumes that blank nodes are located in the middle of a tree whose root has an IRI. The applicability of this method is limited to these special cases.

The number of blank nodes can also be reduced by *removing redundancy* from a graph, e.g., by eliminating sub-graphs entailed by other sub-graphs.

To match blank nodes in two RDF graphs, Carroll [Carroll, 2002] presented an algorithm to check if the graphs are *isomorphic* and to find a bijection between the blank nodes. Node classification, partition refinements, and use of hashes were introduced as ways to enhance the performance of the algorithm. Carroll showed [Carroll, 2003] that if the N3 representation [Berners-Lee and Connolly, 1998] of the RDF graph is ordered lexicographically ignoring blank node names and there are no equivalent entries, the blank nodes can be renamed based on their position in this ordering. This method is called *canonization*. As the method is based on the order, there is no guarantee of the stability of the labeling when entries are added or removed [Lee et al., 2007, Viswanathan and Finin, 2010].

Carroll argued that there is no known algorithm to canonize all possible RDF graphs efficiently [Booth, 2012, Carroll, 2003] since the problem is graph isomorphism complete [Hayes, 2004]. Unfortunately, IFC-derived RDF graphs in which majority of nodes are blank nodes, cannot in practice be canonized with Carroll's algorithm; the algorithm is *nondeterministic* which means that it can give different results depending on the calculation order. The effects of nondeterminism can be negligible when the number of blank nodes is small (e.g., under 5%) but it becomes a major problem if the number of blank nodes is high (e.g., over 50%).

Sayers and Karp [Sayers and Karp, 2004] suggested annotating RDF graphs with the assigned IRI names — that is, with "_:b hasLabel L" triples — to make it possible to assign the same name for the entities in the subsequent versions. This method works if the RDF graph contains the master data, i.e. the data is created and maintained as RDF. However, it does not work when RDF graph is converted from another source, such as from IFC model. In this case, the labels should be stored with the source data, and this kind of a solution would require modifications to the existing BIM tools.

Tummarello et al. argued in [Tummarello et al., 2005] that since blank nodes are not globally addressable, all additions, removals or changes need to be handled at the level of subgraphs having only internal blank node references. These kinds of subgraphs are called *Minimum Self-contained Graphs* (MSG). They are minimal triple sets whose triples only refer to blank nodes not referred by any triple outside the set. It was shown in [Tummarello et al., 2005] that it was also possible to use the Carroll's canonization of triples to sign the MSGs. An efficient graph synchronization algorithm that used those signatures was introduced in [Tummarello et al., 2007]. However, Newman et al. pointed out in [Newman et al., 2008] that if there are many blank nodes in a graph, MSGs can grow large. It is also our experience that in IFC-derived RDF graphs there is usually one MSG that contains 60-99% of all the blank nodes, which renders the MSG method unusable for us.

In [Mallea et al., 2011] Mallea et al. suggested using either global or a local constant name generators for assigning names for blank nodes. This is closely related to Skolem IRIs that were introduced in the most recent RDF specification [Cyganiak et al., 2013]: they are global and unique identifiers that can be converted back to blank nodes. If RDF graphs are manually created or generated only once, this may be a reasonable solution but insufficient for versioned models where the same IRI should be given to the same object across versions.

It is argued in [Booth, 2012, Sayers and Karp, 2004] that if a deterministic labeling algorithm providing stable IRIs could be created, it would solve the RDF graph canonization problem. Our approach is closest to the signature-based mapping algorithm Alg_{Sign} presented by Tzitzikas et al. [Tzitzikas et al., 2012]. It constructs signatures as character delimited strings for blank nodes using the triples from their immediate neighborhood. Alg_{Sign} is deterministic but does not produce unique names for all the nodes which means that diffs can become unnecessarily large.

6.1.4 Signature algorithms

The *signature of a node* is a *string based on a collection of characteristic information* about the node. The information to include should be selected so that it helps to differentiate the node from other nodes, and that it is likely to stay the same between different versions of the graph. Since blank nodes do not have stable differentiating information in themselves, this information has to be collected from their graph environment. In the case of RDF graphs, the information could consist of the predicate names, IRIs, and literal values in the node neighborhood as well as some structural properties of the neighborhood such as crossings of outbound paths.

A *checksum* can be used to produce a *compact digital fingerprint* of a signature. There are different algorithms for computing checksums, such as MD5 message-digest algorithm [Rivest, 1992] that creates a 128-bit hash value of a string. Although theoretically there is an unlimited number of strings that will have the same MD5 hash, in practice, it is extremely unlikely for two non-identical strings to have same hashes. Therefore, if the signatures of the nodes are different, in practice, their checksums are different too.

The *identity of an RDF node* is represented with an IRI. When a new IRI for a blank node is created, it is composed of three parts:

1. *Base-IRI* which is basically the address of the graph.

2. *Checksum of the signature* of the node.
3. *Index*, if necessary.

As a simplified example, consider the following RDF graph that has one blank node (with local label "`_:a1`") and two triples:

```
_:a1 :address "http://example.com/page1.html" .
_:a1 :size    1234 .
```
(6.1)

As an example, the signature of `_:a1` could be the string::

```
"(_ :address http://example.com/page1.html) (_ :size 1234)"
```
(6.2)

It should be noted that the local label of the blank node "`_:a1`" has been omitted from the signature since it is likely to be different in every version of the graph. Now the MD5 checksum of the signature of `_:a1` is:

```
afcf1ed250a7335dee122e1207145f0f
```
(6.3)

If the Base-IRI (which is common to all blank nodes of a same graph) is `http://example.com/model1/` then the IRI assigned to `_:a1` would be:

```
http://example.com/model1/afcf1ed250a7335dee122e1207145f0f
```
(6.4)

If an IRI is not unique within a graph, it can be appended with a running index to enforce uniqueness. For instance:

```
http://example.com/model1/afcf1ed250a7335dee122e1207145f0f/1
```
(6.5)

When the blank node `_:a1` is assigned an identity, the resulting graph is completely grounded (without blank nodes):

```
http://example.com/model1/afcf1ed250a7335dee122e1207145f0f
     :address "http://example.com/page1.html" .
http://example.com/model1/afcf1ed250a7335dee122e1207145f0f
     :size    1234 .
```
(6.6)

The blank node has now a global identity. It can be referred from outside datasets, and if the relevant neighborhood of the node stays the same, it will have the same identity in subsequent versions of the model. The diffs between model versions can therefore be computed as simple set difference of the corresponding triple sets.

The appended indices should be avoided since there is no way to assign them in a stable manner. The instability caused by indices will make diffs unnecessarily large, although not incorrect. It should also be noted that the structure of signatures can be much more complex than above.

The algorithms presented below produce different types of signatures for blank nodes and achieve a different level of uniqueness of the signatures. The goal is to have as small diffs as possible. The diffs are smaller if the identifiers are more stable, and if the node neighborhood used in the signature computation is smaller. There is, consequently, a need to find ways to extract maximum amount of information from as small node neighborhood as possible.

An RDF graph can be defined according to [Hogan et al., 2014] as:

Definition 6.1.1 (RDF Graph) *(RDF Graph) Assume there exists pairwise disjoint infinite sets of I (IRIs), L (literals), and B (blank nodes). An RDF graph G is a collection of tuples $(s,p,o) \in I \cup B \times I \times I \cup B \cup L$, where s is the subject, p the predicate, and o the object.*

Let us define the following additional terminology:

$$
\begin{aligned}
oclass(G,x) &\equiv \{c|(x,\text{rdf:type},c) \in G\} \\
N(G) &\equiv \{s|(s,p,o) \in G\} \cup \{o|(s,p,o) \in G\} \\
B(G) &\equiv N(G) \cap B \\
L(G) &\equiv N(G) \cap L \\
I(G) &\equiv N(G) \cap I
\end{aligned}
$$

Definition 6.1.2 (Diff between Two RDF Graphs) *Given two RDF graphs G_1 and G_2 whose all nodes have unique IRIs ($N(G_1) \cap B = N(G_2) \cap B = \emptyset$), there is a bijection h between the nodes of the graphs, such that $h(lit) = lit$, and $h(IRI) = IRI$ [Klyne et al., 2014, Van Emden and Kowalski, 1976]. Diff is defined as a pair of triple sets: the deletions and additions:*

$$\text{diff}(G_1,G_2) = (G_1 - G_2, G_2 - G_1).$$

The definition of isomorphism in [Klyne et al., 2014] states that, if there are two isomorphic graphs G_1 and G_2, each triple $(s,p,o) \in G_1$ has to have a counter-pair $(h(s),p,h(o)) \in G_2$. Therefore $\text{diff}(G_1,G_2) = (\emptyset,\emptyset)$.

Subtraction of triple sets is well-defined [Berners-Lee and Connolly, 2004] and by definition loss-less. It follows that, if we give any unique labeling for blank nodes the computed difference sets contain the actual change. Numerous labelings can be generated and they can be evaluated by comparing their sizes.

All the algorithms presented below have a *linear worst-case computation time* with respect to the number of blank nodes in a graph. The number of triples connected to one node can be considered constant; it depends on the ontology and modeling tools and does not grow with the number of nodes. The algorithms also use various limits to the size of the node neighborhood studied and to the number of iterations over the nodes. Consequently, the computing time for each node has an upper limit.

The main difference between the algorithms is the quality of results: the diffs they produce are of different sizes, as will be seen in the empirical tests.

$\mathbf{Alg_{Sign}}$

Alg_{Sign} [Tzitzikas et al., 2012] is a deterministic algorithm for computing signatures for blank nodes: if the neighbourhood of a node does not change, its signature remains the same every time it is computed. This does not apply to other previous approaches. Carrol's Canonization algorithm is non-deterministic and can achieve stability only by adding annotations to the RDF graph, which in our case is not a feasible approach as the graph is converted from an IFC model. Tummarello's MSG algorithm has the same problem since it uses Carrol's Canonization.

Algorithm 1 Alg_{Sign}

```
1:  function Alg_Sign(n ∈ B(G))
2:      list in, out, cls ← {}
3:      for (s, p, n) ∈ G do
4:          if s ∈ B then
5:              in ← in ∪ " ∗ " + ("♣" + p)
6:          else
7:              in ← in ∪ +" ∗ " + (s + p)
8:      for (n, o, p) ∈ G do
9:          if o ∈ B then
10:             out ← out ∪ " ∗ " + (p + "♣")
11:         else
12:             out ← out ∪ " ∗ " + (p + o)
13:     for c ← oclass(G, n) do
14:         cls ← cls ∪ " ∗ " + ("type" + c)
        return sort(in) +" ⋄ "+ sort(cls) +" ⋄ "+ sort(out)
```

Tzitzikas et al. [Tzitzikas et al., 2012] use a set of triples that are connected to a node to generate an identifier for an anonymous node. Alg_{Sign} is presented in Algorithm 1. Signatures are composed using a set of separator or placeholder characters ("∗", "⋄", and "♣") that do not appear in IRIs or literals.

Alg_{Sign} describes a way to compose a signature for a node, but it does not guarantee unique naming in the graph. When signatures are used for IRI generation, an extra step differentiating them with appended indices is needed to keep nodes unique within a graph.

Incremental Alg_{Sign}

Table 6.2 indicates that IFC-derived data is much harder to process than more typical RDF datasets. The Alg_{Sign} signatures for blank nodes are relatively unique when a dataset is generated with BGEN system [Lantzaki et al., 2014] that creates "normal" RDF data when blank nodes are considered. However, IFC data have different characteristics. IFC datasets have more repetitive patterns in the graph neighborhoods of blank nodes, and the repetition makes Alg_{Sign} to create same signatures for many nodes. Finding the correct matches across successive versions of a graph within these groups of nodes having same signatures is NP-hard in general [Dickinson et al., 2003]; we have, therefore, developed an incremental version of Alg_{Sign} that iteratively improves the signatures of conflicting blank

nodes.

Algorithm 2 Incremental Alg_{Sign}

```
1:  procedure IncAlg_Sign(Graph G)
2:      a ←< http://annotations/>
3:      N ← B(G)
4:      for MAXTIMES do
5:          Ga ← ∅
6:          for n ∈ N do
7:              Ga ← Ga ∪ (n a:csum Alg_Sign(n,G))
8:          N' ← ∅
9:          for n ∈ N do
10:             if count((? a:csum Alg_Sign(n,G)) ∈ Ga)=1 then
11:                 n ← IRI(checksum(Alg_Sign(n,G)))
12:                 N' ← N' ∪ {x|(x ? n)∪(n ? x)∧x ∈ B}
13:         if no new IRIs then
14:             Break the loop
15:         N ← B(N')
16:     for n ∈ B(G) do
17:         n ← IRI(make unique(checksum((n a:csum ?))))
```

	Triples	Avg MSG size	Unique Checksums	
			Alg_{Sign}	Incremental Alg_{Sign}
BGEN_JAR_V3 dataset easy	7,790.00	1.80	90.68 %	100.00 %
BGEN_JAR_V3 dataset medium	8,549.00	3.53	90.85 %	100.00 %
BGEN_JAR_V3 dataset hard	8,121.00	67.27	88.49 %	100.00 %
IFC ArchiCAD 1	7,834.00	7.62	41.51 %	85.10 %

Table 6.2. The uniqueness of the signatures

Incremental Alg_{Sign} (Algorithm 2) implicitly expands the environment from which the information for a signature is collected. The IRIs generated at a previous round and concise information from the neighborhoods of the named nodes, can at the subsequent rounds be used to create signatures for other nodes. However, the expansion needs to be balanced with the fact that the more triples we use for a signature, the more probable it is that a change contains a triple that is part of the signature.

Although Incremental Alg_{Sign} is not fundamentally different from Alg_{Sign}, the change is clear. As Table 6.2 shows, Incremental Alg_{Sign} has significantly more uniquely named nodes in the graph.

Incremental Alg_{Sign} uses Alg_{Sign} to compute the signatures for all blank nodes, and then converts the unique ones of those into IRIs. The signatures of the remaining blank nodes are recomputed using Alg_{Sign} again, possibly utilizing the newly created IRIs in their environments. The procedure is repeated until no new IRIs are found, or a maximum repetition count is reached. Finally, the remaining blank nodes are given a unique IRI by appending the conflicting signatures with a running index.

The use of the list of active nodes is applied to limit the calculation to only the nodes that are concerned. At the start, it consists all blank nodes (line 3) of the graph, but at the subsequent rounds, only immediate neighboring nodes of the newly changed nodes are used (line 12).

In practice, the node neighborhood is extended with new IRIs only when a unique IRI cannot be composed. The triple count used for the signature computation remains small, which helps to minimize the impact of a change of the graph to the signature, but it also makes the signatures unique when possible. Consequently, Incremental Alg_{Sign} can be expected to find smaller diffs than Alg_{Sign}.

Short Paths Crossings Algorithm

In the *Short Paths Crossings Algorithm* (SPCA) the identity of each blank node is computed based on a set of paths with limited lengths. The set of paths and the points where the paths intersect can provide a lot of information that makes it possible to differentiate the blank node from other nodes. Below we follow the conventions presented in [Udrea et al., 2007].

Definition 6.1.3 (Nondirectional RDF Path) *Given an RDF graph G and a set of properties $P \subseteq I$, a Nondirectional RDF Path is a sequence $T = (e_1, ..., e_n)$ with $\forall i \in [1, n], e_i = (s_i, p_i, o_i) \in G$, such that*

1. $\forall i \in [1, n-1] : o_i = s_{i+1} \vee s_i = o_{i+1} \vee o_i = o_{i+1} \vee s_i = s_{i+1}$
2. $\forall i \in [1, n] : p_i \in P$

The set of nodes in a path is $nodes(T) = \{d | d = s \vee d = o, (s, p, o) \in T\}$.

Definition 6.1.4 (Signature Path) *Given a Nondirectional RDF Path $T = (e_1, ..., e_n)$ with $\forall i \in [1, n], e_i = (s_i, p_i, o_i) \in G$, the corresponding Signature Path is $S = (a_1, ..., a_n)$ where $\forall i \in [1, n] : a_i = Alg_{Sign}(s_i)$.*

Definition 6.1.5 (Limited Tree Width Path Set) *The Limited Tree Width Path Set of a node x, $\text{LTWP}(x)$ is the set of all Nondirectional RDF Paths $T = (e_1, ..., e_n)$*

with $\forall i \in [1,n], e_i = (s_i, p_i, o_i) \in G$*, where the lengths of paths are no more than* MaxLength *and the number of triples connected to a node in the path is no more than* MaxWidth*:*

1. $x = s_1 \vee x = o_1$
2. $\forall y \in nodes(T) : |\{(y,p,o) \in G\} \cup \{(s,p,y) \in G\}| \leq \text{MaxWidth}$
3. $n \leq \text{MaxLength}$.

Without using the limits, LPWP would use CPU and memory exponentially in proportion to the path length. However, using the limits there exists always a constant maximum amount of CPU time and memory space for LPWP(n) for a node that the processing never exceeds.

The basic idea of SPCA (Algorithm 3) is to find the best set of RDF paths that are augmented with a sorted set of ontology classes and related literals. This is done for each anonymous node, and the nodes are named on the basis of the path set and the crossing points where the paths intersect.

A signature path states that there exists a series of connected nodes that have the specified signatures. The pattern does not use the blank node identifiers. So, we cannot differentiate two blank nodes that have exactly the same set of signature paths unless we can state that one of the path sets shares some nodes while the other does not. By using path crossings in the algorithm, we can collect more information for differentiation without the need of using more triples. That is what SPCA does in lines 7-10.

Algorithm 3 Short Paths Crossings Algorithm

```
1:  procedure PathChecksum(Graph G, Map BottomUpCksums)
2:      a ←< http://annotations/>
3:      Ga ← ∅
4:      for n ∈ B(G) do
5:          P ←LTWP(n)
6:          P ← filter out paths from P that contain a cycle
7:          sum ← MD5(sort(Signature paths of P))
8:          sumlist ← emptylist
9:          for all points where paths in P cross do
10:             sumlist ← sumlist add MD5(position in path + path)
11:         sum ← MD5(sum + sort(sumlist))
12:         Ga ← Ga ∪ (n a:csum sum)
13:     for CH ∈ Sets of nodes that share the same (n a:sum ?) do
14:         if BottomUPCksums.get(c) is the same for all c in CH then
15:             for n ∈ CH do
16:                 n ← IRI(MD5((n a:csum ?)))
17:         else
18:             for n ∈ CH do
19:                 n ← IRI(MD5(make unique(n a:csum ?)))
20: procedure SPCA(Graph G)
21:     G ← G - { (x ? ?) ∪ (? ? x) | oclass(x)='IfcOwnerHistory'}
22:     Make IRI nodes of unstable identity blank nodes in G
23:     BottomUpCksums ←empty map
24:     for n ∈ L(G) do
25:         BottomUpCksums.put(n, n)
26:     for MAXTIMES do
27:         for n ∈ {n|(n ? o) ∈ G ∧ ∀o ∈ keyset(BottomUpCksums)} do
28:             sum ← {(n p BottomUpCksums.get(o))|(n p o) ∈ G}
29:             sum ← sum + {(n p o)|(s p o) ∈ G ∧ o ∈ L}
30:             BottomUpCksums.put(n, MD5(sort(sum + oclass(n))))
31:     PathChecksum(G,BottomUpCksums)
```

The algorithm recognizes entries that are identical. Repeating structures like `IfcDirection`, `IfcCartesianPoint`, and `IfcColourRgb` are identified starting with literals (line 23) and repeatedly applying the rule: "calculate a checksum if all the pointed nodes are already handled" (line 25-28). The values are used in line 12 to prevent differentiating nodes that are the same.

SPCA uses some information from IFC ontology. Metadata is removed since it

often contains changes that do not affect the contents of models. For instance, `IfcOwnerHistory` is removed in line 19 since every version has a timestamp that always changes. Moreover, nodes of classes that are known to have unstable IRIs, e.g. `IfcPropertySet`, are changed to blanks in the line 20.

It is worth noting that the only domain-specific parts of the algorithm are the lines 19 and 20. However, the criteria used in the development of the algorithm are specific to IFC-derived RDF graphs, and the results below have only been computed for IFC-derived graphs.

6.1.5 Estimating the Actual Change

In a real life scenario, diff computation is performed to a sequence of versions of a building model produced by external BIM tools. When a new version of a model is converted into RDF, no separate information is provided about the actually added or removed triples or about the size of the change. Thus, to evaluate how close the diffs produced by the algorithms are to the real diff, a way to estimate the size of the actual change is needed.

We have studied the estimation of actual change based on the idea that when a designer adds an element to a BIM model, the changed structure contains not only a new IRI but also dependent structures like a polyline describing the geometry of the element. When an element is removed, the case is the same.

Change Estimate (CE) is the estimate of the triples that are most probably removed or added together with an IRI. If there is a list structure directly connected to a new IRI, the list is probably added at the same time if no already existing IRIs are involved. The case is the same when a new IRI is connected to an islet of triples that has the only connection to the other graph through one or two new IRIs. This case can be generalized so that the change area can be extended from a triple connected to a new or removed IRI when the area has no connection to the old IRIs.

We rely on the function $connectedTriples(node, tabu, invalid)$ which contains all triples connected to node and reachable from the node without traversing the elements (nodes or triples) in a tabu set. If any of the element in $invalid$ set is encountered, the result is empty.

Definition 6.1.6 (Dependent Change) $\mathrm{DependentChange}(t, n, iris)$ *for* $t = (s, p, o) \in \mathrm{triples}(n)$*:*

$$\mathrm{DependentChange}(t, n, iris) = \{t\} \cup \mathrm{connectedTriples}(o, \{t, n\}, iris)$$

Definition 6.1.7 (Change Estimate) *Change Estimate (CE) for graphs G_1 and G_2 is computed using the set $J = I(G_1) \cap I(G_2)$. The CE is a diff that consists of sets of deleted and added triples:*

$$\mathrm{CE}(G_1, G_2) = (\bigcup \mathrm{DependentChange}(t, n, J), \forall t = (n, ?, ?), n \in I(G_1) - I(G_2),$$
$$\bigcup \mathrm{DependentChange}(t, n, J), \forall t = (n, ?, ?), n \in I(G_2) - I(G_1))$$

Intuitively, *Dependent Change* of a triple t is a set of dependent triples in a sense that if t is removed from the graph, this dependent set would become disconnected[5]. The *Change Estimate* is the union of all dependent changes of the triples in a diff. *Change Estimate* approximates the size of the overall change but is smaller than it. The estimate is used below as a point of comparison to the diff algorithms.

6.1.6 Results

The IFC to RDF conversion and the change detection algorithms described in this paper were implemented in Java [Gosling et al., 2005]. In general, the conversion process follows the IFC EXPRESS Schema [Liebich et al., 2012] mapping to RDFS/OWL [Brickley and Guha, 2003] [Dean et al., 2004] and from a IFC-SPF [Pratt, 2001] to RDF described in [Pauwels et al., 2011]. However, IRIs are used for those entities of a IFC-SPF that have a GUID, and blank node are used to represent other entities.

To hide the differences between algorithms and make the comparison more exact, all triples produced by the algorithms were serialized to strings. The string-set subtraction operations were used to compute the sizes of diffs.

Complete source code and all the presented tests are published in our Git repository[6]. Also the test materials are available in the repository.

Two test cases were designed to compare the performance of SPCA with $\mathrm{Alg}_{\mathrm{Sign}}$, with Incremental $\mathrm{Alg}_{\mathrm{Sign}}$, and with the base case where no naming was done.

Table 6.3 reports the characteristics of the datasets. It is shown that there are relatively few nodes identified with an IRI.

[5] We use a domain-specific enhancement to the algorithm: the triples with the predicate `placementRelTo` are removed from a graph before the computation of the change estimate, as these triples connect almost all entities together with a repeatable pattern.

[6] `https://github.com/jyrkioraskari/IFCChangeDetection`, Accessed Oct. 24. 2023

Data set	Triples	IRIs	Blank Nodes	Literals
ArchiCAD 1 (A1)	460	31	185	84
ArchiCAD 2 (A2)	872	63	361	103
ArchiCAD 3 (A3)	888	61	370	123
ArchiCAD 4 (A4)	1,280	71	575	170
Tekla Structures	4,039	181	1,513	200
Solibri Model Checker 1 (S1)	803,132	12,901	422,209	12,849
Solibri Model Checker 2 (S2)	802,041	12,961	421,478	12,960

Table 6.3. The used data sets

To estimate the real change in the graphs, four measures were calculated. First, added and deleted globally unique identifiers (GUID) were counted. The count gave a good estimate of the added and deleted IFC elements in the graphs. Then the Change Estimate (CE defined in Definition 6.1.7) was computed, and two brute force methods were tested.

In the brute force methods, a huge number of bijections are created using a signature algorithm and differentiating the generated signatures. The purpose of this sampling step is to find a change set that is close to a minimal one. The test uses a simple path collection for a node and Alg_{Sign} as algorithms for the signatures. The test was run without any modification to a graph – meaning that the diff should be empty – and with 100 000 samples. However, as shown in Table 6.4 the results of the brute force method were not even close to zero. This is a sign of the difficulty of the problem.

In the first test case, ArchiCAD was used to create a small model and then, a series of modification was crafted and saved into separate IFC files. The versions were separately saved in an IFC file and then converted into RDF. Table 6.5 shows the comparison of the diff results.

In the experiment, SPCA gave the smallest removal sets for all the test data and outperformed in most of the comparisons in the added triple sets sizes. Incremental Alg_{Sign} d excelled when two isomorphic graphs were diffed, but it was only slightly better than AlgSign on average. They all performed notably better than using a dummy algorithm that applied static strings as signatures. The calculation times are small. Even for the largest graph, it took only some seconds, which poses the question, can we use more CPU time to get better results. Java heap spaces are clearly comparable with the sizes of the graphs.

Solibri Model Checker demonstration data[7] is one of the rare publicly available IFC data that are given versions of the same model. In contrast to ArchiCAD data set, the modifications are not artificial but represent a real change made to a building

[7] http://www.solibri.com/. The data is available as a part of the software installation.

by a designer. As shown in Table 6.6, SPCA could handle it decently. The algorithms were tested using a PC that has Intel i7-3820QM 2.79Ghz CPU and 16 GB RAM. The used operating system is 64-bit Microsoft Windows 7 Enterprise. To make the comparison complete, computation times with the test configuration are shown in the table.

ArchiCAD testset	**Statistics**		**Change Estimate, CE**		**Paths checksum max 2 path length**		**Alg_{Sign}, best value**	
	Removed GUIDs	Added GUIDs	Removed triples	Added triples	Removed triples	Added triples	Removed triples	Added triples
No modification A4-A4	0 %	0 %	0.0 %	0.0 %	9.7 %	9.7 %	27.1 %	27.1 %
A1-A2	8 %	64 %	11.2 %	20.2 %	26.0 %	51.9 %	24.4 %	51.1 %
A2-A3	30 %	22 %	9.8 %	10.4 %	37.2 %	38.0 %	33.4 %	34.3 %
A3-A4	0 %	14 %	0.0 %	3.7 %	18.9 %	37.9 %	23.2 %	41.0 %

Table 6.4. The estimates of the minimum change sets

ArchiCAD testset	**No naming**		**SPCA - MD5, path len=2**		**Incremental Alg_{Sign}**		**Alg_{Sign}**	
	Removed triples	Added triples	Removed triples	Added triples	Removed triples	Added triples	Removed triples	Added triples
No modification A4-A4	69.7 %	69.7 %	**5.8 %**	7.0 %	5.9 %	**5.9 %**	29.5 %	29.5 %
A1-A2	67.3 %	73.7 %	**26.2 %**	**38.4 %**	27.0 %	52.4 %	28.0 %	52.9 %
A2-A3	68.9 %	69.3 %	**28.9 %**	**36.4 %**	42.0 %	42.8 %	36.0 %	36.9 %
A3-A4	67.4 %	71.7 %	**14.4 %**	**33.2 %**	19.8 %	38.6 %	26.3 %	43.2 %
S1-S2	65.6 %	65.6 %	**28.0 %**	**27.3 %**	34.7 %	34.6 %	41.8 %	41.7 %
	Time in ms	Java Heap in KB	Time in ms	Java Heap in KB	Time in ms	Java Heap in KB	Time in ms	Java Heap in KB
No modification A4-A4	75	6,173	397	7,512	134	6,242	65	6,459
A1-A2	50	5,257	173	5,851	83	5,298	82	5,396
A2-A3	59	5,553	236	6,364	96	5,606	50	5,741
A3-A4	66	5,880	273	6,958	103	5,942	51	6,119
S1-S2	4,177	1,016,956	74,056	2,064,169	24,866	1,112,977	6,458	1,239,527

Table 6.5. Results of the first test case

Solibri Model Checker modification	Removed triples	Added triples	Time usage in seconds
SPCA path len 1	36.0 %	36.0 %	103
SPCA path len 2	28.0 %	27.3 %	113
SPCA path len 3	23.5 %	24.2 %	153
SPCA path len 4	22.1 %	25.1 %	203

Table 6.6. Results of Solibri Model Checker Test Data

In the second test, a small model was created using Tekla Structures[8] software, converted into RDF, and randomly modified. Random triples were removed from the data set. In practice the generated triples were run through a filter, which made a new version of the dataset by removing every 500th randomly selected triple of the set. A data structure was created to keep track of the removed triples. The original and the modified dataset were then saved into the file format where the original Jena IDs of the nodes were saved. The idea here is that the actual change is saved as an annotation, but the tested diff algorithms are given only the actual RDF graph. It enabled us to categorize the cases when a diff algorithm found the actual change and when false positives or false negatives were listed.

Results are shown in Table 6.7. The first column shows the percentage of the unique node names generated by the algorithm before the duplicate entries are given a running index to make them unique. In general, the better the algorithm is, the higher is the percentage. Here SPCA and Incremental Alg_{Sign} gave the best results.

The test checks whether the checksum associated with an added or removed node is as well added or removed from the graph. SPCA and Incremental Alg_{Sign} recognize all the actual changes and lose no information. Furthermore, a closer look of Alg_{Sign} results reveals that a node is replacing another node in the graph. The resulting graph is isomorphic to the real one. This supports the idea that the signature algorithms are capable to find the real changes in a graph.

As a detail, SPCA performed as well when a checksum algorithm was used and when all data were kept in the form they were collected. This indicates that the MD5 checksum does not deteriorate the diff accuracy at all; instead, it has benefits in the areas of IRI creation and reduced memory use.

	Unique node names	Removed triples		Added triples	
		Correctly excluded	Recognized actual change	Correctly excluded	Recognized actual change
SPCA - MD5	95.3 %	89.1 %	100.0 %	89.1 %	100.0 %
SPCA - no hash	95.3 %	89.8 %	100.0 %	89.8 %	100.0 %
Incremental AlgSign	82.9 %	90.7 %	100.0 %	90.7 %	100.0 %
AlgSign	27.8 %	64.5 %	87.5 %	64.6 %	100.0 %
No Naming	26.5 %	13.9 %	100.0 %	13.9 %	100.0 %

Table 6.7. Results of the test where 1 in 500 of the triples were removed randomly

The time and space efficiency of the algorithms were tested using models of our first test case and the Open BIMServer[9] test data sets of various sizes. Time usage

[8] http://www.tekla.com/

[9] `http://bimserver.org/`

was assessed by measuring how many nanoseconds it took to give each blank node a unique identifier while the Java heap memory requirement was estimated taking a memory usage snapshot when the labeling was ready. The complete table of measurements and the used data sets is included in the Git repository. Table 6.8 shows the equation of linear regression lines of the time usage and Java heap memory usage sizes of each algorithm. The coefficient of determination (R^2) for the equations tells how well the linear equation explains all variety of the data. The range is $[0, 1]$, where 1 means the best fit. Therefore, as all the R^2 values are near 1, we deduce that there is a clear linear relation between the data set sizes and both the time usage of the algorithms and the memory consumption.

	Time in ms equation estimate	R^2	Java Heap space equation estimate	R^2
No naming	0.0031x - 13.602	0.9922	0.8934x - 795.29	0.9996
SPCA - MD5, path length=2	0.0639x + 2271.2	0.9117	1.7676x - 676.72	0.9980
Incremental Alg_{Sign}, limit 20	0.0131x - 7.0512	0.9954	0.9749x - 2631.4	0.9987
Alg_{Sign}	0.0049x + 8.2209	0.9936	1.0923x - 2163.3	0.9995

Table 6.8. The CPU time and Java heap size usage estimates for the algorithms

6.1.7 Conclusions and Future Work

In this paper we have studied the problem of detecting changes in IFC-derived RDF graphs. The research focus has been on the computation of diffs between RDF graphs in the presence of an unusually large number of blank nodes.

The research has benefited from the algorithmic opportunities created by the uniform graph structure of RDF. The starting point of the study was the Alg_{Sign} algorithm presented in the literature for computing signatures of blank nodes based on their immediate graph neighborhoods.

We have presented two improved algorithms: an iterative version of the Alg_{Sign} called Incremental Alg_{Sign}, and the Short Path Crossings Algorithm (SPCA). Both algorithms create signatures for blank nodes but collect more information for them from a slightly larger node neighborhood.

Both the Incremental Alg_{Sign} and SPCA performed well in the empirical tests and the resulting diffs are small when compared against Alg_{Sign} and the actual size of the modifications. Using IFC-derived test materials, SPCA consistently produced the smallest diffs with only a modest increase in time and memory consumption when compared to other algorithms.

The paper shows the usefulness and benefits of RDF-based graph algorithms to address IFC change detection problems. However, the results are specific to IFC-derived RDF graphs; whether they can be generalized to other domains is a topic of future research.

There are several lines of future work to continue this research. The results can be further validated with empirical tests. For this reason, an experimental study with a larger set of IFC models and model versions reflecting actual changes made by designers is in progress. The work includes more rigorous difference analysis of the model versions and the visualization of diffs. Experiments to improve diffs with random sampling and local search have also been made, with promising initial results.

The diff algorithms developed in this research will be used in the Web of Building Data / Linked Building Data tools and applications in the ongoing DRUMBEAT project[10]. The accurate change detection functionality makes it possible to implement change management protocols in which the automatically computed diffs are used to create change notifications or change requests. The possibilities to integrate the change detection with messaging in BIM Collaboration Format (BCF) will also be studied.

The log of diffs can be used to maintain the version history of BIM models and related linksets. The crucial questions to study are how to maintain the log of diffs, how to support the access to versioned data over the Web, and how to use the diffs when answering queries against RDF graphs.

Acknowledgments

This research has been carried out at Aalto University in two research projects — DRUM “Distributed Transactional Building Information Management” (RYM/PRE 2010-2014) and DRUMBEAT “Web-Enabled Construction Lifecycle” (2014-2017) — both funded by Tekes, Aalto University, and the participating companies. We would like to thank the anonymous reviewers for their detailed and constructive comments on a previous version of this article.

[10] http://www.drumbeat.fi/

References

J Beetz, W Coebergh van den Braak, R Botter, S Zlatanova, and R de Laat. Interoperable data models for infrastructural artefacts–a novel ifc extension method using rdf vocabularies exemplified with quay wall structures for harbors. *eWork and eBusiness in Architecture, Engineering and Construction: ECPPM 2014*, page 135, 2014.

Michael K. Bergman. Advantages and myths of rdf. In *Semantic Scholar*. AI3, Adaptive Information blog, 2009. URL https://api.semanticscholar.org/CorpusID:18286677. Accessed on Sep. 05. 2023.

Tim Berners-Lee and Dan Connolly. Notation3 (n3): A readable rdf syntax. *W3C Team Submission (January 2008) http://www. w3. org/TeamSubmission, (3)*, 1998.

Tim Berners-Lee and Dan Connolly. Delta: an ontology for the distribution of differences between rdf graphs. *World Wide Web, http://www.w3.org/DesignIssues/Diff*, 4(3):4–3, 2004.

Tim Berners-Lee, Roy T. Fielding, and Larry M Masinter. Uniform Resource Identifier (URI): Generic Syntax. RFC 3986, January 2005. URL https://www.rfc-editor.org/info/rfc3986.

Paul Biron, Ashok Malhotra, World Wide Web Consortium, et al. Xml schema part 2: Datatypes. *World Wide Web Consortium Recommendation REC-xmlschema-2-20041028*, 2004.

David Booth. Well behaved rdf: A straw-man proposal for taming blank nodes, 2012.

Dan Brickley and Ramanathan V Guha. Rdf vocabulary description language 1.0: Rdf schema. https://www.w3.org/2001/sw/RDFCore/Schema/200212bwm/, 2003. Online; accessed Oct. 17. 2023.

Jeremy J Carroll. Matching rdf graphs. In *The Semantic Web-ISWC 2002*, pages 5–15. Springer, 2002.

Jeremy J Carroll. Signing rdf graphs. In *International Semantic Web Conference*, pages 369–384. Springer, 2003. https://doi.org/10.1007/978-3-540-39718-2_24.

Pierre-Antoine Champin and Christine Solnon. Measuring the similarity of labeled graphs. In *Case-Based Reasoning Research and Development*, pages 80–95. Springer, 2003.

Lei Chen, Haifei Zhang, Ying Chen, and Wenping Guo. Blank nodes in rdf. *Journal of Software*, 7(9):1993–1999, 2012.

Richard Cyganiak, David Wood, and Markus Lanthaler. Rdf 1.1 concepts and abstract syntax. *World Wide Web Consortium, Working Draft WD-rdf11-concepts-20130723*, 2013.

Mike Dean, Guus Schreiber, Sean Bechhofer, Frank van Harmelen, Jim Hendler, Ian Horrocks, Deborah L McGuinness, Peter F Patel-Schneider, and L Andrea Stein. Owl web ontology language reference. *W3C Recommendation February*, 10, 2004.

Peter J Dickinson, Horst Bunke, Arek Dadej, and Miro Kraetzl. *On graphs with unique node labels*. Springer, 2003.

Li Ding, Tim Finin, Yun Peng, Paulo Pinheiro Da Silva, and Deborah L McGuinness. Tracking rdf graph provenance using rdf molecules. In *Proc. of the 4th International Semantic Web Conference (Poster)*, 2005.

M Duerst and M Suignard. RFC3987 - Internationalized Resource Identifiers. Technical report, IETF, 2005.

Chuck Eastman, Paul Teicholz, Rafael Sacks, Kathleen Liston, BIM Discover, and BIM Handbook. *BIM Handbook: A Guide to Building Information Modeling for Owners, Managers, Designers, Engineers, and Contractors*. John Wiley & Sons, 2011.

James Gosling, Bill Joy, Guy Steele, and Gilad Bracha. *Java(TM) Language Specification, The (3rd Edition) (Java (Addison-Wesley))*. Addison-Wesley Professional, 2005. ISBN 0321246780.

Jonathan Hayes. A graph model for rdf. *Master's thesis, August*, 2004.

Aidan Hogan, Marcelo Arenas, Alejandro Mallea, and Axel Polleres. Everything you always wanted to know about blank nodes. *Web Semantics: Science, Services and Agents on the World Wide Web*, 2014.

ISO. Industry foundation classes (ifc) for data sharing in the construction and facility management industries. Iso, International Organization for Standardization, Geneva, Switzerland, 2013.

G Klyne, JJ Carroll, and B McBride. Rdf 1.1 concepts and abstract syntax. *W3C Recommendation*, 2014.

Christina Lantzaki, Thanos Yannakis, Yannis Tzitzikas, and Anastasia Analyti. Generating synthetic rdf data with connected blank nodes for benchmarking. In *The Semantic Web: Trends and Challenges*, pages 192–207. Springer, 2014.

Ora Lassila and Ralph R Swick. Resource description framework (rdf) model and syntax specification. https://www.w3.org/TR/1999/REC-rdf-syntax-19990222/, 1999. Accessed October 17, 2023.

Paul J. Leach, Rich Salz, and Michael H. Mealling. A Universally Unique IDentifier (UUID) URN Namespace. RFC 4122, July 2005. URL https://www.rfc-editor.org/info/rfc4122. Accessed October 17, 2023.

Dong-Hee Lee, Dong-Hyuk Im, and Hyoung-Joo Kim. A Change Detection Technique for RDF Documents Containing Nested Blank Nodes. The 1st International Conference of Ubiquitous Technology, Dubai in U.A.E, 2007.

T. Liebich, Y. Adachi, J. Forester, J. Hyvarinen, K. K.arstila, K. Reed, S. Richter, and J. Wix. Industry foundation classes ifc2x edition 3 technical corrigendum 1, 2012.

Alejandro Mallea, Marcelo Arenas, Aidan Hogan, and Axel Polleres. On blank nodes. *The Semantic Web–ISWC 2011*, pages 421–437, 2011. Springer.

Andrew Newman, Jane Hunter, Yuan-Fang Li, Chris Bouton, and Melissa Davis. A scale-out rdf molecule store for distributed processing of biomedical data. In *Semantic Web for Health Care and Life Sciences Workshop*, 2008.

Pieter Pauwels, Davy Van Deursen, Ruben Verstraeten, Jos De Roo, Ronald De Meyer, Rik Van de Walle, and Jan Van Campenhout. A semantic rule checking environment for building performance checking. *Automation in Construction*, 20(5):506–518, 2011.

Michael J Pratt. Introduction to iso 10303 - the step standard for product data exchange. *Journal of Computing and Information Science in Engineering*, 1(1): 102–103, 2001. American Society of Mechanical Engineers.

ITUTX Recommendation. 667 (2004)| iso/iec 9834-8: 2005. *Information technology–Open Systems Interconnection–Procedures for the operation of OSI Registration Authorities: Generation and registration of Universally Unique Identifiers (UUIDs) and their use as ASN. 1 object identifier components*, 2005.

Ronald L. Rivest. The MD5 Message-Digest Algorithm. RFC 1321, April 1992. URL https://www.rfc-editor.org/info/rfc1321.

Craig Sayers and Alan H Karp. Computing the digest of an rdf graph. *Mobile and Media Systems Laboratory, HP Laboratories, Palo Alto, USA, Tech. Rep. HPL-2003-235*, 1, 2004.

Seppo Torma. Semantic linking of building information models. In *Semantic Computing (ICSC), 2013 IEEE Seventh International Conference on*, pages 412–419. IEEE, 2013.

Seppo Törmä. Web of Building Data - Integrating IFC with the Web of Data. In Ardeshir Mahdavi, Bob Martens, and Raimar Scherer, editors, *eWork and eBusiness in Architecture, Engineering and Construction: ECPPM 2014*, pages 141–148. CRC Press, 2014.

Seppo Törmä, Jyrki Oraskari, and Nam Vu Hoang. Distributed transactional building information management. In *Proceedings of the 1st International Workshop on Linked Dat a in Architecture and Construction (LDAC 2012), Ghent, Belgium*, pages 9–11, 2012.

Giovanni Tummarello, Christian Morbidoni, Paolo Puliti, and Francesco Piazza. Signing individual fragments of an rdf graph. In *Special interest tracks and posters of the 14th international conference on World Wide Web*, pages 1020–1021, 2005. https://doi.org/10.1145/1062745.1062848.

Giovanni Tummarello, Christian Morbidoni, Reto Bachmann-Gmür, and Orri Erling. Rdfsync: efficient remote synchronization of rdf models. *The Semantic Web*, pages 537–551, 2007. Springer.

Yannis Tzitzikas, Christina Lantzaki, and Dimitris Zeginis. Blank node matching and RDF/S comparison functions. In *The Semantic Web–ISWC 2012*, pages 591–607. Springer, 2012.

Octavian Udrea, Andrea Pugliese, and VS Subrahmanian. Grin: A graph based rdf index. *Proceedings of the National Conference on Artificial Intelligence*, 22(2): 1465, 2007.

Maarten H Van Emden and Robert A Kowalski. The semantics of predicate logic as a programming language. *Journal of the Association for Computing Machinery (JACM)*, 23(4):733–742, 1976. Association for Computing Machinery.

Krishnamurthy Koduvayur Viswanathan and Tim Finin. *Text based similarity metrics and delta for semantic web graphs*. PhD thesis, University of Maryland, Baltimore County, 2010.

Max Volkel, Wolf Winkler, York Sure, S Ryszard Kruk, and Marcin Synak. Semversion: A versioning system for rdf and ontologies. In *Proc. of ESWC*, 2005.

Dimitris Zeginis, Yannis Tzitzikas, and Vassilis Christophides. On the foundations of computing deltas between RDF models. *The Semantic Web*, pages 637–651, 2007. Springer.

6.2 Computationally Efficient IFC File Comparison Using RDF Graph Paths (CIB W78 2015)

Hypothesis 6.1 claimed that creating unique labels for the building data elements using checksums from the graph neighbourhood makes tracking changes in the model more accessible compared to the IFC/STEP presentation, where elements do not have a unique identifier or the naming is unstable. This was handled in Section 6.1. The new algorithms improved the performance of the calculation.

Table 6.9 summarizes the use of the design science research method in this research.

My contribution to the paper: All this work is my contribution.

Table 6.9. The work of Oraskari [2015] in the Hevner et al. [2004] design science research framework

Guideline 1: Design as an artifact
I created two new methods and optimized algorithms to create signatures for blank nodes for ifcOWL graphs using their surroundings in the graph. The algorithms were presented in an implementation-independent pseudocode, and a proof of construct was written in Java using Apache Jena. The high-level pseudocode and associated description allow developers to use any programming framework to reimplement the presented ideas.

Guideline 2: Problem relevance

In ISO 19650 information container models, reviews could be facilitated by an efficient change extractor for the new data revision. The business problem is to provide concise and practical difference descriptions for the model versions on time while using reasonable resources.
When handling model versions in IFC/STEP format, it is fruitful to be able to compute what has changed in the model between versions. The difference sets are valuable information for supervision and when synchronizing model changes over the net and in version control. In addition, comparing the similarities of the two models can give designers valuable insights.
Earlier in 2015, I presented two new methods for creating unique naming for nameless nodes in RDF graphs. The algorithms were Incremental AlgSign and SPCA.

Existing tools for calculating differences in RDF graphs were topodiff, Jena, and RDFContextTools RDFC14Ner. The problem was that the change sets were large, making them unusable for synchronization purposes. Also, tracking changes is better if change sets mainly contain the actual changes.
The constituent user community of the artefact is the researchers of the Linked Building Data Community, but also developers in the AEC sector that want to use BIM data in their software. The solution belongs to the Common Data Environment and Integrity Problem classes.

Guideline 3: Design evaluation

Hevner et al. Design Research Evaluation Method: In an Experimental Controlled Experiment, random modifications were made to a model and it was compared how well the algorithms were able to perform (difference set sizes and time).

Guideline 4: Research contributions

The presented Short Paths algorithm provided changesets that were notably smaller than commonly available RDF diff tools: topodiff, Jena, and RDFContextTools RDFC14Ner offered and it better self adaptive than earlier presentes d Incremental AlgSign or SPCA.
The presented work show how node neighbourhoods can be used to calculate checksums. The method has many applications: integrity checks, comparing models for versioning, but also from different origins, and pre-step for signing content. My contribution focused on Improving the existing methods and enabling the calculation in the ifcOWL context.

Guideline 5: Research rigor

The work was peer-reviewed. The source code for the implementation and test data were made available in my GitHub IFCDiff2015 repository at https://github.com/jyrkioraskari/IFCDiff2015 with the permissive Apache License version 2.0.

Guideline 6: Design as a search process

The presented algorithm is the best of numerous tested algorithms. The paper presents the best one.

Guideline 7: Communication of research

CIB W78 Conference, Eindhoven, 2015

The research work is originally published in [Oraskari, 2015].

Jyrki Oraskari Department of Computer Science, Aalto University, Konemiehentie 2, 02150 Espoo, Finland, Jyrki.Oraskari@aalto.fi

Abstract: We study computationally efficient methods of calculating changes in IFC models that have been converted to RDF graphs. It shows how to calculate difference sets that are small enough to be descriptive and usable for synchronization while still using limited computing resources. We present an algorithm that utilizes the neighbourhood of an anonymous node to find a probable matching node pair in the second version of the model. By limiting the neighbourhood only to the most informative paths in the RDF graph, a balance between the certainty of the matching pair and the size of the neighbourhood can be found. The algorithm is compared with commonly available RDF diff tools: Jena, and RDFContextTools RDFC14Ner. When compared, our algorithm provided significantly better results.

keywords: BIM, RDF, IFC, Diff

6.2.1 Introduction

It is common practice to use BIM (Building Information Modeling) tools (Eastman, et al., 2011). The tools have a fundamental role in enhancing the productivity and the quality of the design work, but the change management may require some manual work from designers.

All BIM tools have a native internal representation of the model, but they usually support exporting the model into a standard Industry Foundation Classes (IFC) format (ISO, 2013). The format can be converted into a Resource Description Framework (RDF) graph (Lassila & Swick, 1999) which allows us to use the W3C-specied Web of Data technologies, and enable us to create new ways of computing to change differences in the models without the need for modifying the existing BIM tools.

The difference between two RDF graphs is typically called a diff that can be presented as the sets of removed and added triples (Berners-Lee & Connolly, 2004).

The diff between IFC model versions is not trivial. Rare objects are identied with Globally Unique Identiers (GUIDs) (Leach, et al., 2005). Furthermore, a typical IFC export is presented in the IFC-SPF format (IFC STEP Physical File) (Pratt, 2001) where most objects are identied with line numbers that identify them only in the single file. They cannot be used to trace objects between versions since the line number identifiers are not shared between the IFC exports. It makes identifying the changed objects difficult.

When an IFC model is converted to RDF, each attribute reference of an The IFC element is mapped into a triple using the IFC ontology. The conversion principles are described in (Pauwels, et al., 2011; Beetz, et al., 2014). Our approach is the same with two exceptions. We use blank nodes to convey the meaning of the document-local identifiers (Beckett & McBride, 2004), and the GUIDs of the graph are converted into Internationalized Resource Identiers (IRIs) (Dürst & Suignard, 2005) which emphasizes their roles as the identifiers of the elements.

The result of the conversion, the IFC/RDF graph, has the same structure and content as the original IFC model. They share the difficulty of calculating a diff. The result of the conversion, the IFC/RDF graph, has the same structure and content as the original IFC model. In this study, our objective is to find an algorithm that gives as near the optimal difference sets as possible in practical memory consumption and CPU usage.

In this paper, we present ideas about how the diff algorithm that compares

the consecutive IFC/RDF exports of a native BIM model can be designed to be scalable. The CPU and the memory aspects are considered.

The remainder of this paper is organized as follows. Section 2 reviews the related research. Section 3 introduces the Short Paths algorithm. In Section 4, the result of comparing the Short Paths algorithm with Jena, and RDFContextTools RDFC14Ner is shown. Finally, conclusions and ideas for future research are presented.

6.2.2 Related research

In (Carroll, 2002) Carroll concentrated on the problem of checking if two RDF graphs are isomorphic.

The presented performance improvement for the basic isomorphism test was based on classifying blank nodes according to the number of connected triples where classes could be refined using the adjacent classifications. Furthermore, the same algorithms were extended to classify blank nodes by using the role they appear in the statements. This is employed in Jena (Carroll, et al., 2004).

In (Carroll, 2003) he presented the idea of converting RDF documents into a canonized form so that by using checksums it was possible to check if two RDF graphs represent the same facts. In Carroll's canonization, blank nodes were named depending on the order they are in the lexically ordered and N3 formatted (Berners-Lee & Connolly, 1998) triple set. An individual fill character was used so that the blank node identifiers could be ignored. Furthermore, annotations were used to guarantee that the sorting order of the triples would always be the same.

In 2004 Berners-Lee et al. introduced the general idea of RDF delta in (Berners-Lee & Connolly, 2004). In that, the difference between two graph versions was defined as triple-set subtractions. Insertions were deduced by calculating G2-G1 and deletions using G1-G2. Weak and strong diffs were also defined. By a weak diff, they refer to a difference set that can only be applied to the datasets that they were calculated from. A strong diff can be used in a multitasking system and as a patch for an already changed dataset. Blank nodes were handled by ordering triples by the connected subject nodes that had URIs. That format was named Pretty Printing.

Then, Tummarello et al. stated in (Tummarello, et al., 2005) that, since blank nodes are not addressable from outside, an RDF graph-subgraph of connected blank nodes should be addressed as a unit. The subgraphs are called Minimum

Self-contained Graphs (MSG),. An efficient synchronization algorithm that uses Carroll's canonization for signing subgraphs was introduced.

Auer and Herre put the idea further They presented an atomic change in (Auer & Herre, 2007). It conveyed the same idea as a changed MSG has, but they could be aggregated hierarchically to express how a series of changes are associated.

Our algorithm is presented in the next chapter. It extends the signature diffing method we presented originally in (Oraskari & Törmä, 2015).

6.2.3 Short Paths Algorithm

The basic ideas of the Short Paths algorithm (SP) are in most parts the same as shown in SPCA in (Oraskari & Törmä, 2015). However, the algorithm is redesigned to introduce the ideas of how to keep the calculation efficient and scalable. It is enhanced to handle the problems raised when diffing large IFC graphs. The ideas could most probably be generalized, but the scope of this article is in the BIM and IFC domain.

SP uses a string-based collection of characteristic information of a blank node neighbourhood to differentiate it from other ones. For a blank node and its environment, the information is collected from short triple paths that start from the node to the close neighbourhood of the node. The paths can be presented as a group of strings. That characterizing information of the vicinity of the node is called the signature. An RDF graph and the limited set of paths are defined formally:

The positioning structure of an IFC graph describes how an element is located in space in relation to some other components. The chains of relative positioning form a relatively stable skeleton of the model. We use that to give a geometrical structure a generated unique identifier. It decreases the number of blank nodes and makes the corresponding islet of blank nodes smaller. Listing in Figure 6.3 shows the calculation steps. The IfcLocalPlacement nodes are given a checksum based on the data that constitutes the rotation/translation matrix at the placement. If the data is unique, an IRI can be generated (lines 14 -16).

Since the signature is based on the information that is collected from the close vicinity of the node, we can expect that the existence of the triple pattern in a graph segment has a probability (Bose, 2002). The more signatures we form in the graph segment, the more probable is that there are more than one node neighbourhoods that are given the same signature. This has a direct connection to scalability. The

Definition 1: RDF graph: For mutually disjoint sets of IRIs (I), blank nodes (B) and literal (L). A graph G is set of triples (s p o) $\in I \cup B \times I \times I \cup B \cup L$.

Definition 2: Non-directional RDF Path: Given an RDF graph G, a non-directional RDF path T = (t_1, t_2, t_3, … , t_n) is a sequence of triples, where $\forall i \in [1,n]$, $t_i = (s_i,p_i,o_i) \in G$, and $\forall i \in [1,n-1]$, $s_i=s_i+1 \vee o_i=o_i+1 \vee s_i=o_i+1 \vee o_i=s_i+1$.

Definition 3: The Limited Width Path Set LWPS(*x*, *max_path*) for a node x is defined as follows: LSWPS is a collection of non-directional RDF paths {T_1,T_2,T_3, …, T_m}, where $\forall j \in [1,m]$,T_j is a non-directional path T = (t_1, t_2, t_3, … , t_n) such that $t_1 = (x,p_i,o_i) \in G$, the length of the path $|Tj| \leq$ *max_path*, and $\forall y \in$ nodes({T_1,T_2,T_3, …, T_m}), $|(y, ?, ?) \in G| \leq$ MAXWIDTH $\wedge$ $|(?, ?, y) \in G| \leq$ MAXWIDTH.

Definition 4: Signature Path for a non-directional RDF Path T= (t_1, t_2, t_3, … , t_n) is a string that is composed of ordered set of triples (Y_1, Y_2, Y_3, … , Y_n), where each set Y_i corresponds $t_i = (s_i,p_i,o_i)$ in the path.
Y_i = sort({(s_i rdf:type ?)$\in$ G}) + sort({(s_i ? l))$\in$ G $\wedge$ l $\in$ L})+ sort({(s_i ? ‘B’)| (s_i ? b) $\in$ G $\wedge$ b $\in$ B}) + sort({(‘B’ ? s_i)|(b ? s_i) $\in$ G $\wedge$ b $\in$ B}).
Likewise, Signature(P) for a non-directional RDF path set P is a sorted collection of the Signature Path strings of the paths.

more blank nodes we have, the more probable it is that a small neighbourhood is not enough to give a unique identity to a node. SP handles this by dividing the RDF graph into areas that have a unique identity. Thus, the signature needs to find a differentiation only locally. When an RDF graph gets larger, the graph segments may still be of a reasonable size.

The idea of dividing the graph into areas is twofold. First, when the graph can be divided into areas that have unique identifiers, the needed information count to differentiate a node in an area is smaller than in the case where the whole graph is used. Second, since an RDF graph is usually connected, we expect that a new blank node is attached to one or many nodes of an identity that can be referred to, i.e. GUIDs.

Furthermore, if we suppose that, in general, the objects of the graph triples point to elements that are dependent on the subject, we can transitively relay the dependency relation in the graph. That forms a tree structure in the graph. The paths be assumed quite stable, but the IRI at the beginning of the path set can be recognized in a deterministic way even if there are changes in the subgraph. Thus, only the set of the nearest IDs is used as an identifier for the graph area. Listing in Figure 6.4 shows the algorithm.

Fig. 6.3. Listing: The placement checksum calculation

```
1    a ← http://annotations/
2    Procedure Placement_checksums(Graph G)
3     Ga ← ∅ // Creates an empty annotation graph
4      Function traverse(Node n, current_sum)
5       sum ← sort(coordinates of relativePlacement)
6       sum ← MD5(sum + current_sum)
7       Ga ← Ga + (n a:pc sum)
8       for(x ∈ Πx(x placementRelTo n)do
9         traverse(x, sum)

10    N ←{x|(x type IfcLocalPlacement) ∈ G ∧ |(x placementRelTo ?)∈ G|=0}
11    for n ∈ N do
12     traverse(n, '')
14    temp ← {x, where |(? a:pc Πsum(x a:pc sum))|=1 in Ga }
15     For n ∈ temp do
16      n ← IRI(Πsum(n a:pc sum))
17    Return Ga
```

As in SPCA in (Oraskari & Törmä, 2015), the bottom-up checksums are calculated to recognize the repeating structures of the graph. They are, for example, IfcColor, IfcPoint, and IfcMaterial. The generic function that is presented in Listing in Figure 6.5 works using a simple and repeated rule. First, all literals are assumed to have the checksum (lines 4-5). Then a checksum is added to all nodes that are pointing to nodes that have a checksum already (lines 6-11). That is repeated a maximum round of times. The checksums are used in the main algorithm (Listing in Figure 6.7, line 24) to give the duplicate structures the same identifier.

Listing in Figures 6.6 and 6.7 show the overall description of the Short paths algorithm. First data like the timestamps of the IFC Export are removed since it is presumed that they do not convey information about the actual change (Listing in Figure 6.6 line 1).

SP uses information of the neighbouring triples of a blank node to generate a name that has a reasonably good probability of matching the node with the corresponding one in the second version of the graph. The larger the neighbourhood is, the more easily the node can be given a unique name, but a large collection of triples used for naming makes the name prone to changes. To achieve the balance, only the close neighbourhood of the node is used.

We use dynamic path lengths. It means that those nodes that have not given a unique signature in the last round, are given a new one using step longer paths. The idea behind this is that, in contrast to SPCA, SP finds the path length for a node adaptively. So, the same value can be used for a wide range of graphs. Furthermore,

Fig. 6.4. Listing: The division of the graph

```
1   a ← <http://annotations/>
2   Function CheckDependencies(Graph G)
3    Ga ← ∅ // Creates an empty annotation graph
4    N ← iri nodes of the graph G
5    for all x in N do
6      Ga ← Ga +  (x a:dep x) + (x a:len 0)
7    for maximum rounds
8     for all x in N do
9      N ← N – x
10     j ← Πl{(x a:len l)∈ Ga}+1.
11     for all o in (x ? o) ∈ G ∧ o ∈ B
12       i ← if exist (o a:len l)∈ Ga, l, otherwise ∞
13       if j<i do
14         Ga ← Ga – {(o a:len ?) ∈ Ga} +  (o a:len j)
15         Ga ← Ga – {(o a:dep ?) ∈ Ga}
16       if j<=i
17         N ← N + o
18         Ga ← Ga + {(o a:dep w) |(x a:dep w)∈ Ga }
19   Return Ga
```

since duplicate entries are more probably made unique nondeterministically so that they do not match a pair in the other graph, a new round with longer paths likely increases the chance for the node to get a match. For a larger graph, the paths are automatically longer when needed. However, the maximum values set a limit for computation that is not exceeded for a node. The upper limit forces the computation time to follow a linear path.

On the other hand, the algorithm keeps the triple neighbourhood of only the active node and just the minimum amount of data of the whole graphs in the memory. For this reason, when comparing huge models, the triple count and the minimum quantity of annotations dominate the memory consumption. For that reason, the memory consumption is linear for the large models.

The unique checksums are made into IRIs in line 18 and removed from the node-set in line 17. Lines 6-20 are repeated until a maximum path length is achieved.

Fig. 6.5. Listing: A function to find which IRIs a blank node is dependent to

```
1    a ← <http://annotations/>
2    Function Bottom-up_Checksums(Graph G)
3    Ga ← ∅ // Creates an empty annotation graph
4    for n ∈ L and n ∈ nodes of the graph G do
5      Ga ← Ga + (n a:bu n)
6    for maximum rounds
7     for n ∈ { n | (n ? o)∈ G ∧ ∀o ∈ {Π_key(key a:bu ?)})
8       sum ← sort(bu|{(o a:bu bu)∈ Ga ∧ (n ? o)∈ G})
9       sum ← sum + sort({(n ? l))∈ G ∧ l ∈ L})
10      Ga ← Ga + (n a:bu MD5(sum + sort({t|(n rdf:type t)∈ G})))
11    Return Ga
12
13
```

To make the triple subtraction possible, the nodes that had given a name that shared with other nodes in a graph is given a unique naming by adding an index number (lines 22-28).

The actual SP algorithms are shown in Figure 6.7:

Fig. 6.7. Listing: Short Paths algorithm

```
1    G1 ← Short Paths blank node naming algorithm for graph G1
2    G2 ← Short Paths blank node naming algorithm for graph G2
3    diff ← { G1 - G2, G2 - G1}
```

6.2.4 The memory usage

The memory consumption of the algorithm is an important aspect when the scalability is considered. A checksum can be used as a compact digital fingerprint that effectively has the same capability to differentiate the blank node from other nodes. For a checksum, SP uses the MD5 message-digest algorithm that can be used to generate the 128-bit hash value of the signature, i.e. the sorted set of path strings. Although theoretically, a checksum algorithm loses information, when MD5 is used, the uniqueness of the checksums is almost the same as the original data .

When the memory consumption of the algorithm was tested using Open BIMServer datasets[11], the usage of the MD5 checksumming saved a notable amount of Java heap memory. The results can be seen in Table 1. The memory usage was measured generating an extra full garbage collection in Java and writing down the usage after the clean-up.

Table The memory consumption

Datasets	SP with MD5	SP without MD5
FJK Project Final	9,105K	15,370K
PART02 Wilfer 200302 20070209	95,848K	2,027,109K
Export datasets 1 and 3	39,611K	927,741K

6.2.5 Results

The software that was needed for the experiment was written in Java (Gosling, et al., 2005). The IFC- to RDF conversion follows the principles described in (Pauwels, et al., 2011). However, IRIs are used for those entities of IFC-SPF that have a GUID, and blank nodes are used to represent other entities. The ontology of the test data reflects the IFC specification of the exported dataset. It is based on the ISO/PAS 16739 standard and can be expected to be stable. Therefore, in this study, only the RDF data is diffed. The implementation details, the source code of the experiments, and the datasets are available in our GitHub repository[12].

In the experiment, we used a previously crafted series of changes of a small model to test the algorithms. For these, Tekla Structures (Tekla Corporation) and Graphisoft ArchiCAD were used. The test sets were enriched with Solibri Model Checker (SMC)[13] models and the *export* datasets from Open BIMServer (Beetz, et al., 2010). They were used since they are publicly available and consist of a series that can be diffed. SMC models are useful in that they describe the structure of a reasonably large building, and it is one of the rare publicly available IFC models that are given versions. So, all the models and changes are created using a BIM tool.

[11] https://github.com/opensourceBIM/BIMserver/tree/master/TestData/data

[12] The code is republished in https://github.com/jyrkioraskari/IFCDiff2015, Accessed Oct. 24. 2023

[13] http://www.solibri.com/products/solibri-model-checker/

The dataset statistics are shown in Table 2. In the data, the ArchiCAD and Tekla Structures datasets are the smallest ones. They also have a relatively large number of IRIs. The data were created to serve as the primary benchmark of an IFC diff. The original IFC data and converted RDF/N3 datasets are available in the code repository. Solibri Model Checker (SMC) example datasets are larger and more complex. They represent a whole office building and contain a realistic change. Thus, they are suitable for testing the scalability of the algorithms. Furthermore, the Open BIMServer *export* datasets represent a two-storey detached house. The sets are relatively large and contain very few GUIDs.

Table Statistic of the data used in the experiments

DataSet Name	Short name	Triples	RDF Nodes	IRIs	Blank Nodes	Literals
ArchiCAD 1	A1	706	470	13%	43%	44%
ArchiCAD 2	A2	1362	822	11%	47%	42%
ArchiCAD 3	A3	1372	849	10%	47%	43%
ArchiCAD 4	A4	1954	1212	8%	50%	42%
SMC 1	S1	1137184	615088	1.2%	69.5%	29,2%
SMC 2	S2	1135693	614328	1.2%	69.5%	29.3%
BIMserver Export 1	E1	41344	22865	0.7%	68.4%	30.9%
BIMserver Export 3	E3	41604	23017	0.7%	68.3%	31.0%
Tekla Structures 1	T1	6839	3340	5.9%	46.2%	47.9%
Tekla Structures 2	T2	7439	3648	5.8%	46.6%	47.6%

The comparison was made so that SP and RDFC14Ner provided RDF string sets that were subtracted. topodiff provided the same functionality. Jena created a new model as a result of a diff. So, the statement count of the difference set models was reported. The Jena Model difference method was used in the tests.

topodiff (Ragozin, 2015) was not usable without modification since it worked only if all of the blank nodes were attached to a unique literal. This was true when the original line numbers of the IFC entity were added as an object literal to the blank nodes that represented the entity in the RDF world. When the unique literals were added, the code was able to identify the blank nodes, and the sorting could take place. The results of the sorting were quite good. So, there was no reason to reject the algorithm from the experiment.

RDFC14Ner (Panaioli, 2005) is Fabio Panaoili's implementation of Carroll's

Canonization algorithm. Its problem is that it adds triples to give a blank node the same identity each time for the same dataset. It is useful when one would like to prove that a dataset is not tampered. In this case, the IFC/RDF dataset versions are independent. There is no way to save the insertions to the native data and thus, they only increase the dataset size. That explains why a removal set can be larger than the whole original dataset.

The removal and addition difference set sizes are shown in Table 3 and Table 4. SP provides the smallest sets of removed and added triples in all but the cases where the sets were isomorphic. topodiff gave the best results in the test where the same dataset was compared with itself. The result set sizes were clearly the second smallest. RDFC14Ner values were large because of the added annotations. Finally, Jena had consistent values, but the variation of the result set sizes is small. SP was the only algorithm that had small enough result set sizes so that it is practical in synchronization or finding real changes in the datasets.

Table: Removed triples set sizes generated by the algorithms

Datasets	SP	topodiff	RDFC14Ner	Jena
A4-A4	1%	0%	25%	76%
A1-A2	11%	17%	86%	77%
A2-A3	12%	43%	80%	77%
A3-A4	3%	14%	86%	74%
S1-S2	16%	64%	100%	83%
E1-E3	25%	77%	107%	85%
T1-T2	50%	87%	89%	80%

Table: Added triples set sizes generated by the algorithms

Datasets	SP	topodiff	RDFC14Ner	Jena
A4-A4	1%	0%	25%	76%
A1-A2	21%	57%	96%	81%
A2-A3	19%	43%	80%	77%
A3-A4	22%	40%	94%	77%

Datasets	SP	topodiff	RDFC14Ner	Jena
S1-S2	16%	64%	100%	83%
E1-E3	26%	78%	107%	85%
T1-T2	51%	88%	90%	80%

The algorithms were tested using a PC that has Intel i7-3820QM 2.79Ghz CPU and 16 GB RAM. The used operating system is 64-bit Microsoft Windows 7 Enterprise. The CPU usage was tested using timestamps that were created using Java System.nanoTime(). The runtime was measured from the period where the actual comparison was made.

As Table 5 shows, in this experiment, SP and topodiff had similar computation times. RDFC14Ner was clearly faster in large datasets. Jena used the least amount of CPU time. However, it does not compensate for the poor performance when considering the accuracy of the results. The difference sets were inadequately large.

Table: Calculation time in seconds

Datasets	SP	topodiff	RDFC14Ner	Jena
A4-A4	0.27	0.29	0.27	0.05
A1-A2	0.10	0.17	0.14	0.05
A2-A3	0.17	0.21	0.20	0.04
A3-A4	0.21	0.24	0.23	0.03
S1-S2	208.19	214.40	60.92	8.33
E1-E3	11.43	2.88	2.61	0.35
T1-T2	0.51	0.60	0.64	0.07

The memory consumption was estimated by taking a snapshot of the Java heap memory allocation at the time when the change detection was ready, and the data structures of the diff algorithm were still referenceable. The statistics are presented in Table 6. In the experiment, the memory consumption was in line with topodiff, and RDFC14Ner. Jena used the least amount of memory, which is understandable since SP and RDFC14Ner use Jena as a tool for reading the triples into memory. All

algorithms had all the graph data in memory all the time. topodiff handles the N3 presentation of the triples directly. For that reason, we had expected it to use less memory in general. However, topodiff uses more memory when the datasets get larger.

Table: Java heap memory consumption in MB

Datasets	SP	topodiff	RDFC14Ner	Jena
A4-A4	9.4	5.1	8.5	7.6
A1-A2	8.4	2.9	7.6	7.1
A2-A3	8.8	3.7	7.9	7.3
A3-A4	9.1	4.4	8.2	7.5
S1-S2	1,096.0	2,624.4	1,217.0	651.6
E1-E3	43.1	94.3	46.0	29.3
T1-T2	11.9	17.3	13.5	10.1

6.2.6 Conclusions

The paper shows the effectiveness of the RDF graph neighborhood-based algorithms to detect the changes in IFC-derived RDF graphs. The presented algorithm provides an accurate low-level tool for diff IFC/RDF graphs, which allows us to use it as an enabler in the Linked Building Data toolset in the DRUMBEAT project. It can be used as a core for a version management system, synchronization tools, change analysis and notification streams. However, the results are domain-specific. The generalization of the ideas needs more extensive analysis of the existing RDF datasets.

The Short Paths algorithm provides changesets that are notably smaller than commonly available RDF diff tools: topodiff, Jena and RDFContextTools RDFC14Ner offer. As the difference in set size is a crucial aspect in notification systems, in this aspect, SP excels in the comparison.

Since all the systems keep the RDF graphs in the memory during the change detection, the memory consumption is the major limiting factor when considering diffing larger models. The tools did not differ much from each other's, but the memory consumption of the Short Paths the algorithm was better than the average when diffing the largest models.

The time usage of all the algorithms was usable for practical considerations. Less than 4 minutes time is quite reasonable when diffing a model of the size of an office building. Nevertheless, as only 12% of the CPU resources were used in the experiments, a considerate amount of optimizing the CPU usage is possible.

These results are the first steps of exploiting the benefits of change detections in parallel design in the Web of Building Data. There are several lines of future work to continue this research. The Efforts of using random sampling from a collection of IFC/RDF models have been made with promising initial results. Optimizing the calculation provides us with much potential to study more. Concurrent processing and partial diffing are interesting research topics. However, our primary focus will be on integrating the diff results into the Linked Building Data toolset. Thus, how the changes can be meditated to linked building information models is one of the crucial questions. Visualizations and smart notifications may reduce the manual work of designers.

6.2.7 Acknowledgements

This research has been carried out at Aalto University in two research projects — DRUM “Distributed Transactional Building Information Management” (RYM/PRE 2010-2014) and DRUMBEAT “Web-Enabled Construction Lifecycle” (2014-2017) — both funded by Tekes, Aalto University, and the participating companies.

6.2.8 References

Auer, S. & Herre, H., 2007. A versioning and evolution framework for RDF knowledge bases. Perspectives of Systems Informatics, pp. 55-69.

Beckett, D. & McBride, B., 2004. RDF/XML syntax specification (revised). W3C recommendation, Volume 10.

Beetz, J., van Berlo, L., de Laat, R. & van den Helm, P., 2010.

BIMserver. org–An open source IFC model server. s.l., s.n.

Beetz, J. et al., 2014. Interoperable data models for infrastructural artefacts–a novel IFC extension method using RDF vocabularies exemplified with quay wall structures for harbors. eWork and eBusiness in Architecture, Engineering and Construction: ECPPM 2014, p. 135.

Berners-Lee, T. & Connolly, D., 1998. Notation 3 (N3): A readable RDF syntax. W3C Team Submission (January 2008) http://www. w3. org/TeamSubmission,(3).

Berners-Lee, T. & Connolly, D., 2004. Delta: an ontology for the distribution of differences between RDF graphs. s.l.:s.n.

Bose, R., 2002. Information theory, coding and cryptography. s.l.:Tata McGraw-Hill Education.

Carroll, J., 2003. Signing RDF graphs. The Semantic Web-ISWC 2003, pp. 369-384.

Carroll, J. J., 2002. Matching RDF graphs. In: The Semantic Web-ISWC 2002. s.l.:Springer, pp. 5-15.

Carroll, J. J. et al., 2004. Jena: implementing the semantic web recommendations. s.l., s.n., pp. 74-83.

Dürst, M. & Suignard, M., 2005. Internationalized resource identifiers (IRIs), s.l.: s.n.

Eastman, C., Teicholz, P., Sacks, R. & Liston, K., 2011. BIM Handbook: A Guide to Building Information Modeling for Owners, Managers, Designers, Engineers and Contractors. s.l.:John Wiley & Sons.

Gosling, J., Joy, B., Steele, G. & Bracha, G., 2005. Java(TM) Language Specification, The (3rd Edition) (Java (Addison-Wesley)). s.l.:Addison-Wesley Professional.

ISO, 2013. Industry Foundation Classes (IFC) for data sharing in the construction and facility management industries, Geneva, Switzerland: s.n.

Lassila, O. & Swick, R. R., 1999. Resource description framework (RDF) model and syntax specification.

Leach, P. J., Mealling, M. & Salz, R., 2005. A universally unique identifier (uuid) urn namespace.

Oraskari, J. & Törmä, S., 2015. RDF-based signature algorithms for computing differences of IFC models, In Press. Automation in Construction, June.

Panaioli, F., 2005. RDFC14Ner. s.l.:s.n.

Pauwels, P. et al., 2011. A semantic rule checking environment for building performance checking. Automation in Construction, 20(5), pp. 506-518.

Pratt, M. J., 2001. Introduction to ISO 10303 - the STEP standard for product data exchange. Journal of Computing and Information Science in Engineering, 1(1), pp. 102-103.

Ragozin, A., 2015. topodiff - RDF model oriented diff algorithm - Google Project Hosting. s.l.:s.n.

Tekla Corporation, n.d. Tekla Structures BIM software | Tekla. s.l.:s.n.

Tummarello, G., Morbidoni, C., Puliti, P. & Piazza, F., 2005. Signing individual fragments of an RDF graph. s.l., s.n., pp. 1020-1021.

Fig. 6.6. Listing: Short Paths blank node naming algorithm

```
1   G ← G – {(x ? ?) ∪ (? ? x)| {x|(x rdf:type IfcOwnerHistory)}}
2   Ga ← Placement_checksums(G)
3   Ga ← Ga + CheckDependencies(G)
4   Ga ← Ga + Bottom-up_Checksums(G)
5   N ← blank nodes of G
6   For plen=1 to MAXPATH do
7     For all nodes n ∈ N do
8       P ← LWPS(n,plen)
9       P ← filter out paths of P that contain a cycle
10      csum ← MD5(Signature(P))
11      tmp ← empty list
12      For all points where paths in P cross do
13        tmp ← tmp + (position in the paths + path)
14      csum ← MD5(csum+sort(tmp))
15      csum ← MD5(csum+sort(Π_dep(n a:dep dep))) // the area id
16      csum ← csum + Π_pc(n a:pc pc) // the position information
17      Ga ← Ga + (n a:csum csum)
18    temp ← {x, where |(? a:csum Π_csum(x a:csum csum))|=1 in G }
19    For n ∈ temp do
20      n ← IRI(Π_csum(n a:csum csum))
21    N ← N – temp
22    if N is empty stop the loop
23  For all nodes n ∈ N do
24      for M← { m | (m a:csum Π_csum(n a:csum csum)) in G}
25        if all m in M have the same Π_bu(m a:bu bu)
26          m ← IRI(Π_csum(m a:csum csum))
27        else
28          m ← IRI(Π_csum(m a:csum csum)+ index)
29        N ← N - M
```

6.3 Discussion and Conclusions

This chapter focuses on the second part of the information security triad, integrity. I have concentrated on change detection. In the results presented in Sections 6.1 and 6.2, the computation performs very well for small changes, such as changes in the property values of ifcOWL Abox models. When it is sufficient to determine whether a model has been tampered with, it is most important to minimise false positives. The computational complexity increases exponentially with the size of the graph neighbourhood used for the computation. The dynamic neighbourhood size introduced in section 6.2 makes the computation much more efficient.

In the data integrity study the aim was to find an algorithm that creates unique identifiers for nameless nodes while minimising the neighbourhood size when using the neighbourhood to create names. Two improved algorithms for Incremental AlgSign were presented. The Short Paths Crossings Algorithm (SPCA) computes sets of paths of the limited length from nameless nodes, taking into account path crossings, and produces significantly smaller difference sets for IFC-derived graphs than existing algorithms for RDF change detection. The Short Paths algorithm produced smaller change sets than commonly available RDF diff tools such as topodiff, Jena and RDFContextTools RDFC14Ner. When diffing the largest models, it had better than average memory consumption, and the time consumption was usable for practical considerations.

I have discussed how to compute unique checksums for the triples of a graph node's neighbourhood when the length of the property paths emanating from the node is adjusted. I also considered whether certain properties can be excluded from the calculation and how this affects the results. For example, IFC GUID attributes have a big impact. They create a strong identity for a node, but if there are elements in the model that are generated each time the IFC model is exported from the native model, or if changes are made by deleting and creating an identical element

with a different ID, then excluding GUID values from the calculation will produce a more stable calculation.

According to information theory, the amount of information collected is strongly correlated and has a inverse functional relationship with the probability that the checksum generated from it will occur in a dataset. That is, when the probability of an incident decreases, the contained information increases and vice versa. As the collected amount of information cannot be infinite, it follows that the probability of the occurrence cannot never be zero. So, while we can create unique identifiers for a context and limited solution space of provided data, there is always a theoretical probability that another calculation will end up with the same checksum. This was a significant case to handle.

The calculation was tested with many automatically generated changes and a small example. The challenge is to get test data that corresponds to real life. Therefore, a significant result is that the difference between the models can always be calculated and the result depends only on the quality of the checksums obtained. That is, even if the identities were given random names, the difference could be computed, but the difference set would include all triplets involving the randomized names (in the exlusion of a random match). Similarly, if a theoretically optimal mapping of the elements is achieved, where complete naming could be generated without any contextual data from the nodes, the difference set would only cover triplets that express the actual change. In reality, the size of the change set lies somewhere between these. I have tried to develop methods to optimise the computation to minimise the number of extra triplets.

On the other hand, as I pointed out at the CIB W78 conference in Eindhoven in 2015, absolute coordinates allow for good geometric comparisons. In addition, recently I have also compared models statistically, according to the change in elements and property values, etc., when it was sufficient to compute the magnitude of the change.

I have discussed several different methods for computing the changes made to the model. I started with the challenging ifcOWL model problem of nameless nodes and progressed to the more practical problems of LBD models. As the geometric demo and IPFS methods show, the results were good.

In section 2.3, I showed how identical LBD models created with different converters can be compared programmatically with sufficient accuracy to allow the result to be used for unit testing of the converter. Similarly, in Section 7.1, I split the graph into stable parts. I have developed a version of Carroll's canonicalisation method [Carroll, 2003] where the result is a sufficiently stable Turtle syntax so that IPFS content-based addressing works across different IFC models, allowing model versions to share common parts of the model.

Chapter 7

Information Security Aspects: Availability

In this chapter, I address the research question: How information security aspects of data management related to availability can be handled.

Hypothesis 7.1 *Compared to using the Linked Data web protocols, sharing building data as a peer-to-peer, particularly the InterPlanetary File System (IPFS), improves the availability of information.*

I will introduce what support I have for the claims and discuss arguments that are against them.

Here, by the web, I refer to the common HTTP/1.1 protocol [Fielding et al., 1999], but the claims also apply to HTTP/2 and HTTP/3. I show how Linked Building Data can be implemented using the InterPlanetary File System (IPFS). It is also possible to implement it to fulfil the four design principles for Linked Data [Berners-Lee, 2006] listed in Section 1.3. The claim is addressed in Section 7.1.

Linked Data is based on using standard Web protocols – HTTP [Fielding et al., 1999, Belshe et al., 2015] in particular – whose advantage is its wide usage. The use of HTTP, however, creates a range of problems when the goal is to support machine-to-machine sharing of data:

1. *Limited availability*: Data accessibility may be contingent upon the operational status of the data-publishing server, e.g. the server publishing the data may be down or otherwise having technical issues, have insufficient bandwidth, or have other limited resources. The user may also have connectivity problems that limit the availability of data.
2. *Suboptimal performance*: When the same content is accessed repeatedly, previous access efforts could expedite subsequent access through content sharing among proximate users in the network, thus improving data accessibility (e.g., the network latency is typically smaller when distances are shorter.). In a peer-to-peer network, devices have equal status. They communicate and share resources directly without a central server. This approach can be beneficial when having a poor network connection. When TCP [Forouzan, 2002] is used to convey HTTP (HTTP 1.1 [Fielding et al., 1999], one dropped or lost packet means that the entire connection is halted until the missing packet is resent. [Ludin and Garza, 2017, Stenberg, 2019]
3. *Broken links*: A web link gets "broken" when its target is not accessible any more or points to changed content. The reason for this can be a renaming or disappearance of the domain or reorganising content within it — all of which are common occurrences on the Web. When a broken link is followed, the result is a technical or a semantical error instead of getting the expected web object.
4. *Lost history*: The identified object can be overwritten without providing access to the previous versions.

The effects of these problems are more dramatic when machines than when accessed by humans access data; usually, the former cannot recover from errors while the latter has flexible capabilities to adapt to them. A partial solution to some of these problems has been proposed. For instance, the performance and concurrency of transfer can be improved with the QUIC protocol that implements HTTP/3 [Stenberg, 2019] using the User Datagram Protocol (UDP) instead of TCP. Then, the Constrained Application Protocol (CoAP) of Bormann et al. approaches

the problem of providing Restful communication using UDP for lightweight devices [Bormann et al., 2012]. Protocols like WebSocket (WebRTC for UDP)[Sredojev et al., 2015] can be used in SPARQL queries, but they have limited use for retrieving facts about named things (see 1.3). Furthermore, Popitsch and Haslhofer [Popitsch and Haslhofer, 2010] presented a semi-automatic framework to detect and fix broken links. There are also many services available for checking websites broken links to support manual fixing. Access to historical data can be implemented with a combination of Web archives and the Memento protocol [Van de Sompel et al., 2009] that allows the user to view any web page as it was at a particular time in the past.

Figure 7.1 shows my article in this chapter at a glance.

Publication	Objectives	Method	Key findings
Oraskari et al., Federated Building Models in collaborative AEC/FM settings using IPFS., 2018 Oraskari et al., Sharing Linked Building Data in a Peer-to-Peer Network — ifcOWL meets InterPlanetary File System, 2023	• How could IPFS be used to publish Linked Building Data? • How to apply IPSF in this particular domain? • Are some specific problems encountered and how they could be solved? • What are the performance characteristics of the solution?	The Design Science Research Method: Experimental Controlled Experiment	• the steps needed for publishing an LBD (ifcOWL) model into IPFS • how to persent directory and version information of LBD in IPFS • The content in IPFS can be available even when the original building data publisher host is down.

Fig. 7.1. My article related to information security aspects of data management related to availability at a glance

7.1 Federated Building Models in collaborative AEC/FM settings using IPFS (ICCCBE 2018)

Imagine a situation where a facility managers office wins management services deal for a building, but the architectural office designed it is out of business. Luckily, a governmental office has automatically archived the models and now provides an addressed for the models on the peer-to-peer network. Thus, the facility manager can, by using the address, download the model.

Hypothesis 7.1 claimed that compared to using the linked data sharing building data as a peer-to-peer, particularly the InterPlanetary File System (IPFS), it improves the availability of information. The research presented in this section introduces my work on publishing building data into the IPFS network and my experiments around the availability. It is shown that data is available even when the original copy does not exist anymore. The limitations, i.e., the conditions for data availability, are discussed, and possible mechanisms to improve the existence of the copies in the network are considered.

Furthermore, it is shown that the performance (the publishing time, the query time, and notifications) is sufficient for communicating changes made to a building model. The quality of service was evaluated extensively. When the measurements were made, some outliers of the values (that is, longer waiting times or service failures) were measured. Considering the numerous generated files, the results were good. The typical scenario of using IPFS is sharing documents. In my work, I showed that it can also be used to share information more granularly, IFC element-based.

Table 7.1 summarizes the use of the design science research method in this research.

My contribution to the paper: The work is my contribution supervised by professor D.Sc. Jakob Beetz

and D. S. Seppo Törmä.

Table 7.1. The work of Oraskari et al. [2018a] in the Hevner et al. [2004] design science research framework

Guideline 1: Design as an artifact
I created methods to publish ifcOWL ABox data models and their versions in the InterPlanetary File System (IPFS) and to convey and propagate changes so that replicas of the original version could be kept in the network even if the original copy was deleted. The functionality and the data structures were presented at the conceptual level. A proof of construct was written in Java and Apache Jena framework for each test case.

Guideline 2: Problem relevance
As mentioned in Section 1.5, one aspect of data management is robust and scalable data preservation. Although a peer-to-peer is not foolproof (no method is. E.g., the Redundant Array of Independent Disks (RAID) storages provide only limited safety, although very high), replicas of the data can improve the probability of access in the long run. The benefit is that the storage parties may be independent. The question was how the AEC sector could benefit from that. Linked Building Data is based on the HTTP protocol, which makes it vulnerable to broken links, firewalls, slow connection speed, and server shutdowns. IPFS, as a peer-to-peer system, makes it possible to improve system performance in such conditions.

The constituent user community of the artefact is the researchers of the Linked Building Data Community and software architects of a Common Data Environment.
This work is an enabler for the Common Data Environment problem class, which explores ways to house construction project information to collect, manage, collaborate and share information on the organization of the artefacts.

Guideline 3: Design evaluation

Hevner et al. Design Research Evaluation Method:
In the Experimental Controlled Experiment, the test set of the Auckland Open Ifc Model collection was published in IPFS. In the Descriptive Scenario, IFCOpenHouse was edited and measured how quickly data change was transferred through the network. Also, the case where the original data content is deleted was tested. The results were evaluated against the expected change in the publication frequency of design models.

Guideline 4: Research contributions

The presented publication method and the implementations of the tests provide an example of how IFC data as RDF could be published over a peer-to-peer network and how to improve the probability that a copy would last the whole life-cycle of a building artefact.

Guideline 5: Research rigor

The work was peer-reviewed. The source code for the implementation and test data were made available in my GitHub repositories[1].

[1] https://github.com/jyrkioraskari/IFC-IPFS, https://github.com/jyrkioraskari/IPFSModifyTest1, and https://github.com/jyrkioraskari/ifcOWL-IPFS_Publish_test

Guideline 6: Design as a search process

There are many ways to publish the ifcOWL modes in IPFS. Productivity was favoured. So, the existing IFCtoRDF converter and Carroll's method to canonize the graph was used.

Guideline 7: Communication of research

International Conference on Computing in Civil and Building Engineering, ICCCBE, Tampere, 2018

The work was originally published in [Oraskari et al., 2018a]. The content presented here is based on our final draft of the article. The authors have kept the ownership of the work.

Jyrki Oraskari Department of Computer Science, Aalto University, Konemiehentie 2, 02150 Espoo, Finland, Jyrki.Oraskari@dc.rwth-aachen.de
Jakob Beetz Lehrstuhl CAAD Fakultät für Architektur, RWTH Aachen, SchinkelstraSSe 1, 52062 Aachen Deutschland, j.beetz@caad.arch.rwth-aachen.de
Seppo Törmä VisuaLynk Oy, Espoo, Finland, seppo.torma@visualynk.com

Abstract: The highly dynamic composition of actors, processes, and products in the building industry imposes significant challenges on the organization of data and information flows. The coordinated transmission and synchronization of information across organizational boundaries are facing severe scalability, reliability and security challenges. The concept of Linked Building Data has been identified as a potential means to overcome some of these challenges by shifting the paradigm of collaborative information creation from traditional file-based approaches towards dynamically composed graphs federated across networks. However, even though core technologies such as interoperable Building Information Models have reached the level of international standardization (ifcOWL), efficient, domain-specific methods and tools to process this information have received only a little attention so far. For example, in the context of Linked Building Data, the odds of a broken link can limit the potential of the Building Information Models.

In this context, the permanent web of data created by the InterPlanetary File System (IPFS) is promising. IPFS is a peer-to-peer distributed file system that uses cryptographic checksums of file content for addressing and linking data. It has its roots in GIT, BitTorrent, distributed hash tables, and Self-Certified Filesystems. IPFS addresses typical availability problems of Internet data, i.e., limited lifespan of the content, limited resources, blocking firewalls, and services that are out of business. IPFS has been designed to be interoperable with the existing Web protocols: it aims to put them on a more solid footing by replacing underlying technology layers with more advanced solutions.

In this paper, we analyze the benefits and limitations of IPFS technology to implement linked building data compared to RESTFul architectures. A use case in distributed collaborative settings in the AEC/FM sector using evolving multi-domain models are used to evaluate the work.

keywords: BIM, RDF, IPFS, IFC, ifcOWL, Linked Data, Web

7.1.1 Introduction

Construction industry is in the middle of *digitalization*: the field has been pervaded by digital technologies, albeit more slowly than other industries, but still in a fragmented and siloed manner [WEF, 2016]. Building design is increasingly carried out with BIM (Building Information Modeling [Eastman et al., 2011]) tools, resulting in multiple BIM models that specify the building from different perspectives: architecture, structures, and technical systems. In the construction stage, detailed planning, monitoring and tracking data are generated. Once the building is in use, multiple systems - building automation, asset and facility management, access control - generate streams of data about the use and conditions of a facility. In addition, growing amounts of relevant data have become available outside the facility itself: infrastructure models, urban plans, city models, product data, utility markets, or weather conditions.

Multiple parties — public and private – participate in the production of different aspects of this data over the lifecycle of a facility: owners, municipalities, designers, engineers, product vendors, fabricators, surveyors, inspectors, tenants, facility managers, asset managers, utility companies, and so on. Almost all parties are in a role of prosumers: they both consume data produced by others and produce new data related to their specific role (although this data is often not stored and shared due to the lack of proper infrastructure). Their roles are determined due to their distinct professional capabilities or resources; they have the position and typically the tools, expertise, and experience to generate specific data and to maintain it over time. Consequently, they are also assumed to have the ownership of the specific data they produce and to bear the legal responsibility for its content and correctness.

The upsurge of fragmented and heterogeneous data has created a need for sharing it efficiently with other parties. Traditional file exchange or centralized file or model repositories address only partially the information sharing challenges in decentralized organizations typical in construction industry, especially considering the changing organizations over a construction lifecycle. The parties face the difficulty to access information produced by others or to link it to other related information in meaningful manner. The consequence is a severe under-utilization of building data. The evolution of smart city, Internet-of-Things, BIM and GIS will in future call for open solutions to share information online so that other information can link to it in a granular manner.

Linked Building Data [Pauwels et al., 2015, Pauwels, 2014b] is a decentralized approach to information sharing: building data is published on the Web using the Web of Data technologies [Bizer et al., 2008, 2009]. Data is transformed into

specific type of graphs, using format of RDF (Resource Description Framework). Each building object in the graph has its own Web address (URI); using its URI the description of the object can be retrieved over the Web as structural data (e.g., in JSON or XML). The type and property names used in the description are defined in ontologies, for instance, in ifcOWL [Terkaj and Pauwels, 2014], the ontology version of the IFC. The descriptions can also contain links - Web addresses of related objects - that can be followed to gather further information about the object and its context. As in the regular Web, the links can point to other objects within a same model or objects residing in other hosts on the Web. The advantages of Linked Building Data are that the models are online, objects can be accessed in a granular manner, and objects can be linked with external data using their addresses. Data can be managed in a decentralized manner, respecting the digital sovereignty of each party.

The publication of building data as RDF graphs has been enabled by the DRUMBEAT platform [Hoang and Törmä, 2017]. It takes care of the conversion of data from IFC to RDF according to ifcOWL and the storage of the resulting graphs in an RDF database, and allows the data to be accessed and managed through a REST API.

However, this approach is vulnerable to known weaknesses of the Web infrastructure. The network protocols used by the Web are based on the client/server architecture. In most setups, TCP is used to convey the HTTP messaging of the Web. TCP limits downloading of a file from a data provider to a single connection at a time; this not only makes the source of the data a single point of failure but also prolongs the network connections and consequently makes them more error prone. Moreover, Web suffers from the problem of broken links, that is, links to Web resources that no longer exist because they may have been moved, there could be a blocking firewall, or the service or business has been terminated.

IPFS (InterPlanetary File System) [Benet, 2014] is a solution for the weaknesses of the current Web. IPFS is a peer-to-peer distributed file system that uses cryptographic checksums of file content for addressing and linking data. It has its roots in GIT, BitTorrent, distributed hash tables, and Self-Certified Filesystems. IPFS addresses typical availability problems of Internet data, i.e., limited lifespan of the content, limited resources, blocking firewalls, and services that are out of business. IPFS has been designed to be interoperable with the existing Web protocols: it aims to put them on a more solid footing by replacing underlying technology layers with more advanced solutions.

In this paper we study if IPFS could be used to publish Linked Building Data as well. How to apply IPSF in this particular domain? Are some specific

problems encountered? How could they be solved? What are the performance characteristics of the solution?

We will present an implementation of Linked Building Data using IPFS. The specific problems deal with the modifiable data: how to manage the publication of new versions of models or new linksets in a manner that keeps the references correct. We will describe a solution based on a directory structure that maps object identifiers to their IPFS addresses. The performance evaluation shows that data publication is the most time-consuming phase of processing, but the access to data is significantly faster over the long run. The data is much less vulnerable to unavailability problems.

The structure of the paper is as follows: In Section 2, the state of the art of publishing interlinked building data on the web and technologies are presented. That is followed by a description of the IPFS storing framework. Section 4 outlines the implementation of our proposed method. The applicability of the IFC/IPFS publication and the empirical results of the exercise are described in section 5. The paper is concluded summarizing the findings made.

7.1.2 State of the Art

BIM models have traditionally been shared through point-to-point file exchange. While it is a decentralized and flexible approach, it has several well-known drawbacks. First, it works at the level of complete models — typically of size between 20-200MB — that are slow to open and process with related tools. Second, it tends to be applied in an ad hoc and disorganized manner, leading to confusion over who has which version of a model. Third, it does not offer any support for the interlinking of different models. The container exchange is an approach to relieve the latter two problems. Multiple interrelated models are exchanged in one package (e.g., a zip file) with link models connecting objects within models.

Linked Building Data

Competing Techniques

Unlike in the peer-to-peer file sharing systems where the center is in replicating content, in this work we concentrate on data availability. So, while BitTorrent [Cohen, 2003, Izal et al., 2004] optimizes the network speed making downloader reciprocally share their bandwidths, distributed data storages may use the same

techniques, but they focus on ensuring the accessibility of the data at a certain timespan. In general, there are two major types of distributed data storages: file-based and database systems. IPFS is mainly file based as blockchain [Nakamoto, 2008] file storage systems like Storj [Wilkinson et al., 2014], and Filecoin [Protocol Labs, Inc and Labs, 2017]. Apache Cassandra [Cassandra, 2015] is an example of open distributed NoSQL databases. It is highly efficient, but it shares the common problem of distributed open databases: it is prone to malicious edits. Furthermore, [Noiumkar and Chomsiri, 2014] pointed out, that in particular, Cassandra saved files unencrypted, and, since it uses single thread architecture, it is prone to denial of service attacks.

Figure 7.2 summarizes points presented in [Kochin, 2017]. As seen, (1) if we expect no hash collisions [Peyravian et al., 1998] in the content-based addressing, IPFS is protected against malicious edits [Third and Domingue, 2017]. (2) It is free, and (3) and a strong legal point is: it makes host to save only data they need, including the address meta-data. Its major drawback is that data is not forced to be replicated. In the following chapters, we will show how this can be tacked. On the other hand, that is exactly why Filecoin [Protocol Labs, Inc and Labs, 2017] has been developed. It is an extension to IPFS, a decentralized storage network that uses blockchain to incentive people to save data on their free storages.

7.1.3 InterPlanetary File System (IPFS)

InterPlanetary File System (IPFS) [Benet, 2014] is a content addressable decentralized and peer-to-peer distributed storage net. It this study the most relevant techniques in IPFS are in routing, data addressing, and change notification for the interested parties.

In the IPFS terminology, a node is a participating computer on the distributed hash table (DHT) [Liu, 2006] network. In particular, IPFS uses both Coral [Freedman and Mazieres, 2003] sloppy hash table (DSHT) and Kademlia [Maymounkov and Mazieres, 2002] DHT as a distributed address book to find references to computers hosting a piece of data. While Kademlia searches concurrently and asynchronously favoring reliable nodes, uses lookups messages to configure the network, and routes using XOR distances [Maymounkov and Mazieres, 2002] iteratively to addresses on neighbors of a given node, Coral clusters nodes using network round-trip time to the client, and includes a mechanism to avoid overloading a node. [Zhang et al., 2013] Using the DHT techniques gives the network desirable qualities. It stands high network traffic, which is characteristic for denial of service attacks, server downtimes, shares network load, and minimizes network latency.

In contrast to the push transfer mode of the peer-to-peer (P2P) data sharing where the sender takes the initiative to push content to the client [Li, 2008], IPFS share only data that the user has opted in. That is, it limits data sharing to the content the user had published and cached data that the user has intentionally downloaded from the net.

The principle is a double-edged sword. The user keeps control of the shared data and so, does not face the risk of sharing any illegal content unintentionally, which makes the solution qualify for business use. On the other hand, it, and the active garbage collection policy limit the number of copies in the net efficiently and thus, the lifetime of the data.

Using the cryptographic hash of the data objects as a key to search data makes data immutable. Files can only be altered by publishing a modified version of the content.

7.1.4 Implementation

This section describes the implementation of the IFC/IPFS prototypical engine. The work is twofold. Section 4.1 describes how an IFC model is saved in IPFS network. In section 4.2, restful access to the models and how separate models can be interlinked are presented.

Publishing IFC into IPFS network

One of the core ideas of IPFS is that data with the same content can be unambiguously fetched from the net using its cryptographic hash. So, we had two criteria for the design. The first criterion was that the triples that form a description of an entity do not contain any metadata that is version specific. The second was that the streamed representation is the same when no modification of the element has been made.

The assumptions were: (a) when an IFC/BIM model is changed, the model can be divided into parts so that the change will be isolated in some portions. (b) Globally unique identifiers (GUID) can be expected to be constant, and they represent logical blocks in the model graph, the elements. Thus they can be used to cut the graph into groups of triples that can be referred by using the GUID.

The above-listed criteria led us to a solution where the node that has a GUID as an

attribute is converted into a resource with the GUID as part of its URI. All the other nodes are made nameless (but they still have an identity), and each element graph, a triple set, is converted into the standard form that was presented in [Carroll, 2003]. Jeremy Carrolls RDF C14N canonization algorithm sufficiently guarantees the same serialization of small triple sets when they are representing the same content.

Linked data access to the models

7.1.5 Results

A selected set of IFC models of *Open IFC Model Repository*[2] was used in the tests. The models are listed in Table 7.2.

Computers that were used in the test were (a) Intel i7-766U CPU, 2.50Ghz, with 16 GB RAM, Surface Pro, and (b) Intel i5-6300 CPU 2.40 Hhz, with 8.0 GB RAM, Surface Book. In the test, Surface Pro initiates all transactions. All code in the test setup was written in Java [Gosling et al., 2005] programming language version 8. Jena library[McBride, 2002] versions 2.6.4 and 3.1.1 were used for RDF processing. All the source code was made available at the project GitHub repository[3].

As depicted in Fig.7.3, the phases of the IFC to IPFS publication is a straightforward data flow [Sousa, 2012] at the concept level. The inputted IFC/STEP formatted [Pratt, 2001] model is converted to standard IfcOWL using IFCtoRDF [MultiMediaLab, UGent, 2013]. That is followed by steps that extract triples that form the IFC entity. They are sorted an formatted to ensure an identical representation for identical content. Finally, that is published using IPFS Java API [Preston et al., 2019] and [Protocol Labs, Inc, 2018].

Figure 7.4 shows the relative time usages of the conversion steps during the IFC data IPFS publication process. The times are wall clock times. In the graph, the models are listed in the order of the actual triple count, i.e., models without geometry and owner history data. As seen, relatively substantial proportions of the time were used for IFCtoRDF conversion when smaller models were handled.

There is a vague relation between the count of the published RDF triples (Fig. 7.5) and the total time needed for them to be added into IPFS. However, as seen in the coefficient of determination of the linear regression formula in Figure 7.6, there

[2] "http://openifcmodel.cs.auckland.ac.nz/"
[3] https://github.com/jyrkioraskari/IFC-IPFS

	Model ID	IfcOWL triples	Filtered RDF triples	Entitys
1	NIST: ec-001	375	276	9
2	IAI: 10--001	1200	238	18
3	IAI: 13p-001	3210	434	34
4	IAI: 9-0-001	5262	277	17
5	IAI: 1-0-001	16877	1694	97
6	NIST: AI-001	34331	34030	1062
7	NIST: Be-001	48514	53215	1269
8	NIST: De-001	57733	42676	1128
9	KIT: PAR-001	103440	1407	31
10	Four sto-001	228303	221916	1465
11	NIST: an-001	291613	267159	13500
12	KIT: AC1-001	347443	283720	6565
13	KIT: PAR-001	382771	751	13
14	KIT: AC--001	446461	247264	4823
15	KIT: AC1-001	603847	32824	873
16	KIT: All-001	745512	91244	5422
17	West Riv-001	750664	655639	8988
18	KIT: FZK-001	961770	63373	457
19	KIT: Nem-001	1396465	4514	289

Table 7.2. The models that were used in the IPFS publication test.

is a strong correlation between the entity count and the time used to publish the data in IPFS Merkle nodes. A similar correlation is between the entity count and the total publication time (Fig.7.7).

This implies three things: (1) the number of Merkle nodes that are generated when an IFC entity is saved to IPFSis relatively constant. (2) When we know the entity count of an IFC model, we can predict with a high confidence the needed time for the publication, and (3) as we get small publishing times for medium size models here, we can expect reasonable time usage for a hundred times larger model. To practically ensure the above-stated correlation, IfcOpenHouse[4] sample model was published. For each entity, Merkle nodes were counted. Without exception, all entity data sets fitted in a single Merkle node.

While all IPFS hosts communicate to the Internet as an active element of the

[4]https://github.com/aothms/IfcOpenHouse

distributed hash table network graph, it is not apparent how much the speed of the Internet connection affects the IPFS performance. Thus, *NetBalancer*[5] was used to evaluate the network usage of the ipfs.exe process during the publication of *NIST: an-001* model to IPFS. The reporter download count was 189MB and upload 42MB. Then, the same IFC to IPFS publication test was repeated using two network setup: Wi-Fi with estimated[6] down and uplinks of 27Mbs, and LAN with down and uplinks of 98Mbs. The resulted values form almost identical pattern in Figure 7.6.

In IPFS, all editions of a file have a unique address. To be able to address the content even after a change, we need to be able to address the newest version. IPNS is a technique in IPFS to associate a mutable node, i.e., a public key or an identity, with any IPFS content. That was tested using the test set. The average update took 32 seconds, while standard deviation was 18.8 seconds.

DNS [Cheshire and Krochmal, 2013] record update was made to three domain names fractuscan.com, second.fractuscan.com, and ipfs.fractuscan.com. Although the worldwide DNS update propagation is expected to take as long as 24 to 72h worldwide, IPFS saw all three entries in seconds. However, when edited again, none of them got further updates on the one hour test time. Cleaning Windows DNS cache did not affect. Thus, the technique can be used for publishing mutable nodes that have been published using IPNS, but it is not suitable for more frequent changes.

A more realistic use case was used to prove that the stored IFC data in IPFS is usable also in real life situations. In the scenario, a construction company wishes the future residents to take part in the design process. When they want to have comments about the new design details of an element, they send a message to an IPFS publishsubscribe channel where any interested parties could register themselves. So, the model owner sends a message that contains IPFS Merkle node hashes for the model directory and the element. Both are assigned to give the receivers the addressed content and a way to get the model context of the component. In the test, the element with GUID ad8339ce-4378-4e54-b917-ed305724ca9b of the IFCOpenHouse sample model was used as an example, but instead of using that as is, a timestamp was added to the content to ensure that each test round has a unique but comparable set of data and thus the effect of caching was eliminated.

Table 7.3 shows statistics of the sample model.

The first test set was made between two software processes communicating through IPFS. Table 7.4 shows the results. In this, the process A got an almost

[5]https://netbalancer.com/
[6]http://www.speedtest.net/run

	IrcOwl triple count	Filtered triple count	Entities
IfcOpenHouse	20041	624	76

Table 7.3. The IfcOpenHouse data set

immediate (on average 88ms) a response from process B. It looks like created new Merkle nodes were read back directly from the IPFS cache. Tables 7.5 and 7.6 show more realistic cases. The first one is a test setup where two computers communicate through the local Wi-Fi 2.4Ghz 802.11n network. IPFS reported 9ms ping time between nodes which means that the latency of the communication was very low. The latter test was the most realistic one and emulates a situation where people from the same city are are communicating. The average reported ping time was already 170ms and the average responses were got in 4 seconds.

Same computer setup Process A	Node to IPFS	Directory to IPFS	Remote answered	Answer read from IPFS
Average time (ms)	63.9	80.2	88.0	14.3
Stdev	36.6	25.8	101.8	7.4
Median time (ms)	46.0	74.5	57.0	13.0
Min time (ms)	32.0	51.0	28.0	5.0
Max time (ms)	142.0	125.0	371.0	27.0

Remote Process B	Node from IPFS	Node to IPFS
Average time (ms)	25.7	51.5
Stdev	62.9	27.0
Median time (ms)	4.0	47.0
Min time (ms)	2.0	25.0
Max time (ms)	204.0	111.0

Table 7.4. IPFS: Sofware process case measurement

Average Ping time from B to A			9 ms	
Computer A	Node to IPFS	Directory to IPFS	Remote answered	Answer read from IPFS
Average time (ms)	42.8	69.8	504.2	118.1
Stdev	5.1	11.7	275.1	26.4
Median time (ms)	41.5	67.0	358.0	117.5
Min time (ms)	38.0	50.0	280.0	83.0
Max time (ms)	52.0	95.0	1068.0	166.0

Remote Computer B	Node from IPFS	Node to IPFS
Average time (ms)	302.9	64.5
Stdev	225.7	11.1
Median time (ms)	199.5	60.5
Min time (ms)	148.0	53.0
Max time (ms)	742.0	89.0

Table 7.5. IPFS: Local case measurement

Because the response times were only seconds or parts of seconds, in the commenting use case, this would allow many polls per an hour, while a few would suffice for a day.

One of the selling points of IPFS has been the ability to create a datastore which is available even when the original data provider is out of business, or a massive denial of service attack makes it unreachable. IPFS do temporarily cache all downloaded content and share them when needed. It also offers a tool, ipfs-cluster [Sanjuan, 2018], to builds up clusters that guarantee that at least certain number of copies are online. However, to answer what it takes for individuals that are interested in the specific building information content to replicate the model data, a test setup was made. In that, when a version of a bim model was released into IPFS, the Merkle hash of the corresponding directory was sent online using an IPFS public-subscribe broadcast. All individual subscribers reacted the message, read the directory and forced their cache to save the content permanently and make it available. In the test, the IfcOpenHouse test content was replicated on average 66.2 seconds. When considering the 75 to 85 years long lifecycle time of a building

Average Ping time from B to A	170 ms			
Computer A	Node to IPFS	Directory to IPFS	Remote answered	Answer read from IPFS
Average time (ms)	32.4	46.8	4061.3	1651.7
Stdev	3.7	2.3	2085.3	682.8
Median time (ms)	31.0	48.0	4180.0	1602.0
Min time (ms)	28.0	42.0	1160.0	718.0
Max time (ms)	39.0	50.0	6507.0	2883.0

Remote Computer B	Node from IPFS	Node to IPFS
Average time (ms)	2174.6	41.7
Stdev	1490.1	5.0
Median time (ms)	1681.0	40.0
Min time (ms)	513.0	36.0
Max time (ms)	5597.0	51.0

Table 7.6. IPFS: Network case measurement

[Lippke et al., 2004], and that new major releases are not usually published every day, the needed time was quite reasonable. Furthermore, a short test was made to ensure that IPFS can provide data from the cached copied even if the original provider is down. That was the case.

7.1.6 Discussion and Conclusions

IPFS is a new peer to peer protocol and network that evolves at a fast pace. The time delays seen in the experiments are probably better tomorrow. However, even the current measures permit us implementing applicable status monitors described in [Torma, 2013], and co-operative design systems. On the other hand, data availability is the key feature that makes IPFS stand out in favour. It was shown that the data content was available even when the original building data publisher host was down. Even if linked data can be copied and archived, IPFS is superior in the sense that (1) highly interesting data is cached automatically, (2) replication can be arranged without any organizing, it suffices that the data is made available

for anybody who is interested in the content, and (3) all copies are addressed using the same means and pointer as for the original the piece of information.

The network is dynamic and learns from the send data. It was seen during the tests that shutting down an IPFS node made the next connections times longer and for a period already functioned connection could fail. It is not a surprise in a network where communication continually shapes the topology, and the system favours nodes that are reliable, communicate reciprocally, and have a low network latency. It recovered every time.

IPFS

Pros

- No need of trust, since data is addressed by content
- Each user host only data that it needs
- Supports GIT like version control
- Limited protection against data accessibility failures
- Resistance of flooding

Cons

- Data storage by others is not guaranteed
- Deleting is not supported
- Stored data is immutable
- No replication of data without a request

Blockchaing based file storage

Pros

- No need of trust, since data is addressed by content
- Financial motivation to store data
- Supports GIT like version control
- High availability of data content
- Deleting is possible

Cons

- Storage is not free
- Stored data is immutable
- Stores only files

Open Distributed NoSQL databases

Pros

- Linear scaling of the speed and storage capacity
- High availability of data content
- Mature realizations

Cons

- The need to trust the peers

Fig. 7.2. The summarized comparison of decentralized storage systems

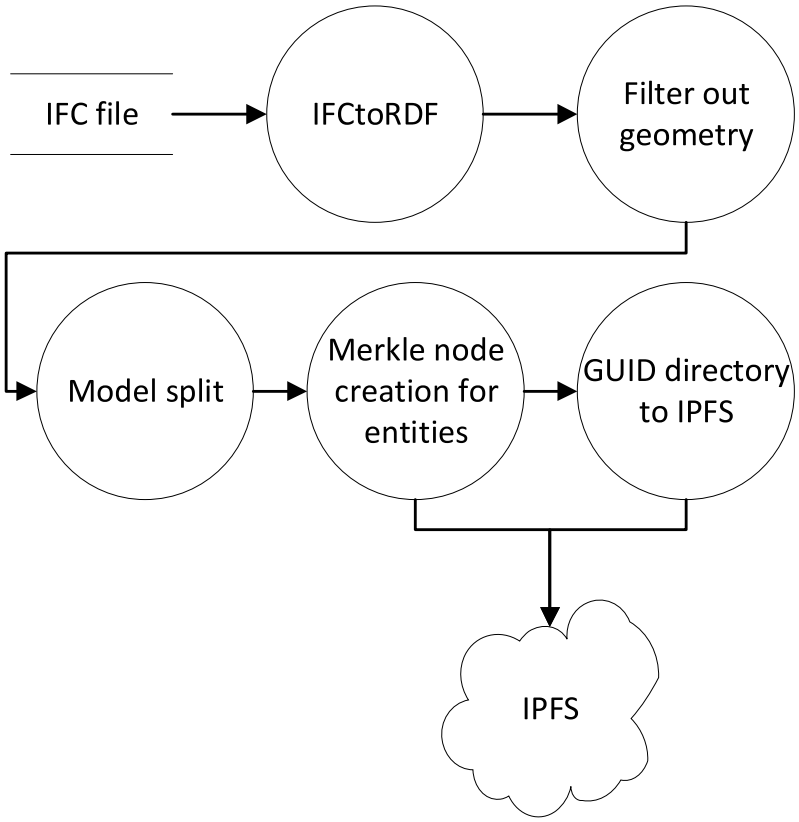


Fig. 7.3. The publication process for a new IFC file.

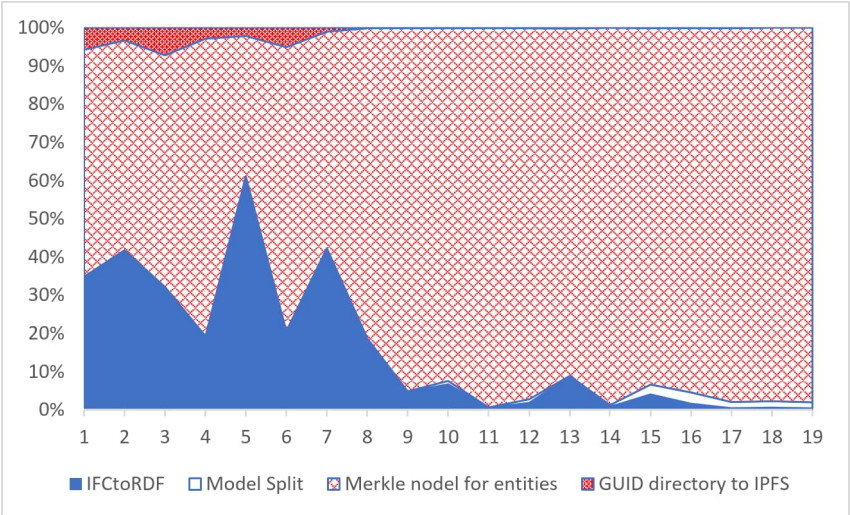


Fig. 7.4. The proportional time usage in publishing an IFC model.

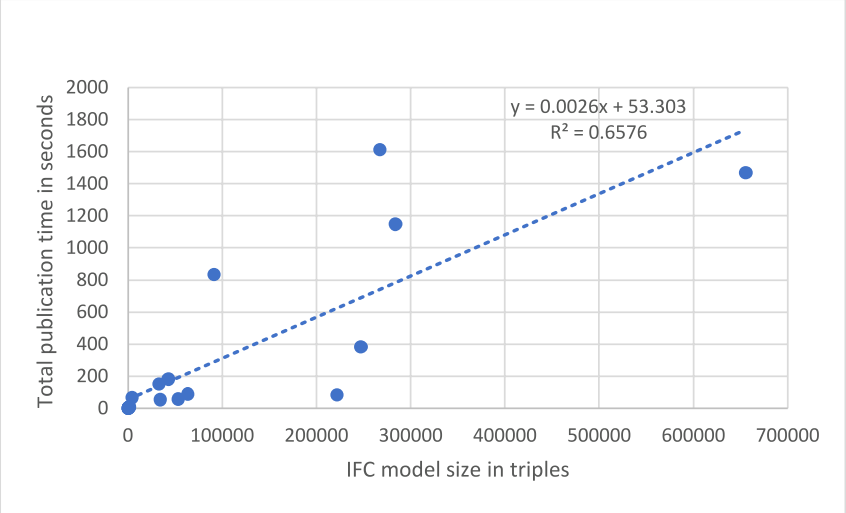


Fig. 7.5. The Filtered triple count versus the total IFC publication time

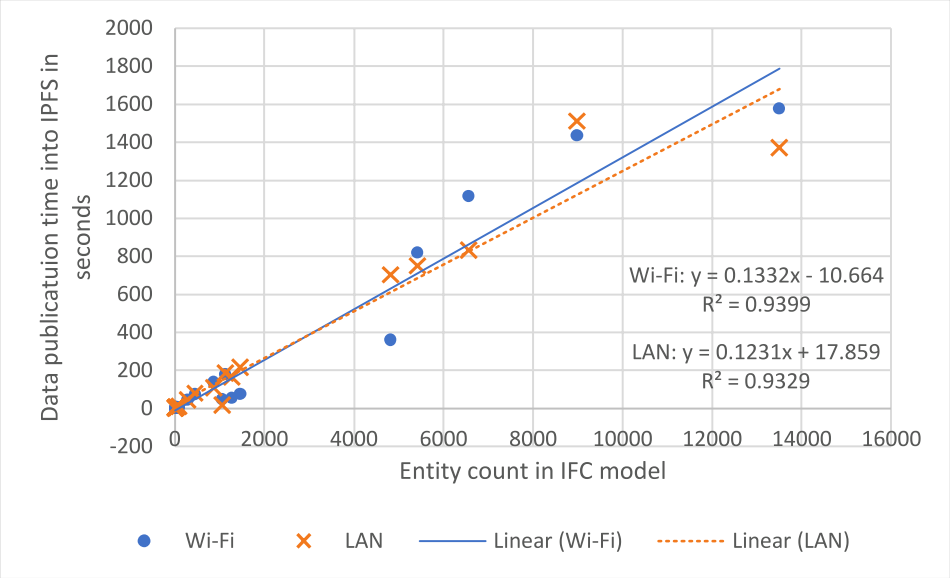


Fig. 7.6. The model entity count versus the IPFS nodes creating time

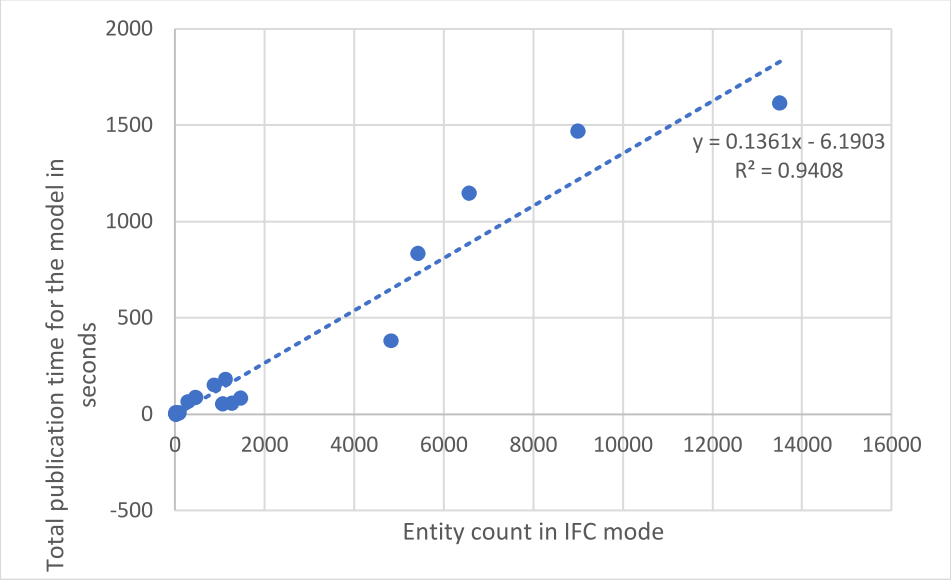


Fig. 7.7. The model entity count versus the total IFC publication time

References

Juan Benet. Ipfs-content addressed, versioned, p2p file system, July 2014. URL https://arxiv.org/abs/1407.3561. Accessed on Jun. 27. 2019.

Christian Bizer, Tom Heath, Kingsley Idehen, and Tim Berners-Lee. Linked data on the web (ldow2008). In *Proceedings of the 17th international conference on World Wide Web*, pages 1265–1266. Association for Computing Machinery New York, NY, United States, 2008.

Christian Bizer, Tom Heath, and Tim Berners-Lee. Linked data-the story so far. *Semantic services, interoperability and web applications: emerging concepts*, pages 205–227, 2009.

Jeremy J Carroll. Signing rdf graphs. In *International Semantic Web Conference*, pages 369–384. Springer, 2003. https://doi.org/10.1007/978-3-540-39718-2_24.

Apache Cassandra. Apache cassandra. *Google Scholar*, 2015.

Stuart Cheshire and Marc Krochmal. DNS-Based Service Discovery. RFC 6763, February 2013. URL https://www.rfc-editor.org/info/rfc6763.

Bram Cohen. Incentives build robustness in bittorrent. In *Workshop on Economics of Peer-to-Peer systems*, volume 6, pages 68–72, 2003.

Chuck Eastman, Paul Teicholz, Rafael Sacks, Kathleen Liston, BIM Discover, and BIM Handbook. *BIM Handbook: A Guide to Building Information Modeling for Owners, Managers, Designers, Engineers, and Contractors*. John Wiley & Sons, 2011.

Michael J Freedman and David Mazieres. Sloppy hashing and self-organizing clusters. In *International Workshop on Peer-to-Peer Systems*, pages 45–55. Springer, 2003.

James Gosling, Bill Joy, Guy Steele, and Gilad Bracha. *Java(TM) Language Specification, The (3rd Edition) (Java (Addison-Wesley))*. Addison-Wesley Professional, 2005. ISBN 0321246780.

N Vu Hoang and S Törmä. Drumbeat platform - a web of building data implementation with backlinking. In *eWork and eBusiness in Architecture, Engineering and Construction: ECPPM 2016: Proceedings of the 11th European Conference on Product and Process Modelling (ECPPM 2016), Limassol, Cyprus, 7-9 September 2016*, page 155. CRC Press, 2017.

Mikel Izal, Guillaume Urvoy-Keller, Ernst Biersack, Pascal Felber, Anwar Al Hamra, and Luis Garces-Erice. Dissecting bittorrent: Five months in a torrents lifetime. *Passive and Active Network Measurement*, pages 1–11, 2004. Springer.

Dmitry Kochin. Tiesnetwork/ties-docs, Sep 2017. URL https://github.com/TiesNetwork/ties-docs/wiki/Where-do-decentralized-applications-store-their-data%3F. [Online; accessed Jun. 06. 2019].

Jin Li. On peer-to-peer (p2p) content delivery. *Peer-to-Peer Networking and Applications*, 1(1):45–63, 2008. Springer.

Bruce Lippke, Jim Wilson, John Perez-Garcia, Jim Bowyer, and Jamie Meil. Corrim: Life-cycle environmental performance of renewable building materials. *Forest Products Journal*, 54(6):8, 2004. Forest Products Society.

Wenguang Liu. Distributed hash table. https://wenguang-liu.github.io/asserts/DistributedHashTablePPT.pdf, 2006. Online; accessed Oct. 17. 2023.

Petar Maymounkov and David Mazieres. Kademlia: A peer-to-peer information system based on the xor metric. In *International Workshop on Peer-to-Peer Systems*, pages 53–65. Springer, 2002.

Brian McBride. Jena: A semantic web toolkit. *IEEE Internet computing*, 6(6):55–59, 2002. Institute of Electrical and Electronics Engineers.

MultiMediaLab, UGent. Ifc-to-rdf converter service, 2013.

Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system, 2008. URL https://bitcoin.org/bitcoin.pdf. Online; accessed Oct. 15. 2023.

Preecha Noiumkar and Tawatchai Chomsiri. A comparison the level of security on top 5 open source nosql databases. In *The 9th International Conference on Information Technology and Applications (ICITA2014)*, 2014.

Pieter Pauwels. Supporting decision-making in the building life-cycle using linked building data. *Buildings*, 4(3):549–579, 2014. Multidisciplinary Digital Publishing Institute.

Pieter Pauwels, Seppo Törmä, Jakob Beetz, Matthias Weise, and Thomas Liebich. Linked data in architecture and construction. *Special Issue Automation in Construction, erscheint Juni*, 2015.

Mohammad Peyravian, Allen Roginsky, and Ajay Kshemkalyani. On probabilities of hash value matches. *Computers & Security*, 17(2):171–176, 1998. Elsevier.

Michael J Pratt. Introduction to iso 10303 - the step standard for product data exchange. *Journal of Computing and Information Science in Engineering*, 1(1): 102–103, 2001. American Society of Mechanical Engineers.

Dr Ian Preston, Chuck LeDuc Díaz, Grégoire Jeanmart, Thomas Diesler, Furkan KAMACI, Piero Ottuzzi, Mykola Nikishov, Richard V., Richard Littauer, Nils Strelow, Luís Bastião, and David Dias. jyrkioraskari/java-ipfs-http-client: java-ipfs- http-client DOI reference, June 2019. URL https://doi.org/10.5281/zenodo.3240413.

Protocol Labs, Inc. ipfs/go-ipfs, Jan 2018. URL https://github.com/ipfs/go-ipfs. Accessed on Feb. 15. 2018.

Protocol Labs, Inc and Protocol Labs. Filecoin, July 2017. URL https://filecoin.io/filecoin.pdf. Accessed on Jun. 06. 2019.

Hector Sanjuan. ipfs/ipfs-cluster, Jan 2018. URL https://github.com/ipfs/ipfs-cluster. Online; accessed Oct. 15. 2023.

Tiago Boldt Sousa. Dataflow programming concept, languages and applications. In *Doctoral Symposium on Informatics Engineering*, volume 130, pages 323–334, 2012.

W. Terkaj and P. Pauwels. Ifcowl ontology file for ifc4, 2014. URL https://standards.buildingsmart.org/IFC/DEV/IFC4/ADD2_TC1/OWL/index.html. Online; accessed Oct. 15. 2023.

Allan Third and John Domingue. Linkchains: Exploring the space of decentralised trustworthy linked data. https://ceur-ws.org/Vol-1934/contribution-06.pdf, 2017. ISWC2017 workshop on Decentralizing the Semantic Web. Online. Accessed on Oct. 17. 2023.

Seppo Torma. Semantic linking of building information models. In *Semantic Computing (ICSC), 2013 IEEE Seventh International Conference on*, pages 412–419. IEEE, 2013.

WEF. Shaping the Future of Construction: A Breakthrough in Mindset and Technology. Industry agenda, World Economic Forum, Geneva, May 2016.

Shawn Wilkinson, Jim Lowry, and Tome Boshevski. Metadisk a blockchain-based decentralized file storage application. Technical report, Technical Report, Available: http://metadisk.org/metadisk.pdf, 2014.

Hao Zhang, Yonggang Wen, Haiyong Xie, and Nenghai Yu. A survey on distributed hash table (dht): Theory, platforms, and applications, 2013. URL https://pdfs.semanticscholar.org/8859/3bf30b9de1497ba22b1136ebc433f648bbc8.pdf. Accessed on Jun. 06. 2019.

7.2 Sharing Linked Building Data in a Peer-to-Peer Network — ifcOWL meets InterPlanetary File System (J. Comput. Civ. Eng.)

Imagine a facility manager visiting a building to check an office's roof window. In the old large building block, the network coverage is poor. She opens her innovative facility management program to view the detailed technical information of the room. Her laptop does not connect to the internet and does not have the needed data, but some office workers in the building have accessed that information earlier. So, the intelligent system fetches the needed piece of information from the office users' computers nearby and shows the needed knowledge to her. Also when she later gets an expensive and slow mobile connection functioning,

Hypothesis 7.1 claimed that compared to using the linked data sharing building data as a peer-to-peer, particularly the InterPlanetary File System (IPFS), it improves the availability of information. The research presented in this section continues the work done in Section 7.1. The focus is on the benefits of sharing content with the computers of the same local network. The work shows the performance increase in data model download, which time wise increases the data availability.

It was shown that saving the data to the IPFS network outperformed when compared with the usual Linked Data publication: (1) when there is another copy/replica of the data in the network, the data was shown to be still available, and (2) the nearby extra copies were shown to improve the performance of the data fetch.

Table 7.7 summarizes the use of the design science research method in this research.

My contribution to the paper: The work is my contribution supervised by professor D.Sc. Jakob Beetz and D. S. Seppo Törmä.

Table 7.7. The work of Oraskari et al. [2018a] in the Hevner et al. [2004] design science research framework

Guideline 1: Design as an artifact
I created methods to divide and publish ifcOWL ABox data models and their versions in the InterPlanetary File System (IPFS) and a test setup (computers and programs to run the test) for testing the performance. The functionality and the data structures were presented at the conceptual level. A proof of construct was written in Java and Apache Jena framework for each test case.
Guideline 2: Problem relevance
As mentioned in Section 1.5, one aspect of data management is a robust and scalable data preservation. Replicas of the data can improve the performance of the data access. The constituent user community of the artefact is the researchers of the Linked Building Data Community and software architects of a Common Data Environment. This work is an enabler for the Common Data Environment problem class, which explores ways to house construction project information to collect, manage, collaborate and share information on the organization of the artefacts.

Guideline 3: Design evaluation

Hevner et al. Design Research Evaluation Method:
In the Experimental Controlled Experiment, in the first part of the work, the Auckland Open Ifc Model collection was used to evaluate the algorithms to divide the IFC models into stable parts. The best one and the results are reported n the article. In the second part of the work, a test model was fetched serially and concurrent piece by piece from a virtual host located in London to Espoo, Finland, using IPFS. Also, the case where the original data content is deleted was tested.

Guideline 4: Research contributions

The presented publication method and the implementations of the tests provide an example of how IFC data as RDF could be published over a peer-to-peer network and how it can be used to increase the performance of the data access. In the implementation, there were three innovative aspects: first, how to split ifcOWL graphs into parts that are addressable in IPFS, how to serialize the parts in the content addressable network, and how to catalogue the data so that it can be found.

Guideline 5: Research rigor

The work has been peer-reviewed. The source code for the implementation and test data were made available in my GitHub repositories[7].

Guideline 6: Design as a search process

There are many ways to publish the ifcOWL modes in IPFS. Productivity was favoured. So, the existing IFCtoRDF converter and Carroll's method to canonize the graph was used.

[7] https://github.com/jyrkioraskari/IFC-IPFS, https://github.com/jyrkioraskari/IPFSModifyTest1, and https://github.com/jyrkioraskari/ifcOWL-IPFS_Publish_test

Guideline 7: Communication of research

The publication was accepted to the American Society of Civil Engineers Journal of Computing in Civil Engineering in 2023. DOI: 10.1061/JCCEE5/CPENG-5381

The content presented here is based on our final draft of the article published originally in [Oraskari et al., 2024]. ASCE copyrighted content includes all ASCE publications. "Thesis or Dissertation: Reuse is permitted, provided the content in total does not account for more than 25% of the new work." [8]

Jyrki Oraskari Department of Computer Science, Aalto University, Konemiehentie 2, 02150 Espoo, Finland, Jyrki.Oraskari@dc.rwth-aachen.de
Seppo Törmä VisuaLynk Oy, Espoo, Finland, seppo.torma@visualynk.com
Mehmet Yalcinkaya VisuaLynk Oy, Espoo, Finland, mehmet.yalcinkaya@visualynk.com
Vishal Singh Indian Institute of Science. singhv@iisc.ac.in
Jakob Beetz Chair of Design Computation, Fakultät für Architektur, RWTH Aachen, Deutschland, RWTH Aachen, Germany, j.beetz@caad.arch.rwth-aachen.de

Abstract: The recent and significant change in the Architecture Engineering and Construction (AEC) industry is the increasing use of Building Information Management (BIM) tools in design and how the generated digital data is consumed. While the designers have primarily published blueprints and model files, multiple parties are now involved in producing different aspects of the data and may be consuming data produced by others. This evolution has created new challenges in sharing and synchronising information across organisational boundaries. Linked Building Data is a community effort identified as a potential means of overcoming the barriers to data interoperability. In this paper, we present how the industry can be strengthened using a peer-to-peer network based on the InterPlanetary File System (IPFS) framework to address the typical availability problems of web-based data. The paper particularly presents how Linked Data serialization of the IFC standard can be used in the context of IPFS and benchmarks the performance for the publication of models between IPFS versus HTTP protocol. We show that Linked Building Data - in particular, Industry Foundation Classes (IFC) models converted into Resource Description Format (RDF) graphs according to the ifcOWL ontology - can be implemented using the framework, with initial indications of significant benefits of a peer-to-peer network in terms of performance, offline access and immutable version history. Two use cases in distributed collaborative environments in the AEC/FM sector using evolving multi-domain models are used to evaluate the work.

Practical Applications In a construction project, stakeholders must collaborate and share design data models for efficient and safe completion. Building information published using Linked Data techniques aims to facilitate data

[8] https://ascelibrary.org/author-center/rights-permissions, Online Accessed 20th September, 2023

sharing between interested parties without a central intermediary. However, the challenges of Linked Data stem from using web protocols, resulting in limited availability, suboptimal performance, broken links [Haslhofer and Popitsch, 2009], and lost history. This study explores using the InterPlanetary File System (IPFS) a communication protocol for computers to communicate with each other directly without needing an intermediary. It addresses issues related to data availability on the Web and data transfer performance. The study demonstrates the suitability of IPFS in addressing the challenges of sharing building data. The data is available even when the original publisher host is down. In addition, the content-based addressing provides a digital fingerprint that can ensure authenticity and uniqueness, eliminating the data's duplication. Therefore, it minimizes the storage space consumed by data backups and archives. However, as IPFS addresses and communication principles are incompatible with the Linked Data standard approach, we designed a way so that the construction data saved in IPFS can be accessed transparently as a part of the standard Linked Data network.

7.2.1 Introduction

Currently, a few software vendors specialize in supplying the Architecture, Engineering and Construction (AEC) industry with software applications. The digitization of the construction industry has resulted in many advanced digital solutions for individual tasks or functions in scanning and measurement, architectural design, structural design, mechanical-electrical-plumbing (MEP) design, cost management, project planning, construction planning, etc. During the operational phase, building automation, asset and facility management, and access control systems generate streams of data about the use and condition of the building. However, the digitalization of the construction industry is chiefly focused on planning procedures that exhibit a lack of integration with practical day-to-day construction activities, particularly for small and medium-sized enterprises. To optimize the quality and timeliness of construction, data must be made usable along the whole value chain of construction [Brell-Cokcan et al., 2023].

The digitalization of the construction industry has been slower due to its fragmented value chain and data heterogeneity, leading to challenges in efficient data sharing between stakeholders [Oprach et al., 2019, Lehmann et al., 2023]. The development of digital solutions has primarily concentrated on individual tasks, leading to fragmentation and disconnection of data. This leads to latency, data re-entry, the need for human interpretation and manual interaction, and conversions between models and formats. Traditional file-sharing or centralized repositories only partially address these challenges, especially in the construction industry. As

smart cities, IoT, BIM, and GIS continue to evolve, online solutions for granular information linking will become necessary.

Linked Building Data [Pauwels et al., 2015, Pauwels, 2014b] is a decentralized solution to facilitate efficient data sharing among relevant parties in a standards-based manner without the need for a central intermediary. Building data is published on the Web using Linked Data technologies, transformed into a graph format based on RDF and each building object in the graph has a Uniform Resource Identifier (URI) as its unique global identifier. Structural descriptions are defined in web ontologies, such as ifcOWL [Pauwels and Terkaj, 2016]. LBD allows for retrieving object descriptions containing links to related objects and their context, enabling easy online access to the latest version of building objects, granular access to large models, linking to external data, and decentralized data management.

Linked Data is based on using standard web protocols – HTTP [Fielding et al., 1999, Belshe et al., 2015] in particular – whose advantage is their wide availability. The use of HTTP, however, creates a range of problems when the goal is to support machine-to-machine sharing of data:

1. *Limited availability*: Data will not be accessible if the server publishing the data is down or the user has connectivity problems. Also, other network issues may limit the availability of data.
2. *Suboptimal performance*: When the same content is retrieved repeatedly, the effort of previous accesses could be utilized to speed up successive accesses in improved caching or sharing of content between users close by in the network. In addition, when Transmission Control Protocol (TCP) [Forouzan, 2002] is used to convey HTTP (HTTP 1.1 [Fielding et al., 1999] and HTTP 2.0 [Belshe et al., 2015]), one dropped or lost packet means that the entire connection is halted until the missing packet is resent. [Ludin and Garza, 2017, Stenberg, 2019]
3. *Broken links*: A link is broken when that no longer leads to its intended destination. The reason for a URI change can be the renaming or disappearance of a domain or the reorganisation of content within a domain — all common occurrences on the Web. Following a broken link will result in an error instead of retrieving the identified object's description. [Rajabi et al., 2014]
4. *Lost history*: The description of an identified item can be overwritten without the ability to access the previous version.

The effects of these problems are more dramatic when machines access data rather than humans; the former are usually unable to recover from errors, while the latter has the flexibility to adapt to them.

This study uses the InterPlanetary File System (IPFS) [Benet, 2014], a peer-to-peer protocol that addresses issues related to Internet data availability. IPFS is a distributed file system that uses immutable URIs, based on cryptographic checksums of file contents [Protocol Labs, Inc., 2019], for addressing and linking data. It can be accessed via an IPFS-web gateway [Huynh et al., 2019].

IPFS is designed to solve the first and second problems above through its caching, pinning and peer-to-peer sharing mechanisms, and problems (3) and (4) through its immutable content-based URIs. However, it is not immediately clear how Linked Data serialization of the IFC standard (ifcOWL on the list of IFC formats in [buildingSMART International, Ltd., 2023]) can be used in the context of IPFS, and whether it fulfils its performance promises in this application. As a research method, we have followed the guidelines of Design Science by Hevner et al. [Hevner et al., 2004], implemented an artefact (the Prototype evaluation described in [Peffers et al., 2012]) to demonstrate the suitability of the construct to address the problems as mentioned above, and evaluated it in a controlled set-up (the Experimental Controlled Experiment evaluation method). Consequently, this study is constructed mainly on two research questions (RQs). The first RQ estimates the technical feasibility that is evaluated by creating prototypes. It focuses on the practice of publishing ifcOWL models. The second RQ focuses on the performance outcomes of the proposed practice with measurable results. The RQs of this study are as follows:

RQ1 What are the key steps, challenges, and technical requirements for publishing ifcOWL models as Linked Data using IPFS?
RQ2 What is the performance comparison of model publishing and object access regarding caching and local peer-to-peer access, IPFS versus HTTP protocol, and the benefits of sequential versus concurrent access?

The aim is to find a granular and efficient way to access the *building objects* contained in the models as linked data, which is helpful in the later stages of the construction lifecycle.

There are existing studies that utilize IPFS for different practices of construction lifecycle; such as project progress tracking and process monitoring [Adel et al., 2023], decentralized document management system [Koç, 2022], authorization and resilience of project documents with smart contracts [Dounas et al., 2020], etc. However, existing research studies do not consider the practicality of proposed solutions or methods regarding performance, open BIM standards such as Industry Foundation Classes (IFC) or Linked Data in data sharing. This paper fills the gap in the domain knowledge by providing a systematic analysis of the performance outcomes of different data access patterns in IPFS. This paper builds on previous research presented at the conference paper of [Oraskari et al., 2018a].

It demonstrates how ifcOWL models can be divided into stable sections, and through a novel test setup, the paper showcases the advantages of concurrent editing and evaluates scenarios where the original publisher's site is unavailable or the data is shared as a complete model over IPFS.

This paper consists of four sections. The following *Related Research* section presents the existing studies about the sharing BIM models, linked (building) data (LBD) and experiments with the IPFS network. Next, the *InterPlanetary File System (IPFS)* is introduced with its underlying data structure and cryptographic details. The section *Linked Building Data Using IPFS* answers the **RQ1** by detailing the publication process developed in the research, including the following steps:

- converting an IFC file into an RDF graph based on ifcOWL,
- segmenting the graph into (interlinked) single object subgraphs,
- canonically serialising the subgraphs, managing new versions of datasets in manner that maintains correct references,
- creating a dictionary that maps object identifiers to IPFS content identifiers,
- and publishing the files to the IPFS.

While this section clarifies the **RQ1** of this study, it also details the computational setup used to run the experiments presented in section *Use Case Tests and Results*. This section describes the use case and test scenarios and presents the collected test outcomes and analysis results. Performance is investigated in several scenarios using the Auckland IFC repository models [Amor, 2011] to provide early indications of the performance characteristics of the approach. It covers **RQ2**.

7.2.2 Related research

This chapter presents the context and related research on the granular exchange of building data based on open standards, such as IFC.

Practice of sharing BIM models

BIM models have traditionally been shared via point-to-point file exchange. While this is a decentralised and flexible approach, it has several well-known drawbacks: (1) It works with complete models. These are slow to open and work with. (2) It is applied ad hoc and disorganised, leading to obsolescence and confusion about who has what version of the model. (3) It does not support the linking of different

models.

The Information Container for Document Delivery (ICDD)[ISO, 2018a,b] is a multi-model container approach that allows using Linked Data. It exchanges multiple interrelated models or datasets in one package with specific link models that link identifiers across models. The main advantage is that it makes data sharing more organised, but it only provides a static snapshot of the links and does not support dynamic data enrichment. However, the main drawback is that it bundles models into a shared package, making the granularity of sharing even coarser than in regular file exchange. There is also an online web implementation for the ICDD containers [Hagedorn et al., 2023].

Centralised repositories like Autodesk BIM360, Trimble Connect, and BIMServer provide a user interface and an Application Programming Interface(API) to make project data accessible to stakeholders while having an access control system. These systems suit closed contexts like BIM data created during a construction project. However, data outside of a project will not be included, and linking and access to it must be managed separately. The centralisation of data and the choice of a particular system raise questions about the centre of the project, tool compatibility, data ownership, and digital sovereignty.

Linked Data

Linked Data [Bizer et al., 2009, Coyle, 2012] aims to achieve a decentralized and interlinked data space by publishing data on the Web according to the following principles:

1. Use URIs to name (identify) things.
2. Use HTTP URIs so that these things can be looked up (interpreted, "dereferenced").
3. Provide useful information about what a name identifies when it is looked up, using open standards such as RDF, SPARQL, etc.
4. Refer to other things using their HTTP URI-based names when publishing data on the Web.

These principles have been applied in the Linked Open Data Cloud [Bizer et al., 2008], where numerous RDF datasets have been published and interlinked.

There is a range of different types of fragments — called Linked Data Fragments [Verborgh et al., 2014] — in which RDF datasets can be shared on the Web:

1. *Data dump* — All triples of an RDF dataset in one unit.
2. *Subject page* — The (outgoing) triples of one subject in the RDF dataset, possibly including the (outgoing) triples of all referred anonymous nodes.
3. *Triple pattern fragment* — The triples of an RDF dataset that match a given triple pattern.
4. *SPARQL result* — The triples returned by a SPARQL endpoint.

A data dump is not a granular way of sharing data and is irrelevant to this study. On the other hand, SPARQL queries are different every time and do not guide how to partition the entire dataset. While SPARQL provides a very flexible way of accessing data, it also uses a lot of server capacity, potentially making a server a bottleneck in data consumption. The middle approaches - subject pages and triple pattern fragments - provide two orthogonal ways of partitioning the dataset: the former into objects and relations. However, only subject pages can partition an RDF dataset based on existing identifiers, such as IFC globally unique identifiers (GUID). This capability is essential for the approach adopted in this study.

Linked Building Data

Linked Building Data combines Linked Data and Building Information Modelling (BIM) to enable the seamless integration and exchange of structured data in the AEC domain. Research on Linked Building Data has mainly focused on developing ontologies and reasoning applications. The building data converted to RDF is mainly consumed through SPARQL queries and occasionally as data dumps. The other publication approaches have been less studied, except for the DRUMBEAT platform [Hoang and Törmä, 2017], which implements the subject page approach. The platform provides several services: conversion of IFC data to RDF, storage of the resulting graphs in an RDF database, and access to the data via a RESTful API [Rodriguez, 2008, Fielding, 2000, Richardson and Ruby, 2007, Pautasso, 2014]. Each building object in an IFC file that has a GUID will have a corresponding URI in the DRUMBEAT platform that can be used to access the object or link data to it. The data provided as a result of an access is a subject page of an object.

Niknam and Karshena [Niknam and Karshenas, 2015] present an example set of tools that enrich building data using semantic web technologies. The output is seen as a publishable result of the process, and BIM is just a static repository of information made available through a SPARQL endpoint. The endpoint is an enabling technology with all a web application's vulnerabilities. The vulnerability can be seen in the need to list the availability status on the World Wide Web Consortium (W3C) SparqlEndpoints page [World Wide Web Consortium comminity, 2017] and in the desideratum to create a system like SPARQLES

[Vandenbussche et al., 2017] that monitors the status of endpoints.

InterPlanetary File System (IPFS)

IPFS [Benet, 2014] is a peer-to-peer decentralised storage network using routing and addressing techniques. In IPFS, a *node* is a computer that participates in a distributed hash table (DHT) [Dymcek, 2011] network. A DHT is a hash table implemented using a network of nodes [Balakrishnan et al., 2003] to locate data based on keys. Each node has a routing table to forward hash queries to the node holding the matching key. IPFS uses Merkle trees, a concept introduced by Ralph Merkle in [Merkle, 1979] and widely used in distributed systems such as Git, blockchain, etc. In a Merkle tree, each leaf node has a hash of its data content, and non-leaf nodes have a hash of their children's labels. IPFS expands on Merkle trees by introducing Merkle Directed Acyclic Graphs (DAGs)[Dias et al., 2023], which enable non-leaf nodes to store data and utilise InterPlanetary Content Identifiers (CIDs), cryptographic checksums of the data, to identify and locate the content uniquely.

IPFS uses Coral [Freedman and Mazieres, 2003] and Kademlia [Maymounkov and Mazieres, 2002] as distributed address books to reference data hosting machines, using eXclusive OR (XOR) distances for routing. Proper use of DHT enables IPFS to handle high traffic, share network load and minimise latency. Unlike push-based peer-to-peer sharing, IPFS is pull-based, sharing data only on demand. This limits the number of copies of data on the network and the potential benefits of peer-to-peer file sharing.

IPFS also uses garbage collection to reduce node storage requirements. Users can permanently mark copies of files to be cached, incentivising them to cache valuable data for immediate access. IPFS uses Merkle trees to identify and verify data content, with each node tagged with an InterPlanetary Content Identifier (CID) that can be used to address the content. Unlike blockchain-based systems, IPFS data is open by default, and security must be added as required [Aleksandersen, 2019]. Editing content is complex because the data is scattered across numerous data repositories.

In [Sicilia et al., 2016], Sicilia describes a method to convert a directed acyclic graph (DAG) to IPFS, starting from leaves and working upwards, and using an index table structure for non-DAG cases. Unless there are frequent changes, data snapshots are published as IPFS files. Only updated parts of the collection are published when a change is made. Index records of IPFS URIs are used to version the collections. However, there is no version management, and RDF is not used

consistently for all links.

7.2.3 Linked Building Data Using IPFS

When leveraging IPFS for implementing Linked Data, the adherence to Linked Data principles [Bizer et al., 2009, Coyle, 2012] must be re-evaluated, given that IPFS uses content-based checksums as addressed. Consequently, addressing resources cannot rely on HTTP URIs, requiring adopting a new identifier scheme. Therefore, this work's main contribution is showing how Linked Building Data is possible in this context.

Publishing Industry Foundation Classes (IFC) data into IPFS network

Autodesk collaborated on the first version of the Industry Foundation Classes (IFC) specification, published in 1997. The collaboration later became a not-for-profit organisation to promote the new neutral specification for data modelling in Architecture, Engineering and Construction (AEC) Industry [Nizam and Zhang, 2017]. IFC version 4.0 was released on 12 March 2013 and adopted as an ISO standard (ISO 16739:2013)[ISO, 2013]. IFC has become essential for data transfer between AEC software applications and is mandated by many government agencies, and is currently widely implemented in existing AEC tools.

This work assumes BIM models exported as IFC can be segmented into parts to isolate changes from unmodified portions. We also assume that IFC Globally unique identifiers (IFC GUIDs) are stable identifiers representing logical units in the model graph, the building elements. New versions of the building model can be exported from any design tool capable of exporting data in the IFC STEP File (IFC-SPF, defined by ISO 10303-21) format and are expected to be controlled by the external tool. The core idea of IPFS is that data with the same content can be fetched using its cryptographic checksum, resulting in the criterion that isomorphic graphs have the exact representation.

The IFCtoRDF [Pauwels et al., 2019] conversion tool generates an attribute value (Abox) [Bergman, 2009] graph output. A line-number-based URI [Berners-Lee et al., 1994] is created for each object with a line-number identifier [Nour et al., 2006] in the IFC STEP physical file [Pratt, 2001, for Standardization", 2002]. Because they are based on object line identifiers, they are not persistent.

According to [Nour et al., 2006], BIM applications can have varying line

numbering, which may not be consistent after exporting to IFC. This leads to the situation in Figure 7.8, where identical data has different Message-Digest Algorithm 5 (MD5) hashes due to different serialization. To increase the likelihood of data availability on the IPFS network, unifying the data content can be useful and made in the following way: To express that there is no persistent reference identifier for an unmodified IFC file line, we propose to make them nameless by default to unify the same, unchanged data in consecutive model exports but also in different models containing identical data. The nodes with an IFC GUID identifier are given URIs [Berners-Lee et al., 1994] and are thus made dereferenceable [Harth et al., 2010, Heath and Bizer, 2011],i.e., can be looked up on the Web. Therefore nodes with an IFC GUID as an attribute are converted into an RDF resource with the IFC GUID as part of its URI [Berners-Lee et al., 1994]. All the other nodes that are not literals are recreated as nameless. Then, the principle of selecting connected triples was influenced by the definition of minimum self-contained graphs presented by Tummarello et al. in [Tummarello et al., 2005]. In our work, triples directly associated with an IFC GUID literal are selected, and recursively, for all empty nodes in the object position of a triple in the existing triple set, RDF triples where they are in the subject position are included. Finally, each resulting triple set is converted to the standard form presented in [Carroll, 2003]. Jeremy Carroll's RDF C14N canonisation algorithm sufficiently guarantees the same serialisation of small triple sets when they represent the same content.

After the unifying process, the RDF model can be rewritten so that the blank nodes can be given deterministic URIs that are based on the element GUID concatenated with the position in the ordered list of triples in the Carroll RDF C14N canonized [Carroll, 2003] form of the entity data.

The design has characteristics in line with the Linked Data principles [Curry et al., 2013, Khusro et al., 2014]. Turtle's standard RDF serialization makes bringing the data fragments into one RDF store possible. The limitation is that blank nodes that point to the same piece in the BIM model but are associated with a different version of the data may not be merged, which is acceptable to avoid confusion. Conversely, if several model versions share an element and the shared content is identical, they are merged. So, reading IPFS nodes from different IFC models, such as architectural and structural models, into the same RDF graph as a merged model allows complex federated queries to be written across model boundaries.

IFC Model Segmentation

To publish BIM data as IPFS, splitting the model into parts is preferable so that when a new version of the model is published, only the changed part of the data

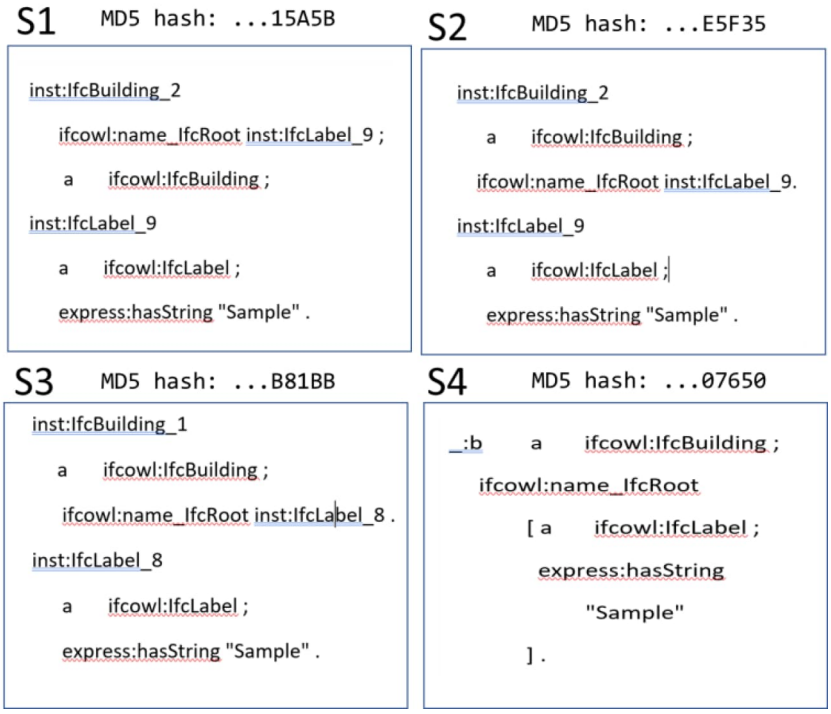


Fig. 7.8. The four RDF triple snippets have the same meaning but a different hash.

is published, and unmodified data can be shared between versions. We assume that IFC elements are the appropriate granularity for splitting. In this work, we expect any IFC model to be represented as a collection of separate directed acyclic RDF graphs. Let us presume that nodes that have any global identifier (**G**), in the ifcOWL context, nodes that are an individual instance of a subclass of ifcowl:ifcRoot[buildingSMART International, Ltd., 2023], and nodes (**N**) that have no incoming arcs, i.e., the nodes are not in the object position in a triple, can be used as root nodes (**R**) [Monma and Wei, 1986] for the rooted subgraphs.

Definition 7.2.1 (Root Node Set) $R \coloneqq G \cup N$

For example, if we have a simple IFC-SPF formatted file shown on the left side in Figure 7.9 and convert it into IfcOWL RDF Abox triples shown on the right, the node that is circled with a continuous line has an IFC GUID literal attached. The other node with no incoming arcs is marked with a dashed line. For clarity, the IfcOwnerHistory entity and contained elements have been removed from the RDF triple set.

```
#1=IFCBUILDING('WJ7yGd42kMW08n7Cp432Kr',#2,'Sample',$,$,$,$,$,$,$,$);

#2=IFCOWNERHISTORY(#3,#6,$,.ADDED.,$,$,$,87763554);

#3=IFCPERSONANDORGANIZATION(#4,#5,());

#4=IFCPERSON($,$,$,(),(),(),(),());

#5=IFCORGANIZATION($,'ArchInc.',$,(),());

#6=IFCAPPLICATION(#7,'V 1.0','Toolkit','BAT1.0');

#7=IFCORGANIZATION($,'Inc.',$,(),());
```

```
@prefix rdf:      <https://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix owl:      <https://www.w3.org/2002/07/owl#> .
@prefix ifcowl:   <https://standards.buildingsmart.org/IFC/DEV/IFC2x3/FINAL/OWL > .
@prefix inst:     <https://ipfs/bim/> .
@prefix express:  <https://w3id.org/express#> .

inst:IfcBuilding_1
    a       ifcowl:IfcBuilding ;
    ifcowl:globalId_IfcRoot
          inst:IfcGloballyUniqueId_7 ;
    ifcowl:name_IfcRoot inst:IfcLabel_8 .
inst:IfcGloballyUniqueId_7
    a       ifcowl:IfcGloballyUniqueId ;
    express:hasString "WJ7yGd42kMW08n7Cp432Kr" .
inst:IfcLabel_8
    a       ifcowl:IfcLabel ;
    express:hasString "Sample" .

inst:
    a       owl:Ontology ;
    owl:imports ifcowl: .
```

Fig. 7.9. On the left is a minimalistic IFC model and on the right the equivalent ifcOWL graph.

Definition 7.2.2 defines simple rules to compose a subgraph when a root node is given. The composed, directed acyclic subgraphs for the example are shown in Figure 7.10. Figure 7.11 shows the two subgraphs' equivalent Turtle - Terse RDF Triple listings.

Definition 7.2.2 (Concise Bounded Description $CBD(\mathbf{r})$) *, where* ***r*** *in* ***R*** *is a root for a subgraph)*

1. If not present, the root node is given a URI. Others are made nameless. 2. Triples that are directly connected to the root node are selected. 3. Recursively, for all contained nodes that are not in the Root Node Set (Definition 7.2.1 or Definition 7.2.3) and, are in the object position of a triple, triples, where they are in the subject position, are included.

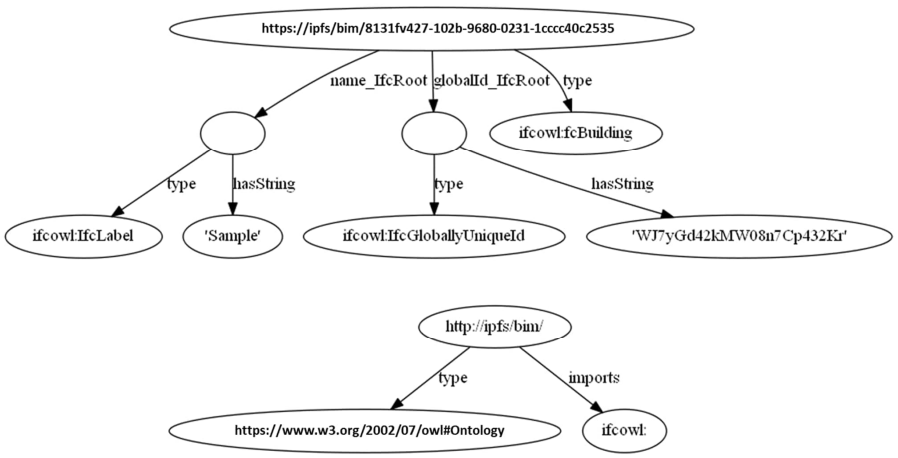


Fig. 7.10. The composed directed acyclic subgraphs.

Subgraph 1:

```
@prefix express:  <https://w3id.org/express#> .
@prefix ifcowl: <https://standards.buildingsmart.org/IFC/DEV/IFC2x3/FINAL/OWL > .
@prefix rdf:    <https://www.w3.org/1999/02/22-rdf-syntax-ns#> .

<http://ipfs/bim/WJ7yGd42kMW08n7Cp432Kr>
    a      ifcowl:IfcBuilding ;
    ifcowl:globalId_IfcRoot
        [ a      ifcowl:IfcGloballyUniqueId ;
          express:hasString
              "WJ7yGd42kMW08n7Cp432Kr"
        ] ;
    ifcowl:name_IfcRoot
        [ a      ifcowl:IfcLabel ;
          express:hasString
              "Sample"
        ] .
```

Subgraph 2:

```
@prefix rdf:    <https://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix owl:    <https://www.w3.org/2002/07/owl#> .
@prefix ifcowl: <https://standards.buildingsmart.org/IFC/DEV/IFC2x3/FINAL/OWL > .
.

[]  a  owl:Ontology ;
    owl:imports
        ifcowl:  .
```

Fig. 7.11. The Turtle - Terse RDF Triple representation of the subgraphs.

When segmentation (Definition 7.2.2) is calculated using Definition 7.2.1 root set, structures that are shared in the IFC data model are copied many times. To avoid this, the root nodes set of the rooted graphs [Monma and Wei, 1986] is extended to contain the first found occurrences of a shared node (S) in the traverse in Definition 7.2.2. The extended root set is defined as Definition 7.2.3.

Definition 7.2.3 (Extended Root Node Set) $\boldsymbol{R} \coloneqq \boldsymbol{G} \cup \boldsymbol{N} \cup \boldsymbol{S}$

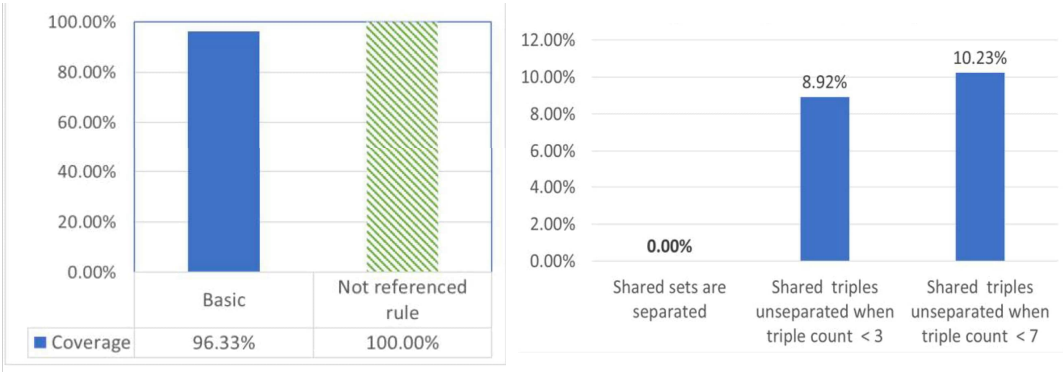
To estimate the completeness of the conversion, the union of composed triples

(Definition 7.2.4) for test models was formed, and the intersection of the partitions was calculated. The IFC model collection in Open IFC Model Repository [Amor, 2011] was used. The result is displayed in Figure 7.12a. In short, when the IfcRoot nodes and unreferenced nodes were used to initiate the conversion (Definition 7.2.1), the result of the conversions covered all triples in every model. No triple in the RDF graph was excluded. Only models with errors (e.g., referring to unexisting line numbers) or were using an IFC version not supported by the IFCtoRDF converter [Pauwels et al., 2019] were excluded. The complete source code for the test and the full results are in [Oraskari, 2019b].

Definition 7.2.4 (The union triple set) $UnionGraph \coloneqq \cup CBD(\boldsymbol{r})$, *for all* $\boldsymbol{r}$ *in* $\boldsymbol{R}$ *in the graph.*

Since the coverage was 100 per cent (Figure 7.12a), the difference between the total count of all triples and the triple count in the ifcOWL model was used to estimate the count of the shared triples in the segments. It was zero in the basic setup using the Definition 7.2.3 root set. Figure 7.12b summarises the RDF segmentation overlap percentages of Open IFC Model Repository in [Amor and Dimyadi, 2010]. Briefly, segmentation of no overlap can be created. However, many shared triples contain measurement values that may logically belong to the hosting entity. The figure's graph illustrates how the partitions' average overlap percentage is affected if the smallest shared triple sets are kept unseparated. The results are based on the repository's 102 error-free and unfiltered IFC2X3 and IFC4 models. The partition overlap in an unfiltered ifcOWL graph is zero per cent when the shared parts of the graph are separated in different IPFS files. A decent amount of overlap is kept when the smallest shared triple sets are kept unseparated.

In addition to the completeness of the conversion, the reachability from an RDF node or a directory is an important aspect when considering the IFC model integrity. Accessing the whole model starting from one node in the RDF graph is preferable when following Linked Data links. A directory can serve this purpose. It is also at least partially true for an arbitrary IfcRoot element when using an ontology reasoner to infer inverse relation properties defined in ifcOWL. In the tests, it was shown that Apache Jena OWLMicroReasoner is capable of adding these for a sample IfcWindows element specified in IFC2X3.



(a) The average model coverage of the partitions **(b)** The average overlap percentage of the partitions

Fig. 7.12. The average partition coverage and the overlap percentage in the partitioned models based on the converted 102 error-free and unfiltered IFC2X3 and IFC4 models in Open IFC Model Repository

To make isomorphic RDF data serialized the same, RDF Normalization described in [Longley, 2023, Kellogg et al., 2023] is used. It is based on Jeremy Carroll's work [Carroll, 2003]. In the test, each subgraph is converted to the standard canonized form presented in [Carroll, 2003]. As it is designed for checksum calculation of the RDF graphs, it formats the RDF data content in the form where the serialization should be the same when the triple set expresses the same information.

Carroll's canonization can be calculated in $O(n \log n)$. Carroll's algorithm sorts triples, ignoring blank node labels. When non-deterministic changes to the graph can be avoided, the sorting is repeated three times. In the process, blank nodes are given new names based on their occurrence position in the sorted file. In the first round, identical consequent entries are skipped. In the second round, still unnamed entries are given names, if possible (non-deterministic changes can be added to be able to name the rest of the nameless nodes). The last round finalizes the serialization.

If preferred, the blank node generation in the algorithm can be replaced by a function that outputs a deterministic set of URIs. One way to achieve this is to calculate a checksum of the canonized representation of the triple set and use that to create a base for the URI set. The corresponding graph serialization for the subgraph is shown in figure 7.13.

Subgraph 1:

```
@prefix inst:    <https://ipfs/bim/> .
@prefix express: <https://w3id.org/express#> .
@prefix ifcowl: <https://standards.buildingsmart.org/IFC/DEV/IFC2x3/FINAL/OWL > .
@prefix rdf:     <https://www.w3.org/1999/02/22-rdf-syntax-ns#> .

_:g000001 rdf:type ifcowl:IfcGloballyUniqueId ;
     express:hasString "WJ7yGd42kMW08n7Cp432Kr" .
_:g000002 rdf:type ifcowl:IfcLabel ;
    express:hasString "Sample" .
inst:WJ7yGd42kMW08n7Cp432Kr ;
    ifcowl:globalId_IfcRoot _:g000001 ;
 ifcowl:name_IfcRoot _:g000002 ;
 rdf:type ifcowl:IfcBuilding .
```

Subgraph 2:

```
@prefix express: <https://w3id.org/express#> .
@prefix rdf:     <https://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix owl:     <https://www.w3.org/2002/07/owl#> .
@prefix ifcowl:  <https://standards.buildingsmart.org/IFC/DEV/IFC2x3/FINAL/OWL > .

_:g000001 rdf:type owl:Ontology ;
  owl:imports ifcowl: .
```

Fig. 7.13. The standard form Turtle representation of the subgraphs.

Linked Data access to the models

Figure 7.14 shows a linked data implementation using IPFS, where each table in the graph represents a separate IPFS file. The inter-element links point to the IPFS content identifiers of the addressed elements via an IPFS gateway. Ontology description files can also be located in IPFS. One drawback is that model versions do not share any content, and even unmodified entries are multiplied.

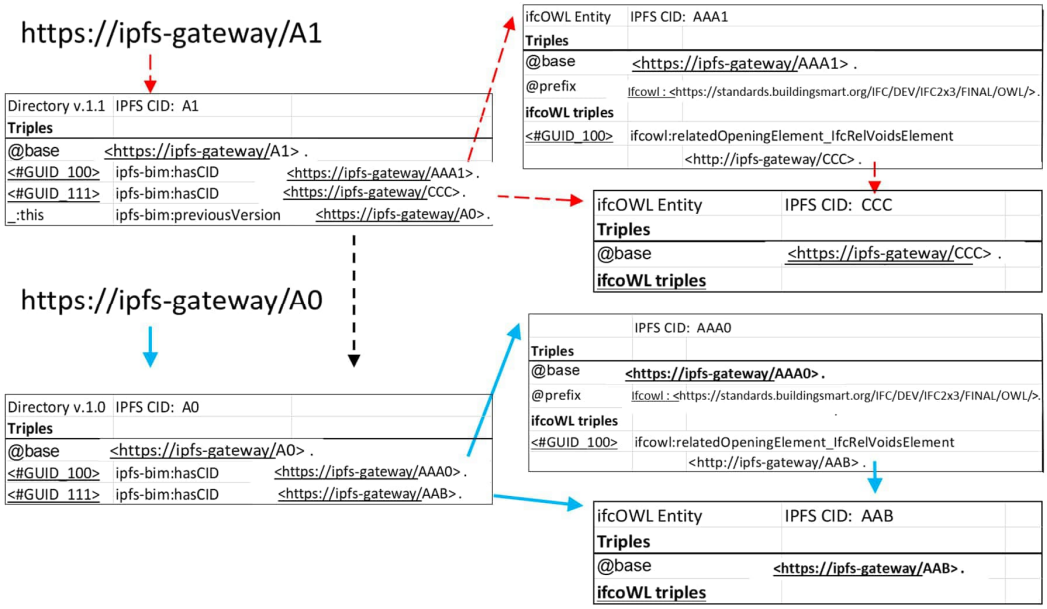


Fig. 7.14. Linked Data access to the model.

To ensure that links remain consistent even when entities change slightly, it is necessary to avoid using content identifiers as the basis for links. Therefore, the addresses are based on IFC GUIDs, and to associate each IFC GUID with the appropriate IPFS content identifier, a separate version directory is utilized. This allows multiple model versions to share content they have in common. Figure 7.15 illustrates how two model versions can share an element using context-dependent links presented in black.

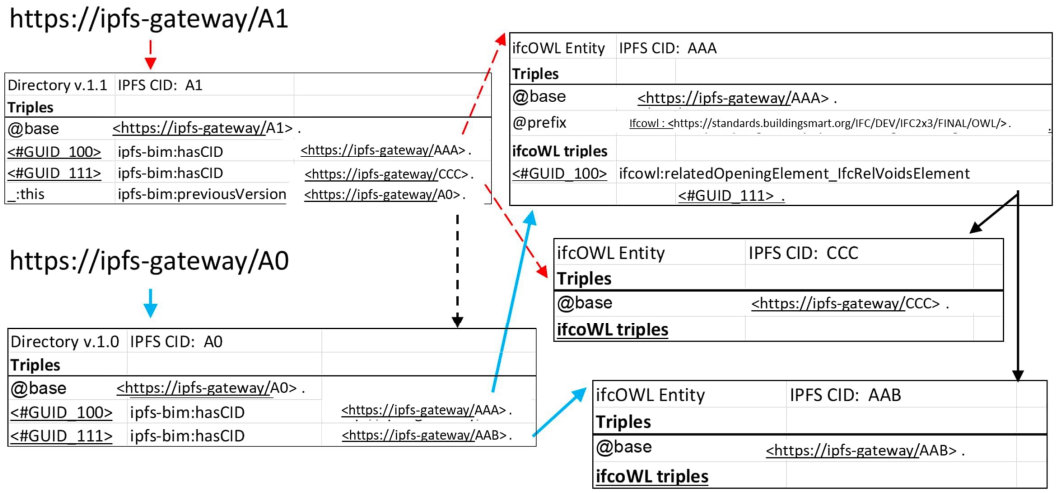


Fig. 7.15. Linked Data access using indirect links to share data.

7.2.4 Use Case Tests and Results

Table 7.8. The models used in the IPFS publication test.

Set	Name	ifcOWL Triples	Filtered Triple Count	IFC Entities
NIST:ec-001	Eccentricity physical	375	276	9
IAI: 10-001	Roof with openings ben 1	1200	238	18
IAI: 13p-001	Plate steel exam tek 1	3210	434	34
IAI: 9-0-001	Railing extrusion tek 1	5262	277	17
IAI: 1-0-001	Wall opening straight ac 1	16877	1694	97
NIST:AI-001	AISC Sculpture param	34331	34030	1062
NIST: Be-OO1	Bentley1 param	48514	53215	1269
NIST: De-OO1	DesignData1 brep	57733	42676	1128
KIT: PAR-001	PART03 Buderus	103440	1407	31
Four sto-OO1	Four storey office building	228303	221916	1465
NIST:an-001	Analysis param	291613	267159	13500
KIT: AC1-001	AC11 Institute Var 2	347443	283720	6565
KIT: PAR-001	PART06 Kermi	382771	751	13
KIT: AC–001	AC 11 Smiley West	446461	247264	4823
KIT: AC1-001	AC11 FZK Haus	603847	32824	873
KIT: All-001	Allplan 2008 Institute Var 2	745512	91244	5422
West Riv-001 2x3	West Riverside Hospital Parking Garage	750664	655639	8988
KIT: FZK-001	FZK Haus EliteCAD	961770	63373	457
KIT: Nem-001	Nem FZK Haus 2x3	1396465	4514	289

Source: Data from Oraskari et al. [Oraskari et al., 2018a].

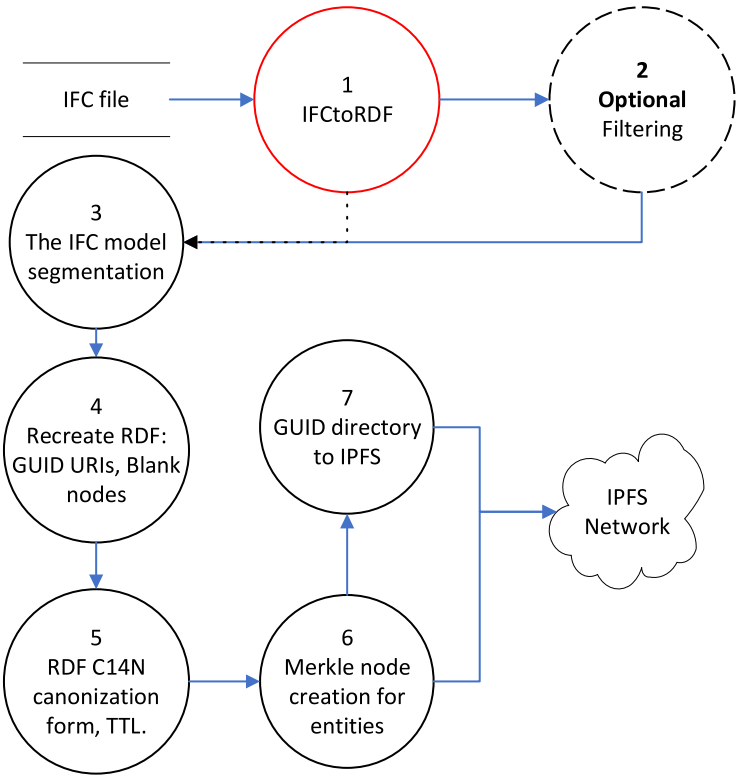


Fig. 7.16. The publication process for a new IFC file. Modified from the original work in [Oraskari et al., 2018a].

A selected set of IFC models of *Open IFC Model Repository* [Amor, 2011] was used in the tests, as listed in Table 7.8. The publication process is depicted in Figure 7.16.

The test use case focuses on a scenario of a significant infrastructure project, such as a bridge or motorway. In such a project, multiple stakeholders, including architects, engineers, contractors and government agencies, would need to collaborate and share design data models to ensure the project is completed efficiently and safely. A designer needs an up-to-date copy of a building's architectural design to continue working. Three colleagues have previously viewed the design; an architect in London has published the original version.

The research questions are:

R1 How much does the user benefit from local copies while using IPFS?
R2 What is the performance gain compared to Linked Data over the HTTP access?

R3 How should the files be downloaded? Sequential or concurrent?

The test setup included four local Hewlett-Packard(HP) workstations in Finland and a remote virtual machine in London, as shown in Figure 7.17.

The HP z240 desktop workstations were used for the purpose. They have Intel Xeon E3-1230 v6 @ 3.50 GHz processors, 32GB RAM, and a 64-bit Microsoft Windows 10 operating system. All the computers are on the same local network and connected to the same cross-connection cabinet. PC1, PC3, and PC4 have a download speed of around 800 megabits per second (Mbps) and an upload speed of 940 Mbps. The values for PC2 were 88 Mbps for download and 94 Mbps for upload. In the network, the upload speed was higher. The virtual host has 1 CPU, 2 GB of main memory, and Ubuntu 18.04. In the test, PC1 was always used to run the measurement. All the test software is open source and available on [Oraskari, 2019a].

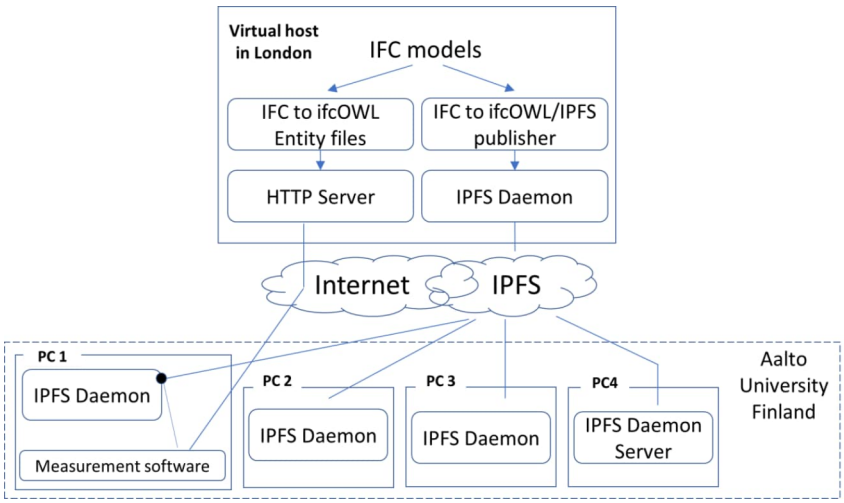


Fig. 7.17. Test setup to measure the download times of the complete models.

The tests were made to analyze the IPFS download data throughput characteristics in various cases. The cases for the sequential tests were:

1. The test model is copied element-wise sequentially from only one remote host. The remote host is the virtual host.
2. The test model is copied element-wise sequentially from one remote host and a local host (PC2)
3. The test model is copied element-wise sequentially from one remote host and two local hosts (PC2 and PC3)
4. The test model is copied element-wise sequentially from one remote host

and three local hosts (PC2, PC3, and PC4)
5. The test model is copied element-wise sequentially from the computer's cache (PC1).

The test runs repeated each download sequence, i.e., the test case, one hundred times. The IfcOpenHouse example model was used in the test. A random number was included in the IPFS files to guarantee that earlier runs did not affect the results. Figure 7.18 shows the total download times for the test cases.

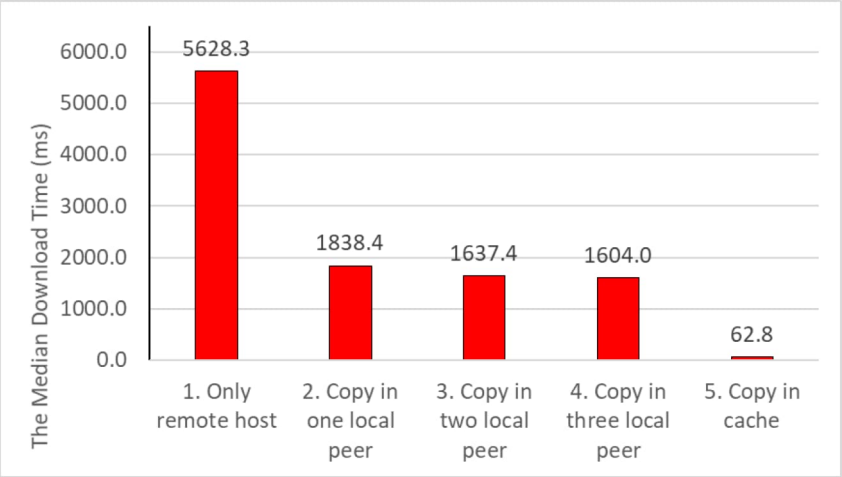


Fig. 7.18. The download time of the sequential element-wise of the whole test model. Results are based on 100 runs.

For the **R1** *"How much does the user benefit from local copies while using IPFS?"*, the results show that the median data copy time from IPFS takes around one-third of the time when there is a local copy, and accessing the same data from the local cache is a hundred times faster than from a single remote node. The performance results were compared with a web-based Linked Building Data implementation using Busybox as an HTTP server. The element-wise download sequence was repeated using HTTP, and the results were summarized in Table 7.9.

For the **R2** *"What is the performance gain compared to Linked Data over the HTTP access?"*, the results show that the HTTP access times are longer than IPFS, and IPFS has a remarkable standard deviation, with occasional breaks in service. Figure 7.19 shows these breaks in the test model's one hundred consecutive sequential IPFS downloads. Two reasons for the outliers may be the network traffic causing IPFS connection manager software to disconnect the peer and IPFS BitSwap protocol's default fairness strategy blocking access until the counter is reset on time. In a typical real-world scenario, delays are not as likely and frequent

Table 7.9. The element level download times of the IfcOpenHouse test model. The results are based on 7600 entity downloads per a case.

Measure	Local filesystem	HTTP server	IPFS (one data holder)	IPFS cache
Average read time (ms)	1.15	130.64	74.06	0.83
Median read time (ms)	0.89	157.87	56.4	0.50
Standard deviation (ms)	3.94	61.31	702.54	2.41

as in the setup. Therefore, the median of the measurements may be a more representative indicator of the system's actual performance.

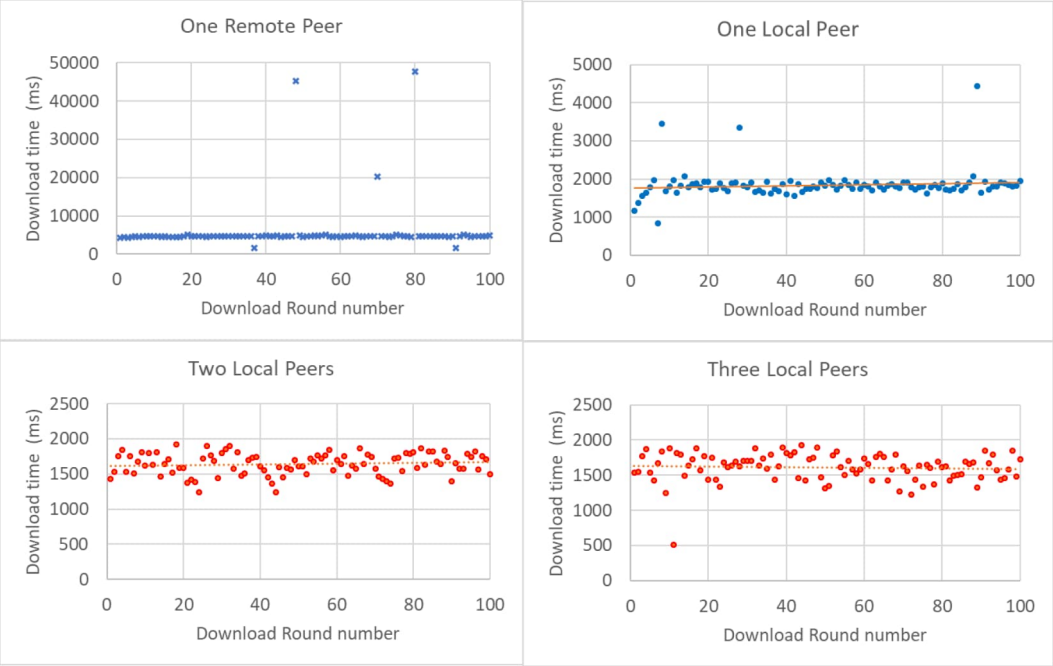
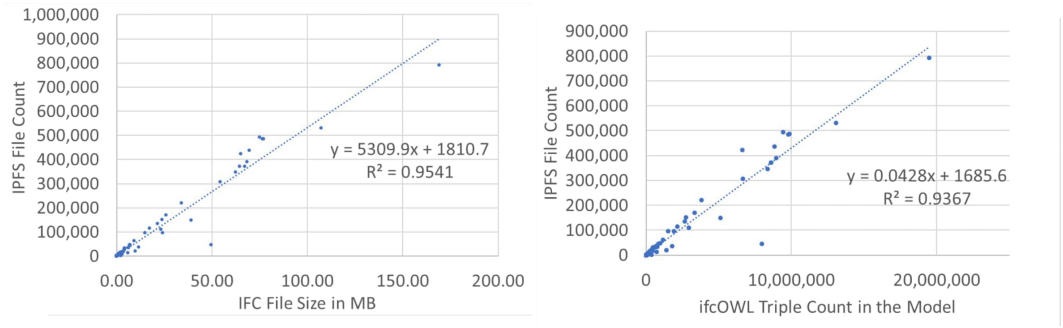


Fig. 7.19. The outliers can be seen in the plot of the consecutive download runs.

To test the hypothesis that the element in the small test model is a representative collection of ifcOWL elements in any model, the test set listed in Table 7.8 was analyzed. It was found that when the geometry is filtered out, all the entities fit in the 256 kB block size, so IPFS handles them as one data unit each. However, when geometry was kept in the data, some ifcOWL entities were larger than the IPFS block size and were automatically divided by IPFS, which affected performance. The main implication of the result is that geometry-filtered partitioned ifcOWL

models have similar performance characteristics regardless of size, so linear regression equations can be used to estimate IPFS publishing and download times of models of any size. Figure 7.20 shows the correlation between the model size and the IPFS file count.



(a) File size versus IPFS file count **(b)** ifcOWL triple count versus IPFS file count

Fig. 7.20. The size of the geometry filtered model versus the created IPFS File count. The results are based on the 102 error-free and unfiltered IFC2X3 and IFC4 models in Open IFC Model Repository in Auckland.

This study also considers the potential limitations of the IPFS network. Therefore, the RQ(3) *"How should the files be downloaded? Sequential or concurrent?"* was taken into consideration into the tests. The cases for the asynchronous tests were:

1. The test model is copied element-wise concurrently from only one remote host. The remote host is the virtual host.
2. The test model is copied element-wise concurrently from one remote host and a local host (PC2)
3. The test model is copied element-wise concurrently from one remote host and two local hosts (PC2 and PC3)
4. The test model is copied element-wise concurrently from one remote host and three local hosts (PC2, PC3, and PC4)
5. The test model is copied element-wise concurrently from the computer's cache (PC1).

The asynchronous tests were otherwise like the sequential element-wise access. However, the test program used Java ExecutorService tasks to create concurrent *IPFS get* requests for all the elements in the test model. The concurrency was only at the IPFS Java API part. The IPFS daemon at the node (PC1) was a black box for the implementation. The correctness of the concurrent operation was validated, fetching the data twice in each task and testing that the read bytes were the same. Figure 7.21 shows the result. The main result is that, for a small model, concurrent access improves performance considerably. Since the operation is expected to

group the concurrent reads into a longer wanted nodes list in the IPFS BitSwap protocol, it may stand a good amount of concurrency. However, further studies are needed to analyze the resource limitations.

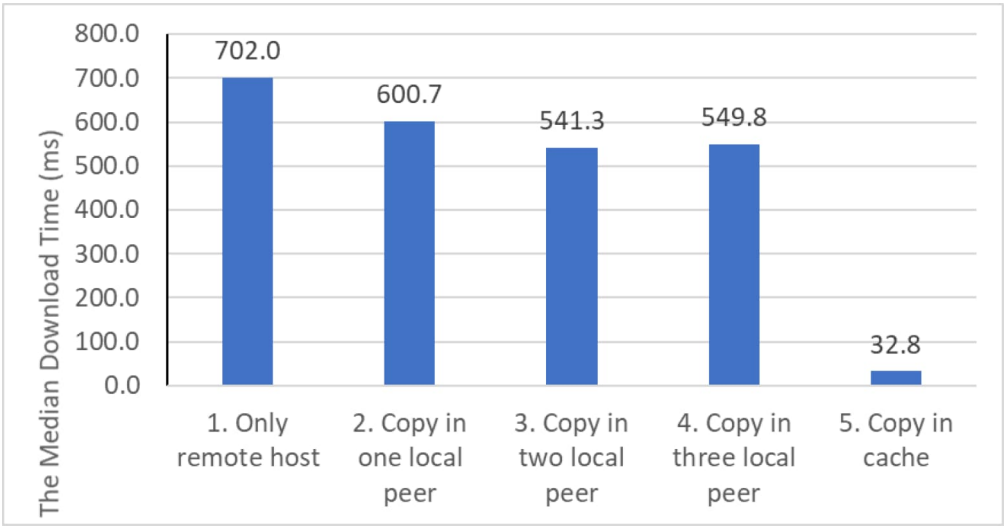
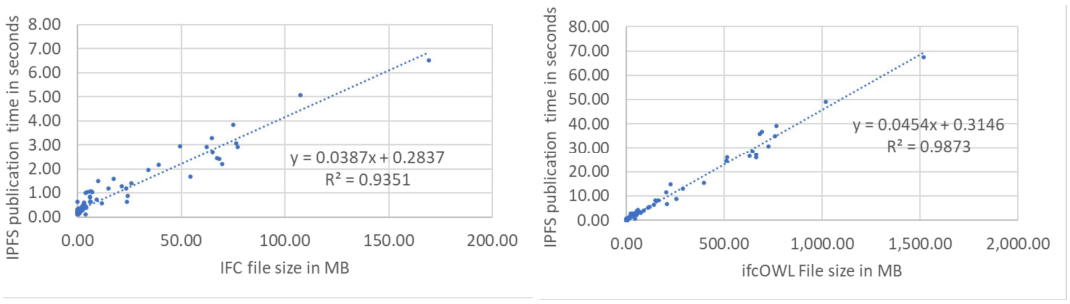


Fig. 7.21. The download time of the concurrent element-wise of the whole test model. The results are based on one hundred runs.

The IFC models were published and downloaded as raw files to study the performance impact of larger file sizes. Figures 7.22 and 7.23 show the results. IPFS performed better than HTTP when file sizes were small, and there were local copies, but HTTP outperformed IPFS when copying large files from a single host. This aligns with the Peer-to-peer (P2P) assumption that sharing data content or resources gives an advantage to P2P parties.



(a) IFC publication time for small IFC files

(b) IFC publication time for large IFC files

Fig. 7.22. The file publication times measures using the IPFS add command. The results are based on the 102 error-free and unfiltered IFC2X3 and IFC4 models in Open IFC Model Repository in Auckland.

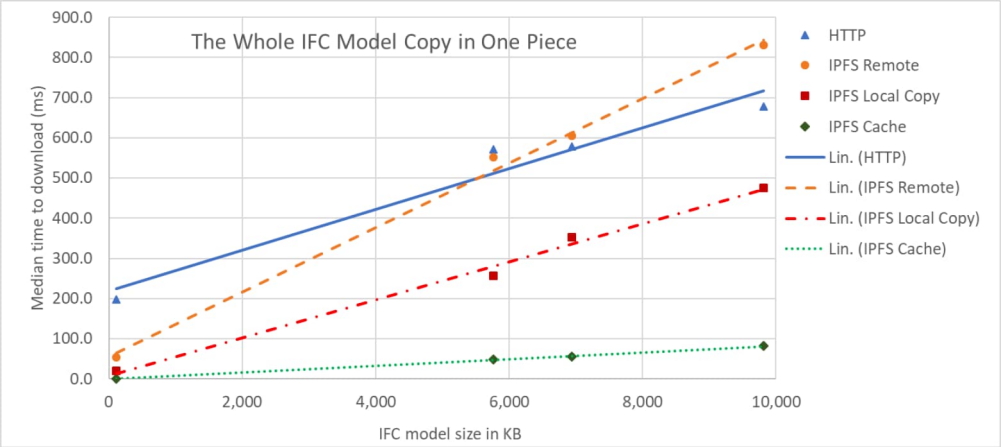


Fig. 7.23. The performance comparison of the download times of the larger IFC files. The results are based on one hundred runs.

7.2.5 Discussion and Conclusions

The article discusses the potential benefits of implementing ifcOWL Linked Building Data using the IPFS framework, including improved performance, offline access, and data availability. The first part addresses **RQ1**, the feasibility of ifcOWL-based Linked Building Data in IPFS. The main challenge mentioned in the article is the different data-addressing principles. It is shown how the ifcOWL graph can be segmented into (interlinked) single-object subgraphs and how to serialize the data in a form suitable for content-based addressing. Combined with the earlier results, this completes the method to publish ifcOWL data in IPFS as Linked Data. The implementation conforms to Linked Data principles, using URIs to name things and directories to make the data version independent.

The second part addresses the performance of the suggested data model (**RQ2**). The article reaches this by reviewing different data access patterns with the following summary of RQs reviewed in tests:

1. **R1** *Data access from IPFS via a local copy vs. local cache*: The median data copy time is around one-third of the time when there is a local copy, and accessing the same data from the local cache is a hundred times faster than from a single remote node. So a local cache is suggested.
2. **R2** *Data access via Linked Data vs. HTTP*: HTTP access times are longer than IPFS, and IPFS has a remarkable standard deviation when the size of each block is small enough so IPFS can handle it as a single item. Therefore, If the block sizes are bigger, the article suggests utilizing an HTTP network.

3. **R3** *Data access with sequential vs. concurrent downloads*: Although this question needs further evaluation in future research studies, this suggests concurrent downloads for performance improvement based on the results.

The data is available even when the original publisher host is down, and IPFS allows automatic caching and replication. However, the experiments were conducted with a limited number of computers, raising the question of how performance would scale with more devices in the network.

Future RQs related to IPFS include investigating the network's performance with a larger number of nodes and identifying factors influencing professionals' adoption of this technology. We suggest studying emerging ontologies for their suitability to be published in the IPFS network and enhancing the distributed data storage and publishing mechanism with advanced cryptographic hash functions.

Further research is suggested to implement a fully distributed role-based access control for data published as copies in many unknown Internet parties, including in the IPFS.

7.2.6 Data Availability Statement

All data and code generated during the study are available in our GitHub repository [Oraskari, 2023] online in accordance with funder data retention policies. All data models that were used in the study were provided by a third party. Direct requests for these materials may be made to the provider as indicated in the Acknowledgements.

7.2.7 Acknowledgements

This research has been carried out at Chair of Design Computation, RWTH Aachen, Germany, RWTH Aachen partly through BIM4Ren EU project (Grant No. 820773), and funded by EIT Digital Doctoral School, Aalto-yliopiston tekniikan tukisäätiö, RWTH Aachen, and VisuaLynk Oy.

Data models of the Open IFC Model Repository of the Faculty of Science, School of Computer Science, University of Auckland, New Zealand were used in the work.

References

Kareem Adel, Ahmed Elhakeem, and Mohamed Marzouk. Decentralized system for construction projects data management using blockchain and ipfs. *Journal of Civil Engineering and Management*, 29(4):342–359, 2023.

Daniel Aleksandersen. A look at ipfs' garbage collector and how long resources remains cached, Jan 2019. URL https://www.ctrl.blog/entry/ipfs-repo-cache-gc.html. Accessed on Sep. 05. 2023.

Robert Amor. Analysis of reference ifc models. In *Proceedings of the 2011 CIB W78 Conference, Sophia Antipolis, France*, pages 26–28. The ITC Digital Library, 2011. URL https://itc.scix.net/pdfs/w78-2011-Paper-50.pdf. Accessed on Sep. 05. 2023.

Robert Amor and Johannes Dimyadi. An open repository of ifc data models and analyses to support interoperability deployment, 2010. URL http://dx.doi.org/10.13140/2.1.1905.7609. Accessed Oct. 05. 2023.

Hari Balakrishnan, M Frans Kaashoek, David Karger, David Karger, Robert Morris, and Ion Stoica. Looking up data in p2p systems. *Communications of the Association for Computing Machinery*, 46(2):43–48, 2003. Association for Computing Machinery.

M. Belshe, R. Peon, and M. Thomson. Hypertext transfer protocol version 2 (http/2). RFC 7540, RFC Editor, May 2015. URL http://www.rfc-editor.org/rfc/rfc7540.txt. Accessed on Jun. 06. 2019.

Juan Benet. Ipfs-content addressed, versioned, p2p file system, July 2014. URL https://arxiv.org/abs/1407.3561. Accessed on Jun. 27. 2019.

Michael K. Bergman. Advantages and myths of rdf. In *Semantic Scholar*. AI3, Adaptive Information blog, 2009. URL https://api.semanticscholar.org/CorpusID:18286677. Accessed on Sep. 05. 2023.

Tim Berners-Lee, Larry Masinter, and Mark McCahill. Uniform resource locators (url). RFC 1738, RFC Editor, December 1994. URL http://www.rfc-editor.org/rfc/rfc1738.txt,. Accessed on Sep. 05. 2023.

Christian Bizer, Tom Heath, Kingsley Idehen, and Tim Berners-Lee. Linked data on the web (ldow2008). In *Proceedings of the 17th international conference on World Wide Web*, pages 1265–1266. Association for Computing Machinery New York, NY, United States, 2008.

Christian Bizer, Tom Heath, and Tim Berners-Lee. Linked data-the story so far. *Semantic services, interoperability and web applications: emerging concepts*, pages 205–227, 2009.

Sigrid Brell-Cokcan, Sven Stumm, Lukas Kirner, and Elisa Lublasser. Transparency and value of data in construction: Potentials within information networks for cross-company collaboration in the production chains of the construction industry. In *The Monetization of Technical Data: Innovations from Industry and Research*, pages 539–558. Springer, 2023.

buildingSMART International, Ltd. IFC Formats - buildingSMART Technical — technical.buildingsmart.org, 2023. URL https://technical.buildingsmart.org/standards/ifc/ifc-formats/. Accessed on Jul. 08. 2023.

Jeremy J Carroll. Signing rdf graphs. In *International Semantic Web Conference*, pages 369–384. Springer, 2003. https://doi.org/10.1007/978-3-540-39718-2_24.

Karen Coyle. Semantic web and linked data. *Library Technology Reports*, 48(4): 10–14, 2012.

Edward Curry, James ODonnell, Edward Corry, Souleiman Hasan, Marcus Keane, and Seán ORiain. Linking building data in the cloud: Integrating cross-domain building data using linked data. *Advanced Engineering Informatics, Elsevier*, 27 (2):206–219, 2013. https://doi.org/10.1016/j.aei.2012.10.003.

David Dias, Juan Benet, Mildred Ki'Lya, Richard Littauer, Ivan Vilata i Balaguer, Whyrusleeping, Lars Gierth, Steven Allen, JesseWeinstein, Friedel Ziegelmayer, Erick Lavoie, noffle, Christian Ferrier, Vasco Santos, Sid Harder, Rob Brackett, Kiko Marrs, W. Trevor King, Jakub Sztandera, David Grisham, Christian Couder, Harlan T Wood, Ed Mazurek, Eamonn O'Brien-Strain, Duke Jones, David A Roberts, Chris Oakman, Brendan O'Brien, Antonio Tenorio-Fornés, and Ali Mirlou. The specs for the ipfs protocol and associated subsystems, February 2023. URL https://github.com/ipfs/specs. Accessed on Sep. 05. 2023.

Theodoros Dounas, Wassim Jabi, and Davide Lombardi. Smart contracts for decentralised building information modelling. In *Proceedings of 38th Education and research in computer aided architectural design in Europe (eCAADe) conference 2020 (eCAADe 2020): anthropologic: architecture and fabrication in the cognitive age, 16-17 September 2020, [virtual conference]. Berlin.* Education

and Research in Computer Aided Architectural Design in Europe (eCAADe), 2020. ISBN 9789491207211.

Julian Dymcek. Survey of distributed hash tables. *Lane Department of Computer Science and Electrical Engineering, West Virginia University, Morgantown WV*, 2011.

Roy T. Fielding, James Gettys, Jeffrey C. Mogul, Henrik Frystyk Nielsen, Larry Masinter, Paul J. Leach, and Tim Berners-Lee. Hypertext transfer protocol – http/1.1. RFC 2616, RFC Editor, June 1999. URL http://www.rfc-editor.org/rfc/rfc2616.txt. Accessed on Jun. 06. 2019.

Roy Thomas Fielding. *Architectural styles and the design of network-based software architectures*. University of California, Irvine, 2000.

"International Organization for Standardization". Industrial automation systems and integrationproduct data representation and exchangepart 21: Implementation methods: Clear text encoding of the exchange structure, 2002.

Behrouz A Forouzan. *TCP/IP protocol suite*. McGraw-Hill, Inc., 2002.

Michael J Freedman and David Mazieres. Sloppy hashing and self-organizing clusters. In *International Workshop on Peer-to-Peer Systems*, pages 45–55. Springer, 2003.

Philipp Hagedorn, Liu Liu, Markus König, Rade Hajdin, Tim Blumenfeld, Markus Stöckner, Maximilian Billmaier, Karl Grossauer, and Kenneth Gavin. Bim-enabled infrastructure asset management using information containers and semantic web. *Journal of Computing in Civil Engineering*, 37(1):04022041, 2023.

Andreas Harth, Katja Hose, Marcel Karnstedt, Axel Polleres, Kai-Uwe Sattler, and Jürgen Umbrich. Data summaries for on-demand queries over linked data. In *Proceedings of the 19th international conference on World wide web*, pages 411–420. Association for Computing Machinery, 2010.

Bernhard Haslhofer and Niko Popitsch. Dsnotify-detecting and fixing broken links in linked data sets. In *2009 20th International Workshop on Database and Expert Systems Application*, pages 89–93. IEEE, 2009.

Tom Heath and Christian Bizer. Linked data: Evolving the web into a global data space. *Synthesis lectures on the semantic web. theory and technology, Springer Nature Switzerland Ag - Cham 6330, Gewerbestrasse 11 , HANDEL. Switzerland.*, 1(1):1–136, 2011. ISSN 2691-2023. doi: https://doi.org/10.1007/978-3-031-79432-2.

Alan R Hevner, Salvatore T March, Jinsoo Park, and Sudha Ram. Design science in information systems research. *MIS quarterly*, pages 75–105, 2004. https://doi.org/10.2307/25148625, JSTOR.

N Vu Hoang and S Törmä. Drumbeat platform - a web of building data implementation with backlinking. In *eWork and eBusiness in Architecture, Engineering and Construction: ECPPM 2016: Proceedings of the 11th European Conference on Product and Process Modelling (ECPPM 2016), Limassol, Cyprus, 7-9 September 2016*, page 155. CRC Press, 2017.

Tam T Huynh, Thuc D Nguyen, and Hanh Tan. A decentralized solution for web hosting. In *2019 6th NAFOSTED Conference on Information and Computer Science (NICS)*, pages 82–87. IEEE, 2019.

ISO. Information container for linked document delivery Exchange specificaton - Part 1: Container, March 2018a.

ISO. Information container for linked document delivery Exchange specificaton - Part 2: Dynamic semantics, March 2018b.

ISO ISO. 16739: 2013 industry foundation classes (ifc) for data sharing in the construction and facility management industries. *International Organization for Standardization*, 2013.

Gregg Kellogg, David I. Lehn, Dave Longley, mattcollier, and David Lozano Jarque. RDF::Normalize, July 2023. URL https://github.com/ruby-rdf/rdf-normalize. Accessed on Sep. 05. 2023.

Shah Khusro, Fouzia Jabeen, Syed Rahman Mashwani, and Iftikhar Alam. Linked open data: towards the realization of semantic web-a review. *Indian Journal of Science and Technology*, 7(6):745, 2014. Indian Society for Education and Environment.

Furkan Koç. A decentralized bim document management system with blockchain and ipfs integration for construction project delivery. Master's thesis, The Graduate School of Natural and Applied Sciences, Middle East Technical University,Çankaya, Ankara, Turkey, 2022. URL https://hdl.handle.net/11511/99593. Accessed on Sep. 05. 2023.

Benjamin Lehmann, Georg Jacobs, Christian Habermehl, Christian Gentz, and Stephan Neumann. Using data from agricultural and construction machinery profitably: Analysis of current monetisation mechanisms and recommendations for the future. In *The Monetization of Technical Data: Innovations from Industry and Research*, pages 427–472. Springer, 2023.

Dave Longley. Rdf dataset normalization: A standard rdf dataset normalization algorithm, Feb 2023. URL https://www.w3.org/TR/rdf-canon/. Accessed on Feb. 20. 2023.

Stephen Ludin and Javier Garza. *Learning HTTP/2: A Practical Guide for Beginners*. " O'Reilly Media, Inc.", 2017.

Petar Maymounkov and David Mazieres. Kademlia: A peer-to-peer information system based on the xor metric. In *International Workshop on Peer-to-Peer Systems*, pages 53–65. Springer, 2002.

Ralph Charles Merkle. *Secrecy, Authentication, and Public Key Systems*. PhD thesis, Stanford University. Department of Electrical Engineering, Stanford University Stanford Electronics Laboratories. Information Systems Laboratory, Stanford, CA, USA, 1979. AAI8001972.

Clyde L Monma and Victor K Wei. Intersection graphs of paths in a tree. *Journal of Combinatorial Theory, Series B*, 41(2):141–181, 1986. Elsevier.

Mehrdad Niknam and Saeed Karshenas. Sustainable design of buildings using semantic bim and semantic web services. *Procedia engineering*, 118:909–917, 2015. Elsevier.

Raja S Nizam and Cheng Zhang. A review of industry foundation class (ifc) based data exchange model. *Integrated Building Information Modelling, Bentham Science Publishers*, page 158, 2017. Bentham Science Publishers.

Mohamed Nour, Berthold Firmenich, Torsten Richter, and Christian Koch. A versioned ifc database for multi-disciplinary synchronous cooperation. In *Proceedings of the Joint International Conference on Computing and Decision Making in Civil and Building Engineering, Montreal, Canada*, pages 636–645. The ITC Digital Library,University of Ljubljana, Ljubljana, Slovenia, 2006.

Svenja Oprach, Tobias Bolduan, Dominik Steuer, Michael Vössing, and Shervin Haghsheno. Building the future of the construction industry through artificial intelligence and platform thinking. *Digitale Welt*, 3:40–44, 2019. Springer.

Jyrki Oraskari. jyrkioraskari/IFC-IPFS: The ICCCBE 2018 version of the ifcOWL/IPFS publishing code. In *GitHub*. Zenodo, CERN, February 2019a. doi: 10.5281/zenodo.2560560. URL https://doi.org/10.5281/zenodo.2560560. Accessed on Jun. 06. 2019.

Jyrki Oraskari. jyrkioraskari/ifcOWLSplitTests: ifcOWL Segmentation Test Results, July 2019b. URL https://doi.org/10.5281/zenodo.3267413. Accessed on Sep. 05. 2023.

Jyrki Oraskari. jyrkioraskari/ifc-ipfs: Ifc-ipfs v2.0, July 2023. URL https://doi.org/10.5281/zenodo.8156164. Accessed Oct. 05. 2023.

Jyrki Oraskari, Jakob Beetz, and Seppo Törmä. Federated Building Models in collaborative AEC/FM settings using IPFS. In *[17th International Conference on Computing in Civil and Building Engineering, 2018-06-05 - 2018-06-07, Tampere, Finland]*. 17th International Conference on Computing in Civil and Building Engineering, Tampere (Finland), 5 Jun 2018 - 7 Jun 2018, Ex Ordo,

Galway, Ireland, Jun 2018. URL http://programme.exordo.com/icccbe2018/delegates/presentation/354/. Accessed on Sep. 05. 2023.

Cesare Pautasso. Restful web services: principles, patterns, emerging technologies. In *Web Services Foundations*, pages 31–51. Springer, 2014.

Pieter Pauwels. Supporting decision-making in the building life-cycle using linked building data. *Buildings*, 4(3):549–579, 2014. Multidisciplinary Digital Publishing Institute.

Pieter. Pauwels and Walter Terkaj. EXPRESS to OWL for Construction Industry: Towards a Recommendable and Usable ifcOWL Ontology. *Automation in Construction*, 63:100–133, 2016. 10.1016/j.autcon.2015.12.003, Elsevier.

Pieter Pauwels, Seppo Törmä, Jakob Beetz, Matthias Weise, and Thomas Liebich. Linked data in architecture and construction. *Special Issue Automation in Construction, erscheint Juni*, 2015.

Pieter Pauwels, Jyrki Oraskari, and Lewis John McGibbney. Ifctordf, June 2019. URL https://github.com/pipauwel/IFCtoRDF. (Accessed on Sep. 05. 2023).

Ken Peffers, Marcus Rothenberger, Tuure Tuunanen, and Reza Vaezi. Design science research evaluation. In *Design Science Research in Information Systems. Advances in Theory and Practice: 7th International Conference, DESRIST 2012, Las Vegas, NV, USA, May 14-15, 2012. Proceedings 7*, pages 398–410. Springer, 2012.

Michael J Pratt. Introduction to iso 10303 - the step standard for product data exchange. *Journal of Computing and Information Science in Engineering*, 1(1): 102–103, 2001. American Society of Mechanical Engineers.

Protocol Labs, Inc. Content identifiers (cids), June 2019. URL http://web.archive.org/web/20190603152509/https://docs.ipfs.io/guides/concepts/cid/. Accessed on Feb. 20. 2023.

Enayat Rajabi, Salvador Sanchez-Alonso, and Miguel-Angel Sicilia. Analyzing broken links on the web of data: An experiment with dbpedia. *Journal of the Association for Information Science and Technology*, 65(8):1721–1727, 2014. Wiley Online Library.

Leonard Richardson and Sam Ruby. *Restful Web Services*. O'Reilly Media, first edition edition, may 2007. ISBN 0596529260.

Alex Rodriguez. Restful web services: The basics. *IBM developerWorks, The International Business Machines Corporation*, 33:18, 2008.

Miguel-Angel Sicilia, Salvador Sánchez-Alonso, and Elena García-Barriocanal. Sharing linked open data over peer-to-peer distributed file systems: the case of ipfs. In *Research Conference on Metadata and Semantics Research*, pages 3–14. Springer, 2016.

Daniel Stenberg. Http/3 explained - the book, July 2019. URL https://daniel.haxx.se/http3-explained/. Accessed on Jul. 07. 2019.

Giovanni Tummarello, Christian Morbidoni, Paolo Puliti, and Francesco Piazza. Signing individual fragments of an rdf graph. In *Special interest tracks and posters of the 14th international conference on World Wide Web*, pages 1020–1021, 2005. https://doi.org/10.1145/1062745.1062848.

Pierre-Yves Vandenbussche, Jürgen Umbrich, Luca Matteis, Aidan Hogan, and Carlos Buil-Aranda. Sparqles: Monitoring public sparql endpoints. *Semantic Web*, 8(6):1049–1065, 2017. IOS Press.

Ruben Verborgh, Miel Vander Sande, Pieter Colpaert, Sam Coppens, Erik Mannens, and Rik Van de Walle. Web-scale querying through linked data fragments. In *Proceedings of the Workshop on Linked Data on the Web (LDOW 2014), Seoul, Korea, April 8, 2014*. CEUR Workshop Proceedings, 2014.

World Wide Web Consortium comminity. Sparqlendpoints, November 2017. URL https://www.w3.org/wiki/SparqlEndpoints. Accessed on Jun. 06. 2019].

7.3 Discussion and Conclusions

Everything hinges on information availability: access and persistence. A separate challenge lies in establishing a standardized architecture to enable decentralized data management, which is essential in large organizations aiming to create a single source of truth for assets. This chapter focuses on the third part of the information security triad, availability. It is also directly related to the data storage aspect of data management mentioned in Section 1.5.

In the use scenario of the article presented at 7.1, an automatic model replication system can be implemented. An external party needs to handle IPFS content addressed when a to-be-saved model is released. The research shows that a pub sub-channel of the IPFS system can be used in communication. Here, it is enough that the architectural office publishes the model into IPFS and, when done, sends the IPFS content address of the saved work to the shared pub/sub channel (hosted as a part of the IPFS network). A party listening to the updates can read (and save) the named models to the local store in time. The publisher (the architectural office) also need to index the models so that when, e.g., the facility managers office asks for a copy, they can offer the corresponding address. The IPFS network will take care of the saving and data transfer of the models.

I have looked at how the ifcOWL building data models can be split into stable components, published peer-to-peer and represented as version data. Splitting the ifcOWL model into stable parts that can withstand version changes is a valuable solution to many research problems. It allows version control and data sharing over slow communication links by transferring only the necessary data. The idea of partitioning is based on the structure of the IFC standard (here, IFC 2x3), where the central part of the model can be partitioned by IFC elements with a global ID (GUID). Therefore, GUIDs were also used in this work to index the sub-models.

Peer-to-peer sharing of building information is beneficial in terms of accessibility. In three cases, the benefit is clear: 1. when the data publisher's server is down for some reason, 2. when a firewall blocks access to the server, and 3. when the data publisher's server is down for some reason. The use of peer-to-peer networks for data sharing has also been shown to reduce data retrieval latency. We demonstrated these benefits in the work I presented in Section 7.1. We also looked at how file size affects accessibility. This has implications beyond efficiency. The probability of finding a replica is higher if the information to be shared is not broken into small pieces (this comes from probability: if the probability of a small primitive search is something, for a bundle of primitives, it is enough that one of the primitives is selected. So the probability of selection is more critical).

As far as long storage is concerned, the situation is mixed. It is true that if a model has many users, and thus replicas in a peer-to-peer network, its likelihood of persistence is high. An example of such data is the copies of Wikipedia. For construction data, the situation is more uncertain. Therefore, in my research, I also consider a scenario where projects have one or more trusted parties (e.g. a government agency) that take care of data storage. I have proposed two mechanisms for how this might work.

There was much variation in the effectiveness of the IPFS network I tested. There were outliers, but the patterns were apparent. Using the network increases data availability, and having a copy on the same network reduces data transfer latency.

From a data management perspective, having a reliable and scalable way of storing data is essential. Of these, replication addresses the former and provides the ability to recover. We have also shown that the system scales well for significant model releases (e.g. we have tested the University of Auckland, Department of Computer Science Open IFC Model Repository[9] 0.8 million RDF triples with the West Riv-001 model). Data security is an issue that needs further research. I have suggested two options from

[9] https://www.re3data.org/repository/r3d100012443

a privacy perspective: private networks, which are possible in IPFS, and data encryption. The confidentiality issue is discussed more in section 5.

Data research questions related to data availability focused on understanding how IPFS (InterPlanetary File System) could be used to publish Linked Building data, the specific application of IPFS in this domain, any problems encountered, and how they could be solved. In addition, the research examined the performance characteristics of the proposed solution. This study presented the steps required to publish an LBD (ifcOWL) model in IPFS. It was found that the content in IPFS was available even when the original building data publishing host was down. Compared to web-based Linked Data, IPFS is superior because the data is automatically cached.

Chapter 8

Discussion and Conclusions

I chose to study how data management of building information can be achieved using linked data and open standards. This thesis has four levels: The first presents the distilled essence of data management: provision of information, supervision of the work and data content, data governance, versioning, provenance, and information security. From them have derived the overarching research questions, which I address in six different chapters. The first focuses on data representation as assertions, the second on terminology assertions (the ontologies), then the topics in the standard Common Data Environment, such as the single source of truth using open standards or the ability to manage one's data. The latter three deal with the triad of information security: confidentiality, integrity and availability. I use a set of hypotheses at the third level to assess the questions. I discuss how the article's topic supports or refutes the hypothesis presented. At the lowest level are the research questions that the articles directly answered.

The provision of information was handled when presenting the converter techniques and ontologies. It is strongly connected with data versioning and how the data can be shared using innovative networks like IPFS. The provenance of the data was handled in many aspects:

the dereferencing of the URIs in the Linked Data principles (Section 1.3) gives a precise internet location for the data. We also have discussed the concept of the single source of truth in the Common Data Environment context and the idea that everyone can host their data. The author information for the comments for the BIM Collaboration Format were handled when considering the bcfOWL ontology, and then, the principle that rights to modify data should be limited for those who have expertise in the domain (Section 5.1) and there is a clear responsible party for the content.

Data management in supervision and control was handled, proving that the Linked Building Data ontologies make data more human-accessible (Section 2.1), a smart semantic searching engine can help to find relevant information (Section 2.4), in works related to issue management (Sections 3.1, 3.2, and 4.4), and implementing version control provides access to the history of the data (Sections 2.1 and 3.2). Finally, in Chapters 5, 6, and 7 the security triad of availability, confidentiality and integrity were handled.

The introduced facility management usage scenarios (see Chapter 1) depict the general data management needs. One core principle has been giving the worker all the relevant information supporting making decisions. It was shown how data can be handed over and put into a local semantic knowledge base, how it can be updated with new information and how links in existing data can be kept up to date. Then, It was shown how issue comments can be made more findable and how all the saved knowledge of the issues, including the history, can be more accessible. So, the facility manager found the earlier comment about the air terminals without knowing the exact needed keywords to find the information. Also, it was shown how the guarantee that a found piece of information is the most up-to-date can help the team to collaborate. Information security was also handled. For example, in the sub topic of data availability, it was shown how the peer-to-peer access of IPFS can help a facility manager when the network connection is inadequate or non-existent. In this case, the intelligent system fetches the needed information from

nearby office users' computers.

For each article, I have highlighted how it implemented the design science research method following the guidelines of Hevner et al., what artefacts were produced, how the results were evaluated and how they can be generalised. Figures 8.1, 8.2 and 8.3 summarise this. The problem areas that stand out in the studies are converters, Model View Definitions, LBD ontologies and the Common Data Environment. The artefacts form a whole around the topic of Linked Data Management.

In each article section, I assessed how much the research presented supports the associated hypotheses. Then, for each chapter, the hypotheses are linked to the overarching research questions and, thus, to the data management themes I selected initially. In addition, in the discussion section of the chapters, I have indicated how the issues discussed relate to the data management themes mentioned above.

Following the FAIR principles [Wilkinson et al., 2016], data should be findable, accessible, interoperable, and reusable. It is considered one view of data management. I have handled findability at the beginning of the thesis (in Chapter 2, accessibility is spread in many subtopics (the Linked Data principles, the concept of the single source of truth, access control, and theme of availability in Chapter 7. The need for interpretability and the aim to use open standards to achieve it was mentioned in the introduction. Also, reusability in the viewpoint of ontologies and linking was handled. They all were handled in the thesis.

In conclusion, this dissertation has explored various aspects of Linked Building data management. Data management is like a fractal; the more you study it, the more topics you will find. Therefore, I have selected subjects to cover the concept and selected research questions to gain insights. The findings on data conversion, ontologies, CDE and information security are part of the scientific discussion on improving managing Linked Building Data. I hope people are finding them useful.

Table 8.1. Summary of the Design Science Method in the work 1/3.

Publication	Artifacts	Problem relevance	Evaluation	Research rigor	Communication
Bonduel et al., The IFC to linked building data converter-current status., 2018	The IFCtoLBD converter	**Problem Class:** Converters **BIM use-cases:** Quantity take off, Building operation and maintenance	Analytical Static Analysis	Peer review, published source code	**LDAC, 2018**
Bourreau & Oraskari, Dynamic BIM model conversion as inference-based ontology alignment., 2021	ifcOWL to LBD, SWRL conversion rules, Apache Jena inference model tests	**Problem Class:** Converters **BIM use-cases:** Quantity take off, Building operation and maintenance	Functional (Black Box) Testing and Experimental Controlled Experiment	Peer review, included code	**LDAC, 2021**
Oraskari et al., SHACL is for LBD what mvdXML is for IFC., 2021	SHACL validation and filtering rules	**Problem Classes:** Model View Definition, unit testing, **BIM use-cases:** Quantity take off, Building operation and maintenance	Descriptive Scenario	Peer review, included code	**CIB W78, 2021**
Kuo & Oraskari, A predictive semantic inference system using BIM Collaboration Format (BCF) cases and machine learning.,2016	An example implementation for a search tool to search BCF comments.	**Problem classes:** queries, machine learning **BIM use-case:** Monitor Maintenance and Monitor Assets	Experimental Controlled Experiment	Peer review	**WBC, 2016**
Schulz et al., bcfOWL: A BIM collaboration ontology., 2021	The bcfOWL ontology	**Problem Class:** LBD ontologies **BIM use-case:** BCF Communication, Monitor Maintenance, and Monitor Assets	Descriptive Scenario	Peer review, published ontology and test data	**LDAC, 2021**

Table 8.2. Summary of the Design Science Method in the work 2/3.

Publication	Artifacts	Problem relevance	Evaluation	Research rigor	Communication
Oraskari et al., Towards describing version history of BCF data in the Semantic Web., 2022	Implementations of the presented four schenario updates to the bcfOWL ontology.	**Problem Class:** LBD ontologies, version management **BIM use-case:** BCF Communication, Monitor Maintenance, and Monitor Assets	Descriptive Scenario	Peer review, published ontology changes, queries and test data	**LDAC, 2022**
Oraskari, Live Web Ontology for buildingSMART Data Dictionary., 2021	The dynamic bSDD ontology	**Problem Class:** LBD ontologies **BIM use-case:** Building operation and maintenance	Controlled Experiment and a Functional (Black Box) Testing	Peer review, published source code	**FBI, 2021**
Senthilvel et al., Common Data Environments for the Information Container for linked Document Delivery., 2020.	Mapping rules from ICDD to LDP and OpenCDE	**Problem class:** the Common Data Environment, conversion	Analytical Static Analysis	Peer review	**LDAC, 2020**
Senthilvel et al., Implementing Information Container for linked Document Delivery (ICDD) as a micro-service., 2021.	An OpenCDE API implemetation that supports ICDD uploads	**Problem class:** the Common Data Environment, conversion	Analytical Static Analysis	Peer review, published source code	**EG-ICE, 2021**
Oraskari et al., Enabling Federated Interoperable Issue Management in a Building and Construction Sector, 2022	Example construction for BCF data	**Problem class:** the Common Data Environment	Analytical Static Analysis	Peer review, published data	**EG-ICE, 2022**

Table 8.3. Summary of the Design Science Method in the work 3/3.

Publication	Artifacts	Problem relevance	Evaluation	Research rigor	Communication
Werbrouckx et al., A Generic Framework for Federated CDEs applied to Issue Management, 2023	Example construction for BCF data	**Problem class:** the Common Data Environment	Analytical Static Analysis	Peer review, published data	**AEI, 2023**
Oraskari & Törmä, Access control for web of building data: Challenges and directions., 2016	Security roles, to be secured entities, access rights	**Problem class:** the Common Data Environment, access control	Descriptive Scenario	Peer review	**ECPPM, 2016**
Oraskari & Törmä, RDF-based signature algorithms for computing differences of IFC models., 2015	Methods to calculate checksums for sub-graphs of ifcOWL	**Problem class:** the Common Data Environment, integrity, checksums	Experimental Controlled Experiment	Peer review, published source code	**AC, 2015**
Oraskari, Computationally Efficient IFC File Comparison Using RDF Graph Paths., 2015	Methods to calculate checksums for sub-graphs of ifcOWL	**Problem class:** the Common Data Environment, integrity, checksums	Experimental Controlled Experiment	Peer review, published source code	**CIB W78, 2015**
Oraskari et al., Federated Building Models in collaborative AEC/FM settings using IPFS., 2018	A method to publish LBD in the IPFS peer-to-peer network	**Problem class:** the Common Data Environment, availability	Descriptive Scenario, Experimental Controlled Experiment	Peer review, published source code	**ICCCBE, 2018**
Oraskari et al., Sharing Linked Building Data in a Peer-to-Peer Network — ifcOWL meets InterPlanetary File System, 2018	A method to publish LBD in the IPFS peer-to-peer network	**Problem class:** the Common Data Environment, availability	Descriptive Scenario, Experimental Controlled Experiment	Peer review, published source code	**J. Comput. Civ. Eng., 2018**

List of Publications

Mathias Bonduel, Jyrki Oraskari, Pieter Pauwels, Maarten Vergauwen, and Ralf Klein. The ifc to linked building data converter - current status. *Proceedings of the 6th Linked Data in Architecture and Construction Workshop*, 2159:34–43, 2018. ISSN 1613-0073. URL https://ceur-ws.org/Vol-2159/04paper.pdf. International Workshop on Linked Data in Architecture and Construction, LDAC ; Conference date: 19-06-2018 Through 21-06-2018, CEUR-WS Team, Redaktion Sun SITE, Informatik V, RWTH Aachen, Germany.

Pierre Bourreau and Jyrki Oraskari. Dynamic bim format conversion as inference-based ontology alignment. In *Proceedings of the 9th Linked Data in Architecture and Construction Workshop - LDAC2021. CEUR-WS Team, Redaktion Sun SITE, Informatik V, RWTH Aachen, Germany.*, 2021. URL https://ceur-ws.org/Vol-3081/11paper.pdf. International Workshop on Linked Data in Architecture and Construction, LDAC ; Conference date: 11-10-2021 Through 15-10-2021, Online; accessed Oct. 20. 2023.

Lukas Kirner, Jyrki Oraskari, Victoria Jung, and Sigrid Brell-Cokcan. dstv: An ontology-based extension of the dstv-nc standard for the use of linked data in the automation of steel construction. In *Proceedings of the 11th Linked Data in Architecture and Construction Workshop*, 2023. URL https://linkedbuildingdata.net/ldac2023/files/papers/papers/LDAC2023_paper_1731.pdf. 11th International Workshop on Linked Data in Architecture and Construction, LDAC; Conference date: 15 - 16 June 2023, Online; accessed Oct. 20. 2023.

Vincent Kuo and Jyrki Oraskari. A predictive semantic

inference system using bim collaboration format (bcf) cases and machine learning. In Arto Saari and Pekka Huovinen, editors, *WBC16 Proceedings : Volume III*, volume 3 of *Raportti / Tampereen teknillinen yliopisto, rakennustekniikan laitos, rakennustuotanto ja -talous*, pages 368–381. Tampere University of Technology, Tampere, Finland, 2016. URL https://trepo.tuni.fi//bitstream/handle/10024/128445/WBC16_Vol_3.pdf. CIB World Building Congress, WBC ; Conference date: 30-05-2016 Through 03-06-2016, Online; accessed Oct. 15. 2023.

J. Oraskari and S. Törmä. Access control for web of building data: Challenges and directions. In *eWork and eBusiness in Architecture, Engineering and Construction*, pages 45–53, 2016. ISBN 9781138032804. doi: https://doi.org/10.1201/9781315386904. European Conference on Product and Process Modelling, ECPPM ; Conference date: 07-09-2016 Through 09-09-2016.

Jyrki Oraskari. Computationally efficient ifc file comparison using rdf graph paths. In *ITC Digital library*, pages 599–608, 2015. URL https://itc.scix.net/paper/w78-2015-paper-062. CIB W78 Conference , CIB W78, Eindhoven, The Netherlands; Conference date: 27-10-2015 Through 29-10-2015, Online; accessed Oct. 15. 2023.

Jyrki Oraskari. Live web ontology for buildingsmart data dictionary. In *32. Forum Bauinformatik 2021. Institut für Numerische Methoden und Informatik im Bauwesen, TU Darmstadt (2022)*, volume 32, pages 166–173, July 2021. doi: 10.26083/tuprints-00019496. URL https://tuprints.ulb.tu-darmstadt.de/21521/. Forum Bauinformatik, FBI, Darmstadt, Germany; Conference date: 09-09-2021 Through 10-09-2021, Online; accessed Oct. 15. 2023.

Jyrki Oraskari and Seppo Törmä. Rdf-based signature algorithms for computing differences of ifc models. *Automation in Construction*, 57: 213–221, 2015. ISSN 0926-5805. doi: https://doi.org/10.1016/j.autcon.2015.05.008. URL https://www.sciencedirect.com/science/article/pii/S0926580515001181. Online. Accessed Oct. 20. 2023.

Jyrki Oraskari, Jakob Beetz, and Seppo Törmä. Federated Building Models in collaborative AEC/FM settings using IPFS. In *[17th International Conference on Computing in Civil and Building Engineering, 2018-06-05 - 2018-06-07, Tampere, Finland]*. 17th International Conference on Computing in Civil and Building Engineering, Tampere (Finland), 5 Jun 2018 - 7 Jun 2018, Ex Ordo, Galway, Ireland, Jun 2018. URL http://programme.exordo.com/icccbe2018/delegates/presentation/354/. Accessed on Sep. 05. 2023.

Jyrki Oraskari, Madhumitha Senthilvel, and Jakob Beetz. Shacl is for lbd what mvdxml is for ifc. In *Proceedings of the 38th International Conference of CIB W78, Luxembourg, 13-15 October*, CIB W78 conference series, pages 693–702. CIB International Council for Research and Innovation in Building and Construction, 2021. URL https://www.cibw78-ldac-2021.lu/. CIB W78 Conference ; Conference date: 11-10-2021 Through 15-10-2021, Online; accessed Oct. 15. 2023.

Jyrki Oraskari, Oliver Schulz, and Jakob Beetz. Towards describing version history of BCF data in the semantic web. In *Proceedings of the 10th Linked Data in Architecture and Construction Workshop co-located with 19th European Semantic Web Conference (ESWC 2022). CEUR-WS Team, Redaktion Sun SITE, Informatik V, RWTH Aachen, Germany*. CEUR-WS, 2022a. https://ceur-ws.org/Vol-3213/paper08.pdf. Accessed November 8, 2024.

Jyrki Oraskari, Oliver Schulz, Jeroen Werbrouck, and Jakob Beetz. Enabling federated interoperable issue management in a building and construction sector. In *Proceedings of the 29th EG-ICE International Workshop on Intelligent Computing in Engineering. Aarhus University Library/Det Kgl. Bibliotek - AU Library Scholarly Publishing Services. Aarhus, Denmark*, pages 92–101. EG-ICE, 2022b. ISBN 978-87-7507-521-8. URL https://ebooks.au.dk/aul/catalog/view/455/312/1848-2. http://doi.org/10.7146/aul.455.c200, Online; accessed Oct. 15. 2023.

Jyrki Oraskari, Seppo Törmä, Mehmet Yalcinkaya, Vishal Singh, and Jakob Beetz. Sharing linked building data

in a peer-to-peer network: ifcowl meets interplanetary file system. *Journal of Computing in Civil Engineering. 1801 Alexander Bell Drive Reston, VA 20191-4400*, 38 (1):04023037, 2024. doi: 10.1061/JCCEE5.CPENG-5381. URL https://ascelibrary.org/doi/abs/10.1061/JCCEE5.CPENG-5381. published online on September 28, 2023.

Oliver Schulz, Jyrki Oraskari, and Jakob Beetz. bcfowl: A BIM collaboration ontology. In María Poveda-Villalón and Pieter Pauwels, editors, *Proceedings of the 9th Linked Data in Architecture and Construction Workshop, Luxembourg, Luxembourg, October 11-13, 2021*, volume 3081 of *CEUR Workshop Proceedings*, pages 142–153. CEUR-WS.org, 2021. URL http://ceur-ws.org/Vol-3081/12paper.pdf. Online; accessed Oct. 15. 2023.

Oliver Schulz, Jyrki Oraskari, and Jakob Beetz. Lessons learned from designing and using bcfowl. In *Proceedings of the 11th Linked Data in Architecture and Construction Workshop*, 2023. URL https://linkedbuildingdata.net/ldac2023/files/papers/papers/LDAC2023_paper_4665.pdf. 11th International Workshop on Linked Data in Architecture and Construction, LDAC; Conference date: 15 - 16 June 2023, Online; accessed Oct. 15. 2023.

Madhumitha Senthilvel, Jyrki Oraskari, and Jakob Beetz. Common data environments for the information container for linked document delivery. *Proceedings of the 8th Linked Data in Architecture and Construction Workshop - LDAC2020. CEUR-WS Team, Redaktion Sun SITE, Informatik V, RWTH Aachen, Germany*, 2636:132–145, June 2020. ISSN 1613-0073. URL https://ceur-ws.org/Vol-2636/10paper.pdf. International Workshop on Linked Data in Architecture and Construction, LDAC; Conference date: 17-06-2020 Through 19-06-2020, Online; accessed Oct. 20. 2023.

Madhumitha Senthilvel, Jyrki Oraskari, and Jakob Beetz. Implementing information container for linked document delivery (icdd) as a micro-service. In *EG-ICE 2021 Proceedings*, page 6675. Universitätsverlag der TU Berlin, ISBN 978-3-7983-3212-6, August 2021. doi: 10.14279/depositonce-12021. URL

https://berlin-2021.eg-ice.org/. International Workshop on Intelligent Computing in Engineering, EG-ICE ; Conference date: 30-06-2021 Through 02-07-2021, Online; accessed Oct. 15. 2023.

Wojciech Teclaw, Jyrki Oraskari, Mads Holten Rasmussen, Nathalie Labonnote, and Eilif Hjelseth. Building system data integration using semantic representation. 2023a. URL https://www.ucl.ac.uk/bartlett/construction/sites/bartlett_construction/files/building_system_data_integration_using_semantic.pdf. The 30th EG-ICE: International Conference on Intelligent Computing in Engineering, Online; Accessed Oct. 17. 2023.

Wojciech Teclaw, Madsholten Rasmussen, Nathalie Labonnote, Jyrki Oraskari, and Eilif Hjelseth. The semantic link between domain-based bim models. In *Proceedings of the 11th Linked Data in Architecture and Construction Workshop*, 2023b. URL https://linkedbuildingdata.net/ldac2023/files/papers/papers/LDAC2023_paper_4062.pdf. 11th International Workshop on Linked Data in Architecture and Construction, LDAC; Conference date: 15 - 16 June 2023, Online; accessed Oct. 20. 2023.

Jeroen Werbrouck, Oliver Schulz, Jyrki Oraskari, Erik Mannens, Pieter Pauwels, and Jakob Beetz. A generic framework for federated cdes applied to issue management. *Advanced Engineering Informatics. Elsevier, Amsterdam, Netherlands*, 58:102136, 2023. ISSN 1474-0346. doi: https://doi.org/10.1016/j.aei.2023.102136. URL https://www.sciencedirect.com/science/article/pii/S1474034623002641.

Bibliography

bcfOWL: A BIM collaboration ontology, 2021. URL https://ceur-ws.org/Vol-3081/12paper.pdf. International Workshop on Linked Data in Architecture and Construction, LDAC; Conference date: 11-10-2021 Through 15-10-2021, Online; accessed Nov. 08. 2024.

Joan E. van Aken. Management research based on the paradigm of the design sciences: the quest for field-tested and grounded technological rules. *Journal of management studies*, 41(2):219–246, 2004. Wiley Online Library.

Wael Alghamdi. *Secure data collection in wireless sensor networks*. PhD thesis, UNSW Sydney, 2017.

Dimitrios Athanasiou, Ariadi Nugroho, Joost Visser, and Andy Zaidman. Test code quality and its relation to issue handling performance. *IEEE Transactions on Software Engineering*, 40(11):1100–1125, 2014. Institute of Electrical and Electronics Engineers.

D. Beckett and B. McBride. Rdf/xml syntax specification (revised). *W3C recommendation*, 10, 2004.

Jakob Beetz, Jos Van Leeuwen, and Bauke De Vries. Ifcowl: A case of transforming express schemas into ontologies. *Ai Edam*, 23(1):89–101, 2009. Cambridge University Press.

M. Belshe, R. Peon, and M. Thomson. Hypertext transfer protocol version 2 (http/2). RFC 7540, RFC Editor, May 2015. URL http://www.rfc-editor.org/rfc/rfc7540.txt. Accessed on Jun. 06. 2019.

Tim Berners-Lee. Linked data, 2006. URL https://www.w3.org/DesignIssues/LinkedData. Online; accessed Oct. 15. 2023.

Tim Berners-Lee, Larry Masinter, and Mark McCahill. Uniform resource locators (url). RFC 1738, RFC Editor, December 1994. URL http://www.rfc-editor.org/rfc/rfc1738.txt,. Accessed on Sep. 05. 2023.

Tim Berners-Lee, Roy T. Fielding, and Larry Masinter. Uniform resource identifier (uri): Generic syntax. STD 66, RFC Editor, January 2005. URL http://www.rfc-editor.org/rfc/rfc3986.txt. http://www.rfc-editor.org/rfc/rfc3986.txt, Online; accessed Oct. 15. 2023.

Abraham Bernstein and Natasha Noy. Is this really science? the semantic webbers guide to evaluating research contributions. Technical report, University of Zurich, Department of Informatics (IFI), Binzmühlestrasse 14, CH-8050 Zürich, Switzerland, 2014. URL https://www.merlin.uzh.ch/contributionDocument/download/6915. Online; accessed Oct. 15. 2023.

Christian Bizer, Tom Heath, Kingsley Idehen, and Tim Berners-Lee. Linked data on the web (ldow2008). In *Proceedings of the 17th international conference on World Wide Web*, pages 1265–1266. Association for Computing Machinery New York, NY, United States, 2008.

Christian Bizer, Tom Heath, and Tim Berners-Lee. Linked data-the story so far. *Semantic services, interoperability and web applications: emerging concepts*, pages 205–227, 2009.

Mathias Bonduel. An introduction to best practices, w3c lbd cg telco, 23 mar 2021. In *W3C Linked Building Data Community Group*, 2021.

Mathias Bonduel, Jyrki Oraskari, Pieter Pauwels, Maarten Vergauwen, and Ralf Klein. The ifc to linked building data converter - current status. *Proceedings of the 6th Linked Data in Architecture and Construction Workshop*, 2159:34–43, 2018. ISSN 1613-0073. URL https://ceur-ws.org/Vol-2159/04paper.pdf. International Workshop

on Linked Data in Architecture and Construction, LDAC ; Conference date: 19-06-2018 Through 21-06-2018, CEUR-WS Team, Redaktion Sun SITE, Informatik V, RWTH Aachen, Germany.

Carsten Bormann, Angelo P Castellani, and Zach Shelby. Coap: An application protocol for billions of tiny internet nodes. *IEEE Internet Computing*, 16(2):62–67, 2012.

André Borrmann, Jakob Beetz, Christian Koch, Thomas Liebich, and Sergej Muhic. Industry foundation classes: A standardized data model for the vendor-neutral exchange of digital building models. *Building information modeling: Technology foundations and industry practice, Springer*, pages 81–126, 2018a.

André Borrmann, Markus König, Christian Koch, and Jakob Beetz. Building information modeling: Why? what? how? In *Building information modeling*, pages 1–24. Springer, 2018b.

Pierre Bourreau and Jyrki Oraskari. Dynamic bim format conversion as inference-based ontology alignment. In *Proceedings of the 9th Linked Data in Architecture and Construction Workshop - LDAC2021. CEUR-WS Team, Redaktion Sun SITE, Informatik V, RWTH Aachen, Germany.*, 2021a. URL https://ceur-ws.org/Vol-3081/11paper.pdf. International Workshop on Linked Data in Architecture and Construction, LDAC ; Conference date: 11-10-2021 Through 15-10-2021, Online; accessed Oct. 20. 2023.

Pierre Bourreau and Jyrki Oraskari. Bim format conversion as inference-based ontology alignment. In *Proceedings of the 9th Linked Data in Architecture and Construction Workshop - LDAC2021, CEUR-WS Team, Redaktion Sun SITE, Informatik V, RWTH Aachen, Germany.*, 2021b. URL https://ceur-ws.org/Vol-3081/11paper.pdf. Online; accessed Oct. 15. 2023.

Tim Bray. The javascript object notation (json) data interchange format. RFC 7158, RFC Editor, March 2013. URL http://www.rfc-editor.org/rfc/rfc7158.txt. Online; accessed Oct. 15. 2023.

buildingSMART International. bsi use case management service (ucms), 2022. URL https://ucm.buildingsmart.org/. [Accessed 23-Dec-2022].

Cambridge Dictionary. Information management. In *Cambridge Dictionary*. Cambridge University Press, 2022. URL https://dictionary.cambridge.org/dictionary/english/information-management. Online; accessed Oct. 15. 2023.

Jeremy J Carroll. Signing rdf graphs. In *International Semantic Web Conference*, pages 369–384. Springer, 2003. https://doi.org/10.1007/978-3-540-39718-2_24.

Eastman Charles et al. An outline of the building descripition system, institute of physical planning, 1974. URL https://eric.ed.gov/?id=ED113833. [Online; accessed 16-December-2022].

Ke Chen, Weisheng Lu, Hongdi Wang, Yuhan Niu, and George G Huang. Naming objects in bim: A convention and a semiautomatic approach. *Journal of Construction Engineering and Management, American Society of Civil Engineers*, 143(7):06017001, 2017.

Efstathia Chioteli, Ioannis Batas, and Diomidis Spinellis. Does unit-tested code crash? a case study of eclipse. In *25th Pan-Hellenic Conference on Informatics*, pages 260–264, 2021.

Wikimedia Commons. Traditional construction handover documentation, 2008. URL https://en.wikipedia.org/wiki/COBie#/media/File:2008-07-14-COBIEBusinessCase-2.jpg. File: 2008-07-14-COBIEBusinessCase-2.jpg, Online; accessed Oct. 15. 2023.

Georg Dangl, linhard, Jon Anders Sollien, Nick Tindall, Ryan Schultz, Tormi Tabor, Klaus Aengenvoort, Lionel, JPurushothaman, and ulferts. BCFRESTAPI. https://github.com/buildingSMART/BCF-API, 2017. Online; accessed Oct. 15. 2023.

Charles Eastman. The use of computers instead of drawings in building design. *AIA journal*, 63(3): 46-50, 1975.

Chuck Eastman, Paul Teicholz, Rafael Sacks, Kathleen Liston, BIM Discover, and BIM Handbook. *BIM Handbook: A Guide to Building Information Modeling for Owners, Managers, Designers, Engineers, and Contractors*. John Wiley & Sons, 2011.

Roy T. Fielding, James Gettys, Jeffrey C. Mogul, Henrik Frystyk Nielsen, Larry Masinter, Paul J. Leach, and Tim Berners-Lee. Hypertext transfer protocol - http/1.1. RFC 2616, RFC Editor, June 1999. URL http://www.rfc-editor.org/rfc/rfc2616.txt. Accessed on Jun. 06. 2019.

Behrouz A Forouzan. *TCP/IP protocol suite*. McGraw-Hill, Inc., 2002.

Aldo Gangemi and Valentina Presutti. Ontology design patterns. In *Handbook on ontologies*, pages 221-243. Springer, 2009.

Leif Granholm. Role of open standards and bim (model-based information) in the construction industry workflow. In *European Conference on Product and Process Modeling 2022, 14th - 16th September, 2022. Trondheim, Norway*, 2022.

Bastiaan Grutters. Ifc formats, Nov 2022. URL https://technical.buildingsmart.org/standards/ifc/ifc-formats/. Online; accessed Oct. 15. 2023.

Fermín Guerrero. Common data environment (cde), Mar 2022. URL https://www.modelical.com/en/common-data-environment/. Online; accessed Oct. 15. 2023.

Jeroen Van Der Ham. Toward a better understanding of cybersecurity? *Digital Threats: Research and Practice*, 2(3):1-3, 2021. Association for Computing Machinery New York, NY, USA.

Alan R Hevner, Salvatore T March, Jinsoo Park, and Sudha Ram. Design science in information systems research. *MIS quarterly*, pages 75-105, 2004. https://doi.org/10.2307/25148625, JSTOR.

N Vu Hoang and S Törmä. Drumbeat platform - a web of building data implementation with backlinking. In *eWork and eBusiness in Architecture, Engineering and Construction: ECPPM 2016: Proceedings of the 11th European Conference on Product and Process Modelling (ECPPM 2016), Limassol, Cyprus, 7-9 September 2016*, page 155. CRC Press, 2017.

Mads Holten Rasmussen, Maxime Lefrançois, Mathias Bonduel, Christian Anker Hviid, and Jan Karlshøj. Opm: An ontology for describing properties that evolve over time. In *CEUR Workshop Proceedings*, volume 2159, pages 24-33. CEUR Workshop Proceedings, 2018.

Andy Hunt and Dave Thomas. *Pragmatic unit testing in Java with JUnit*. The Pragmatic Bookshelf, 2003.

Eero Hyvönen, Jouni Tuominen, Miika Alonen, and Eetu Mäkelä. Linked data finland: A 7-star model and platform for publishing and re-using linked datasets. In *European Semantic Web Conference*, pages 226-230. Springer, 2014.

International Organization for Standardization. Organization and digitization of information about buildings and civil engineering works, including building information modelling (BIM) Information management using building information modelling Part 1: Concepts and principles. Standard, International Organization for Standardization, Geneva, CH, December 2018a.

International Organization for Standardization. ISO 19650-1:2018: Organization and digitization of information about buildings and civil engineering works, including building information modelling (BIM) Information management using building information modelling Part 1: Concepts and principles. Standard, International Organization for Standardization, Geneva, CH, 12 2018b. URL https://www.iso.org/cms/render/live/en/sites/isoorg/contents/data/

standard/06/80/68078.html. Online; accessed Oct. 15. 2023.

ISO. Information container for linked document delivery Exchange specificaton - Part 1: Container, March 2018a.

ISO. Information container for linked document delivery Exchange specificaton - Part 2: Dynamic semantics, March 2018b.

ISO ISO. 16739: 2013 industry foundation classes (ifc) for data sharing in the construction and facility management industries. *International Organization for Standardization*, 2013.

Lukas Kirner, Jyrki Oraskari, Victoria Jung, and Sigrid Brell-Cokcan. dstv: An ontology-based extension of the dstv-nc standard for the use of linked data in the automation of steel construction. In *Proceedings of the 11th Linked Data in Architecture and Construction Workshop*, 2023. URL https://linkedbuildingdata.net/ldac2023/files/papers/papers/LDAC2023_paper_1731.pdf. 11th International Workshop on Linked Data in Architecture and Construction, LDAC; Conference date: 15 - 16 June 2023, Online; accessed Oct. 20. 2023.

Arto Kiviniemi. Iai and ifc-state-of-the-art. In *Proceedings of the 1999 CIB w78 Conference*, pages 2157-2169, 1999.

Vincent Kuo and Jyrki Oraskari. A predictive semantic inference system using bim collaboration format (bcf) cases and machine learning. In Arto Saari and Pekka Huovinen, editors, *WBC16 Proceedings : Volume III*, volume 3 of *Raportti / Tampereen teknillinen yliopisto, rakennustekniikan laitos, rakennustuotanto ja -talous*, pages 368-381. Tampere University of Technology, Tampere, Finland, 2016. URL https://trepo.tuni.fi//bitstream/handle/10024/128445/WBC16_Vol_3.pdf.

CIB World Building Congress, WBC ; Conference date: 30-05-2016 Through 03-06-2016, Online; accessed Oct. 15. 2023.

Ora Lassila, Ralph R Swick, et al. Resource description framework (rdf) model and syntax specification. *W3C recommendation*, 1998.

Paul J. Leach, Rich Salz, and Michael H. Mealling. A Universally Unique IDentifier (UUID) URN Namespace. RFC 4122, July 2005. URL https://www.rfc-editor.org/info/rfc4122. Accessed October 17, 2023.

Stephen Ludin and Javier Garza. *Learning HTTP/2: A Practical Guide for Beginners*. " O'Reilly Media, Inc.", 2017.

Salvatore T March and Gerald F Smith. Design and natural science research on information technology. *Decision support systems*, 15(4): 251-266, 1995. Elsevier.

Salvatore T. March and Veda C. Storey. Design science in the information systems discipline: an introduction to the special issue on design science research. *MIS quarterly*, pages 725-730, 2008. JSTOR.

Merriam-Webster. Management. In *Merriam-Webster.com dictionary*. Encyclopædia Britannica, Inc., 2022. URL https://www.merriam-webster.com/dictionary/management. Online; accessed Oct. 15. 2023.

J. Oraskari and S. Törmä. Access control for web of building data: Challenges and directions. In *eWork and eBusiness in Architecture, Engineering and Construction*, pages 45-53, 2016a. ISBN 9781138032804. doi: https://doi.org/10.1201/9781315386904. European Conference on Product and Process Modelling, ECPPM ; Conference date: 07-09-2016 Through 09-09-2016.

J Oraskari and S Törmä. Access Control for Web of Building Data: Challenges and Directions.

In Symeon Christodoulou and Raimar Scherer, editors, *eWork and eBusiness in Architecture, Engineering and Construction: ECPPM 2016*. CRC Press, London, UK, 2016b. ISBN 9781138032804, https://doi.org/10.1201/9781315386904.

Jyrki Oraskari. Computationally efficient ifc file comparison using rdf graph paths. In *ITC Digital library*, pages 599-608, 2015. URL https://itc.scix.net/paper/w78-2015-paper-062. CIB W78 Conference , CIB W78, Eindhoven, The Netherlands; Conference date: 27-10-2015 Through 29-10-2015, Online; accessed Oct. 15. 2023.

Jyrki Oraskari. Live web ontology for buildingsmart data dictionary. In *32. Forum Bauinformatik 2021. Institut für Numerische Methoden und Informatik im Bauwesen, TU Darmstadt (2022)*, volume 32, pages 166-173, July 2021. doi: 10.26083/tuprints-00019496. URL https://tuprints.ulb.tu-darmstadt.de/21521/. Forum Bauinformatik, FBI, Darmstadt, Germany; Conference date: 09-09-2021 Through 10-09-2021, Online; accessed Oct. 15. 2023.

Jyrki Oraskari and Seppo Törmä. Rdf-based signature algorithms for computing differences of ifc models. *Automation in Construction*, 57:213-221, 2015a. ISSN 0926-5805. doi: https://doi.org/10.1016/j.autcon.2015.05.008. URL https://www.sciencedirect.com/science/article/pii/S0926580515001181. Online. Accessed Oct. 20. 2023.

Jyrki Oraskari and Seppo Törmä. Rdf-based signature algorithms for computing differences of ifc models. *Automation in Construction, Elsevier B.V. Amsterdam, Netherlands*, 57: 213-221, 2015b. ISSN 0926-5805. doi: https://doi.org/10.1016/j.autcon.2015.05.008. URL https://www.sciencedirect.com/science/article/pii/S0926580515001181. Online. Accessed Oct. 20. 2023.

Jyrki Oraskari, Jakob Beetz, and Seppo Törmä. Federated Building Models in collaborative AEC/FM settings using IPFS. In *[17th International Conference on Computing in Civil and Building Engineering, 2018-06-05 - 2018-06-07, Tampere, Finland]*. 17th International Conference on Computing in Civil and Building Engineering, Tampere (Finland), 5 Jun 2018 - 7 Jun 2018, Ex Ordo, Galway, Ireland, Jun 2018a. URL http://programme.exordo.com/icccbe2018/delegates/presentation/354/. Accessed on Sep. 05. 2023.

Jyrki Oraskari, Jakob Beetz, and Seppo Törmä. Federated Building Models in collaborative AEC/FM settings using IPFS. In *[17th International Conference on Computing in Civil and Building Engineering, 2018-06-05 - 2018-06-07, Tampere, Finland]. Galway, Ireland: Galway, Ireland: Ex Ordo*. 17th International Conference on Computing in Civil and Building Engineering, Tampere (Finland), 5 Jun 2018 - 7 Jun 2018, Ex Ordo, Galway, Ireland, Jun 2018b. URL http://programme.exordo.com/icccbe2018/delegates/presentation/354/. Accessed on Sep. 05. 2023.

Jyrki Oraskari, Mathias Bonduel, Kris McGlinn, and Pieter Pauwels. jyrkioraskari/ifctolbd: Ifctolbd 2.33.0, December 2021a. URL https://doi.org/10.5281/zenodo.5772656.

Jyrki Oraskari, Madhumitha Senthilvel, and Jakob Beetz. Shacl is for lbd what mvdxml is for ifc. In *Proceedings of the 38th International Conference of CIB W78, Luxembourg, 13-15 October*, CIB W78 conference series, pages 693-702. CIB International Council for Research and Innovation in Building and Construction, 2021b. URL https://www.cibw78-ldac-2021.lu/. CIB W78 Conference ; Conference date: 11-10-2021 Through 15-10-2021, Online; accessed Oct. 15. 2023.

Jyrki Oraskari, Oliver Schulz, and Jakob Beetz.

Towards describing version history of BCF data in the semantic web. In *Proceedings of the 10th Linked Data in Architecture and Construction Workshop co-located with 19th European Semantic Web Conference (ESWC 2022). CEUR-WS Team, Redaktion Sun SITE, Informatik V, RWTH Aachen, Germany*. CEUR-WS, 2022a. https://ceur-ws.org/Vol-3213/paper08.pdf. Accessed November 8, 2024.

Jyrki Oraskari, Oliver Schulz, Jeroen Werbrouck, and Jakob Beetz. Enabling federated interoperable issue management in a building and construction sector. In *Proceedings of the 29th EG-ICE International Workshop on Intelligent Computing in Engineering. Aarhus University Library/Det Kgl. Bibliotek - AU Library Scholarly Publishing Services. Aarhus, Denmark*, pages 92-101. EG-ICE, 2022b. ISBN 978-87-7507-521-8. URL https://ebooks.au.dk/aul/catalog/view/455/312/1848-2. http://doi.org/10.7146/aul.455.c200, Online; accessed Oct. 15. 2023.

Jyrki Oraskari, Seppo Törmä, Mehmet Yalcinkaya, Vishal Singh, and Jakob Beetz. Sharing linked building data in a peer-to-peer network: ifcowl meets interplanetary file system. *Journal of Computing in Civil Engineering. 1801 Alexander Bell Drive Reston, VA 20191-4400*, 38(1): 04023037, 2024. doi: 10.1061/JCCEE5.CPENG-5381. URL https://ascelibrary.org/doi/abs/10.1061/JCCEE5.CPENG-5381. published online on September 28, 2023.

Pasi Paasiala, Juha Laukala, Lassi Liflander, Klaus Linhard, Erik Pijnenburg, Léon van Berlo, and Jon Anders Sollien. Bcf-xml: File-based implementation of bim collaboration format. https://github.com/buildingSMART/BCF-XML/, 2019. buildingSMART, GitHub repository, Commit: d67e2a16425f4740182084bba4c937b1b36198c1, Online; accessed Oct. 15. 2023.

Pieter Pauwels. Supporting decision-making in

the building life-cycle using linked building data. *Buildings*, 4(3):549-579, 2014a. ISSN 2075-5309. doi: 10.3390/buildings4030549. URL https://www.mdpi.com/2075-5309/4/3/549. Online; accessed Oct. 15. 2023.

Pieter Pauwels. Supporting decision-making in the building life-cycle using linked building data. *Buildings*, 4(3):549-579, 2014b. Multidisciplinary Digital Publishing Institute.

Pieter Pauwels and Ekaterina Petrova. *Information in Construction*. Eindhoven University of Technology, Ghent, belgium, 2020. URL https://pure.tue.nl/ws/portalfiles/portal/167337913/InformationInConstruction_PauwelsPetrova.pdf. Online; accessed Oct. 15. 2023.

Pieter Pauwels and Walter Terkaj. Express to owl for construction industry: Towards a recommendable and usable ifcowl ontology. *Automation in Construction*, 63:100-133, 2016. Elsevier.

Pieter Pauwels and Léon van Berlo. jyrkioraskari/ifcowl: ifcowl doi reference, June 2019. URL https://doi.org/10.5281/zenodo.3240436.

Pieter Pauwels, Seppo Törmä, Jakob Beetz, Matthias Weise, and Thomas Liebich. Linked data in architecture and construction. *Special Issue Automation in Construction, erscheint Juni*, 2015.

Pieter Pauwels, Jyrki Oraskari, and Lewis John McGibbney. Ifctordf, June 2019. URL https://github.com/pipauwel/IFCtoRDF. (Accessed on Sep. 05. 2023).

Richard Chief Petrie. Openbim definition, Dec 2020. URL https://www.buildingsmart.org/about/openbim/openbim-definition/. Online; accessed Oct. 15. 2023.

Niko P Popitsch and Bernhard Haslhofer. Dsnotify: handling broken links in the web of data. In *Proceedings of the 19th international conference on World wide web*, pages 761-770. Association for Computing Machinery, 2010.

Michael J Pratt. Introduction to iso 10303 - the step standard for product data exchange. *Journal of Computing and Information Science in Engineering*, 1(1):102-103, 2001. American Society of Mechanical Engineers.

Jorulv Rangnes. The buildingsmart international virtual summit 26 october 6 november 2020: The nature of interoperability. https://www.buildingsmart.org/the-buildingsmart-//international-virtual-summit-26-october-6-november-2020, 2020. Accessed: 2022-11-02.

Mads Holten Rasmussen, Maxime Lefrançois, Georg Ferdinand Schneider, and Pieter Pauwels. Bot: the building topology ontology of the w3c linked building data group. *Semantic Web*, 12 (1):143-161, 2021. http://doi.org/10.3233/SW-200385,IOS Press.

Jerome H. Saltzer and Michael D. Schroeder. The protection of information in computer systems. *Proceedings of the IEEE*, 63(9):1278-1308, 1975.

Spyridon Samonas and David Coss. The cia strikes back: Redefining confidentiality, integrity and availability in security. *Journal of Information System Security*, 10(3), 2014.

Oliver Schulz, Jyrki Oraskari, and Jakob Beetz. bcfowl: A BIM collaboration ontology. In María Poveda-Villalón and Pieter Pauwels, editors, *Proceedings of the 9th Linked Data in Architecture and Construction Workshop, Luxembourg, Luxembourg, October 11-13, 2021*, volume 3081 of *CEUR Workshop Proceedings*, pages 142-153. CEUR-WS.org, 2021. URL http://ceur-ws.org/Vol-3081/12paper.pdf. Online; accessed Oct. 15. 2023.

Oliver Schulz, Jyrki Oraskari, and Jakob Beetz. Lessons learned from designing and using bcfowl. In *Proceedings of the 11th Linked Data in Architecture and Construction Workshop*, 2023. URL https://linkedbuildingdata.net/ldac2023/files/papers/papers/LDAC2023_paper_4665.pdf. 11th International Workshop on Linked Data in Architecture and Construction, LDAC; Conference date: 15 - 16 June 2023, Online; accessed Oct. 15. 2023.

Madhumitha Senthilvel, Jyrki Oraskari, and Jakob Beetz. Common data environments for the information container for linked document delivery. *Proceedings of the 8th Linked Data in Architecture and Construction Workshop - LDAC2020. CEUR-WS Team, Redaktion Sun SITE, Informatik V, RWTH Aachen, Germany*, 2636: 132-145, June 2020. ISSN 1613-0073. URL https://ceur-ws.org/Vol-2636/10paper.pdf. International Workshop on Linked Data in Architecture and Construction, LDAC; Conference date: 17-06-2020 Through 19-06-2020, Online; accessed Oct. 20. 2023.

Madhumitha Senthilvel, Jyrki Oraskari, and Jakob Beetz. Implementing information container for linked document delivery (icdd) as a micro-service. In *EG-ICE 2021 Proceedings*, page 6675. Universitätsverlag der TU Berlin, ISBN 978-3-7983-3212-6, August 2021. doi: 10.14279/depositonce-12021. URL https://berlin-2021.eg-ice.org/. International Workshop on Intelligent Computing in Engineering, EG-ICE ; Conference date: 30-06-2021 Through 02-07-2021, Online; accessed Oct. 15. 2023.

Michael Sperberg-McQueen, Tim Bray, Eve Maler, Jean Paoli, and François Yergeau. Extensible markup language (XML) 1.0 (fifth edition). W3C recommendation, W3C, November 2008. URL http://www.w3.org/TR/2008/REC-xml-20081126/. Online; accessed Oct. 15. 2023.

Branislav Sredojev, Dragan Samardzija, and Dragan

Posarac. Webrtc technology overview and signaling solution design and implementation. In *2015 38th international convention on information and communication technology, electronics and microelectronics (MIPRO)*, pages 1006-1009. IEEE, 2015.

Daniel Stenberg. Http/3 explained - the book, July 2019. URL https://daniel.haxx.se/http3-explained/. Accessed on Jul. 07. 2019.

W. Terkaj and P. Pauwels. Ifcowl ontology file for ifc4, 2014. URL https://standards.buildingsmart.org/IFC/DEV/IFC4/ADD2_TC1/OWL/index.html. Online; accessed Oct. 15. 2023.

The Apache Software Foundation. Reasoners and rule engines: Jena inference support, 2022. URL https://jena.apache.org/documentation/inference/. [Accessed 25-Dec-2022].

The International Organization for Standardization. Iso 16739-1: 2018: Industry foundation classes (ifc) for data sharing in the construction and facility management industries-part 1: data schema, 2018.

The Pennsylvania State University. Bim uses within the bim project execution plan. https://bim.psu.edu/uses/, 2013. [Accessed 23-Dec-2022].

S. Törmä. Web of building dataintegrating ifc with the web of data. *eWork and eBusiness in Architecture, Engineering and Construction: ECPPM 2014*, page 141, 2014. CRC Press.

Seppo Torma. Semantic linking of building information models. In *Semantic Computing (ICSC), 2013 IEEE Seventh International Conference on*, pages 412-419. IEEE, 2013.

Seppo Törmä, Jyrki Oraskari, and Nam Vu Hoang. Distributed transactional building information management. In *Proceedings of the 1st*

International Workshop on Linked Dat a in Architecture and Construction (LDAC 2012), Ghent, Belgium, pages 9-11, 2012.

Herbert Van de Sompel, Michael L Nelson, Robert Sanderson, Lyudmila L Balakireva, Scott Ainsworth, and Harihar Shankar. Memento: Time travel for the web. *arXiv preprint arXiv:0911.1112*, 2009.

Rebekka Volk, Julian Stengel, and Frank Schultmann. Building information modeling (bim) for existing buildingsliterature review and future needs. *Automation in construction*, 38: 109-127, 2014. Elsevier.

Anna Wagner, Mathias Bonduel, Pieter Pauwels, and Rüppel Uwe. Relating geometry descriptions to its derivatives on the web. In *2019 European Conference on Computing in Construction*, pages 304-313. European Council on Computing in Construction (EC3), 2019.

WEF. Shaping the Future of Construction: A Breakthrough in Mindset and Technology. Industry agenda, World Economic Forum, Geneva, May 2016.

Jeroen Werbrouck, Pieter Pauwels, Jakob Beetz, and Erik Mannens. Mapping federated aec projects to industry standards using dynamic views. In *Proceedings of the 10th Linked Data in Architecture and Construction Workshop co-located with 19th European Semantic Web Conference (ESWC 2022)*. CEUR-WS, 2022.

Jeroen Werbrouck, Oliver Schulz, Jyrki Oraskari, Erik Mannens, Pieter Pauwels, and Jakob Beetz. A generic framework for federated cdes applied to issue management. *Advanced Engineering Informatics. Elsevier, Amsterdam, Netherlands*, 58:102136, 2023. ISSN 1474-0346. doi: https://doi.org/10.1016/j.aei.2023.102136. URL https://www.sciencedirect.com/science/article/pii/S1474034623002641.

Mark D Wilkinson, Michel Dumontier, IJsbrand Jan Aalbersberg, Gabrielle Appleton, Myles Axton, Arie Baak, Niklas Blomberg, Jan-Willem Boiten, Luiz Bonino da Silva Santos, Philip E Bourne, et al. The fair guiding principles for scientific data management and stewardship. *Scientific data*, 3(1):1-9, 2016. https://doi.org/10.1038/sdata.2016.18, Nature Publishing Group.

Index

Abox 26, 34, 111
ACL 234
AEC 76
AECOO 77
AlgSign 347

BCF 230, 249, 330
BCF API 233
BCF XML 126
bcfOWL 117, 153, 176
bcfXML 97
Big BIM 9
BIM ..... *see* The Building Information model
BIMDO 78
BIMSO 78
BIMSPARQL 255
BLIS 258
BLIS-SABLE 258
bSDD ..... *see* buildingSMART Data Dictionary, 158
bsDD 306
buildingSMART Data Dictionary 158
buildinSMART 3

C14N 393
canonized form 362
CDE ..... *see* Common Data Environmen, 232, 250
CIA 309
CityGML 301
COBie 64, 301

COBieOWL 78
Common Data Environmen 185
ConSolid 260, 264

DAC 312
DIN SPEC 91391 249
DIN SPEC 91391-2 . . . . . 191, 194, 212
DOGONT 78
DRUMBEAT 302
DSHT 391

FMI 193

gbXML 53
GUID 330, 361, 392

HTTP/3 381

ICDD *see* Information Container for linked Document Delivery, 191, 213, 417
IDM 257
IDP 233
IFC 301
IFC EXPRESS 346
IFC GUID 428
IFC-SPF 52, 333
IFC-SPFF 33
IFC/RDF 361
IFC2x3 36
ifcOWL 23, 34, 387, 416
ifcRDF 303
IFCtoLBD 26, 39
IFCtoRDF 26, 83, 393
IfcWoD 77, 124
ifcXML 33
Information Container for linked Document Delivery 208
InfraGML 301
IPFS 387, 413
IPNS 395
IPR 312
IRI 361
ISO 10303-21 52
ISO 12006-3 64
ISO 16739:2013 420
ISO 19650 . . . . . 186, 248
ISO 19650 Stage 3 186
ISO 19650-1:2018 185
ISO 2013 301
ISO 21597 193
ISO 29481-1:2010 257

ISO/PAS 16739 11
ISO/TC 59/SC 13 213
isomorphism 362

Kademlia 391

LBD . . . . . *see* Linked Building Data
LBD-CG 124
LCD 252
LD . . . . . *see* Linked Data
LDP 191, 240, 253
Linked Building Data 13
Linked Data 12
Little BIM 9

MagiCloud 306
Merkle nodes 394
Merkle tree 419
microservice 213
Minimum Self-contained Graph 363
MSG 363
MVD 71
mvdXML 77, 257

OAuth 2.0 311
OIDC 253
Omniclass 57
Open BIM 10
OpenAPI 195
OpenCDE 191, 212
OpenCDE API 241
OpenCDE-API 194, 213
OpenID Connect 310, 311
OWL 77

PAS1192:2003 212
PROPS 36
pset 38

RDF C14N 421
RDF path 152
RDF-star 150
RDF/N3 369
RDFC14Ner 362, 369
RESTful API 249
RESTful services 310

SHACL 74, 77, 79
simpleBIM 34
SOLID 195

SPARQL 34, 124, 152
SPARQL endpoint 418
SPARQLES 418
SSoI 248, 256
stopwords 99
SVD 99

TF-IDF 99
The Building Information model 8
the Building Topology Ontology (BOT) 16
The Design Science Research (DSR) method 4
the FAIR data principles 261
the microservices architecture 258
the QUIC protocol 381
topodiff 369

URI 252

VSM 99

W3C 16, 23
WAC 312
Web of Building Data 300
WebID 235, 253, 311
WebID-OIDC 266
WebID-TLS 311
WoBD 300, 312

X.509 311
XSD schema 128

Typeset November 8, 2024