AEC Digital Twin Data - Why structure matters

André Borrmann[1[0000-0003-2088-7254]], Jonas Schlenger[1], Nicolas Bus[2], Rafael Sacks[3[0000-0001-9427-5053]]

[1] Chair of Computational Modeling and Simulation, Technical University of Munich, Germany
[2] Centre Scientifique et Technique du Bâtiment, Sophia-Antipolis, France
[3]Technion – Israel Institute of Technology, Haifa, Israel
andre.borrmann@tum.de

Abstract. With the increasing adoption of the Digital Twin concept in the construction industry in the operations and maintenance phase, researchers and practitioners are increasingly seeking suitable technological solutions for the design and construction phases. While it is widely accepted that the required platforms hosting the digital twin must be cloud-based to fulfill the requirements of ubiquitous accessibility and centralized consistency, questions regarding the need for data schema remain. Some academics argue that a structure-free organization of data is suitable for realizing digital twins and the data streams from and to the respective platform. Hands-on experience in the BIM2TWIN project supports a counter argument, i.e., that structure-free data is insufficient for most use cases around AEC Digital Twins. The sheer information complexity of construction projects requires well-defined data structures enabling unambiguous and errorless interpretation. This becomes apparent when reflecting on the well-established concept of the data-information-knowledge pyramid describing that raw data must be processed into understandable and meaningful high-level information for human decision makers, subsequently providing the basis for cross-project domain knowledge. Based on this observation, we highlight that object-oriented modeling is a widely recognized information modeling technique that facilitates the structuring of complex domain information. We compare it with ontology-based model concepts that provide a similar, yet more abstract means for information modeling.

Keywords: Digital Twin, Data Model, Information Model, Ontology, Object-oriented modeling

1 Introduction

The term Digital Twin gains more and more popularity in the AEC sector. Originally coined in the manufacturing industry (Grieves & Vickers 2017, Kritzinger et al. 2018), it describes a continuously updated digital representation of a real-world entity. In the AEC sector, the physical entity being twinned is usually a built facility, ranging from buildings over industrial facilities to bridges and tunnels (Boje et al. 2020, Sacks et al. 2020, Mafipour et al. 2022).

While many different interpretations of the quite generic term “Digital Twin” exist, most have in common that a) some kind of sensor(s) is applied to capture the current condition of the physical entity and update its digital replica correspondingly, and b) the digital twin is used to test the expected outcomes of possible control interventions, which can then be applied directly to the physical twin.

Disagreement exists, however, when it comes to the data structures used to represent digital twins. While some researchers believe that un-structured or low-structured data is sufficient for representing digital twins of built facilities (Aragao & El-Diraby 2019, Aragao & El-Diraby 2020, El-Diraby 2021), this paper provides an argumentation that a digital twin is a natural evolution of a digital model and as such should be based on the well-established principles of object-oriented data modeling.

To support our argument, we refer to the ongoing research project BIM2TWIN which aims to provide comprehensive DT representation of constructions projects combining the process and the product view (Schlenger et al. 2022). We show how a well-structured data model helps to reduce complexity and allows to perform the high-level analysis tasks required for decision making.

2 The BIM2TWIN project

The BIM2TWIN project,[1] funded by the European Commission in the framework of the Horizon programme, aims at developing concepts and technologies for creating and maintaining digital twins of construction projects. As such it focuses on providing a digital representation of both, the constructed facility as well as the processes that are required to erect it. To separate design intent from actual realization, we distinguish as-designed from as-built in terms of the product description, and as-planned from as-performed in terms of the process description (Sacks et al. 2020).

For the as-designed/as-planned process representation we make use of the well-established 4D BIM technology, combining sophisticated semantic-geometric models with construction schedules on component level. For the as-built and as-performed information, we compile a parallel object schema with abstractions that can accommodate information derived from monitoring technologies, ranging from laser-scan point clouds to mobile construction applications. Both aspects are incorporated in the BIM2TWIN digital twin platform.

[1] https://bim2twin.eu/

In order to allow for a continuous update of the digital twin regarding the as-built facility and the as-performed processes, data is captured from the site by means of a multitude of sensors, including temperature, laser scanners, photo cameras, thermal cameras etc. The low-level data is subsequently processed into higher-level information, such as geometric deviations, surface qualities, task durations, safety hazards, etc., which are finally aggregated into knowledge presented to human decision makers as key performance indicators (KPI), related to quality control, schedule delays, safety issues etc. More background for the underlying philosophy is provided in Section 3.

The B2T platform is a core element of the BIM2TWIN project. It is designed to hold all the as-designed/as-planned as well as the as-built/as-performed information and provides coherent access to both end-users as well as services to the Digital Twin (functions as a single source of truth of the Digital Twin). The B2T platform is built upon the Thing'In platform provided by project partner Orange[2]. Thing'In itself is based on ArangoDB[3] - a multi-model database system supporting three data models (property graphs, JSON documents, key/value) with a unified query language AQL. By being schema-free, the database itself provides a maximum of flexibility, However, for BIM2TWIN, a properly defined data structure (information model) has been specified to achieve reliable interoperability.

In this paper, we discuss the information model that has been designed for the B2T platform and implemented using the property graph mechanism. We use it as example to illustrate our argumentation for well-defined data structures being required for properly handling digital twin representations and enabling decision making.

3 Data – Information – Knowledge

The differentiation of data, information and knowledge is a well-established concept in computer science (Rowley 2007). Ackoff (198) introduced the DIKW pyramid that consist of different layers of abstraction, namely *data, information, knowledge* and *wisdom* (**Fig. 1**). In this work, we focus on the three abstractions of data, information and knowledge because these are expressed in the B2T platform.

[2] https://hellofuture.orange.com/en/thingin-the-things-graph-platform/
[3] https://www.arangodb.com/

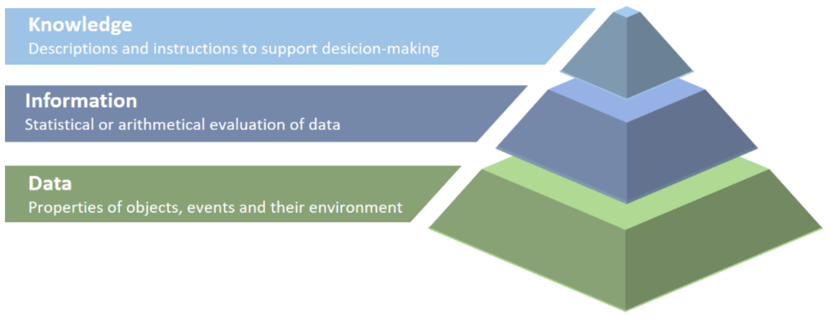

Knowledge
Descriptions and instructions to support desicion-making
Information
Statistical or arithmetical evaluation of data
Data
Properties of objects, events and their environment

Fig. 1. The data-information-knowledge pyramid.

For these layers we provide the following definitions in the B2T context:

Data layer. With “Data” we refer to raw data produced by sensors. Typical examples in the context of BIM2TWIN include images, measurement time-series, point-clouds etc.

Information layer. “Information” has a direct meaning for the end user. Information can either be produced by processing data or manually entered by a human user. A typical example in the context of BIM2TWIN is the completion status of a dedicated building element.

Knowledge layer. The knowledge layer provides a higher layer of abstraction by aggregating and transforming information. It provides the basis for decision making. In the context of BIM2TWIN, this layer allows representation and tracking of Key Performance Indicators (KPI) of the construction project like cycle time, equipment utilization rate, and accident frequency.

4 Data structures and information modeling – a brief history

There is a long history of information modeling in computer science. The developments were dominated by the database sector for a long time; early approaches included the hierarchical model and the network model, however both are limited in terms of expressive power. This changed in 1970 with Codd’s seminal work on the application of formal relational theory onto databases (Codd 1970), which laid the ground for relational databases, which are still the most widespread type of database in use today. A key feature of relational data models is the possibility to connect data records in different relations (tables) through primary and foreign keys, which allows one to model complex information networks with minimal redundancy. A key aspect in this regard is the notion of normalization, which describes a set of rules which, when obeyed, result in a clean, redundancy-free database design.

Although powerful and generic, the relational approach to information modeling has limitations (Robie & Bartels 1994, Damesha, 2015). For example, complex data types and relationships do not exist as such in the relational model, but have to be mimicked by a combination of relations (tables) linked through primary and foreign keys.

Especially, when the data to be queried is distributed across multiple tables, a large number of join operations is needed, and the statements in the query language SQL become long and complex, slowing down the response time significantly. In addition, the important concept of class inheritance can hardly be mapped onto the relational schemata. These limitations became apparent when another very successful paradigm of information modeling became popular in the 1990ies: Object-oriented modeling (OOM) (Jacobsen et al. 1992).

In its core, OOM is based on the concepts of classes that act as templates for concrete instances or objects. Classes have attributes and methods and relations to other classes. An important concept is inheritance, where a subclass inherits all the attributes of its superclass, thus emphasizing modularity and reusability. OO programs essentially instantiate objects of predefined classes, fill their attributes with values and let them interact. Thanks to many features that come along with OO paradigm, such as encapsulation and reusability, OOP has been extremely popular and is implemented in almost all major programming languages, including C++, Java, and Python. Based on these developments, the concept of model-driven architectures for distributed systems was established (Mellor et al. 2002).

While first employed for programming, the OO paradigm has also soon been adopted for more general analysis and design (OOAD) tasks, including the definition of data models for data exchange and persistent storage. In this context, the Unified Modeling Language (UML) was developed for visual definition of object-oriented data models. For the computer-processable form, a number of textual data modeling languages have been developed. These include EXPRESS, which has been employed across the large product modeling standard STEP (ISO 10303) as well as the AEC-focused data model Industry Foundation Classes (IFC) (ISO 16739). Later, XML and XML schema were introduced. Although they evolved from a different background (SGML, HTML), they ended up with very similar features regarding object-oriented data modeling. In consequence, many data models have been encoded in XML schemata, including IFC (denoted ifcXML), but also the GML data models produced by the Open Geospatial Consortium (OGC) (Portele 2012), and many, many more.

A more recent development is the JavaScript Object Notation (JSON) format originally used to serialize JavaScript objects, but increasingly adopted across a wide range of languages and applications in the context of web development (Peng et al. 2011). In comparison with XML, it provides a leaner syntax (improving readability for humans and reducing data footprint) and has the advantage of direct support by programming languages such as Python without the need for sophisticated parsers. The schema language for JSON is JSON schema (Agocs et al. 2018). Again, a number of data models have been encoded into JSON schema, including the IFC data model (Afsari et al. 2017). However, limitations exist, particularly when diverting from a pure tree-like structure and using references to existing objects. This however, is very relevant for more complex graph-like information structures such as building models.

In the early 2000s, semantic web technologies were established with ontologies being the chosen approach towards information modeling. Ontological modeling adopts many concepts of object-oriented design while providing an even richer expressive power allowing a more fine-grained description of real-world entities (Berners-Lee et al. 2001). Here, the Ontology Web Language (OWL) is used for describing the schema level (T Box), while instances are represented by RDF graphs (A Box). SPARQL provides the query language for retrieving information from both class and instance graphs. Semantic web technologies have seen increasing popularity over the last decade, particularly in the context of the Linked Data (LD) philosophy that respects the heterogeneity of the information model landscape while allowing one to flexibly connect corresponding entities across different data models (Bizer et al. 2008). In the AEC context, a number of ontologies have been developed and standardized by W3C (Pauwels et al. 2018). Typically, the semantic extent remains narrower than in conventionally defined data models (Pauwels & Roxin, 2017), but also a mapping of the full IFC model onto OWL exists (Beetz et al. 2009, Pauwels & Terkaj, 2016). It is also worth mentioning that the linked data approach has recently been brought together with the transport format JSON resulting in JSON-LD and providing significant synergies (Bonduel 2021).

Graph databases are a more recent innovation. While some graph databases implement the Semantic Web approach, maintaining RDF graphs with so-called SPARQL endpoints as their interface (Buil-Aranda et al. 2013), others implement the concept of Property Graphs that allow assignment of attributes directly to individual nodes (Junghanns et al. 2016). More recently, semantic-web databases also provide this feature through the RDF-star extension (Hartig & Champin 2021). Graph databases often provide a large degree of flexibility when implementing a schema-free approach where instance graphs can be populated with literally any data. This flexibility, however, comes at a cost, namely the potential risk of incompatibility with applications relying on specific data structures. In consequence, many graph databases now implement a "meta-model" that clearly defines the type of nodes their attributes and the potential connections between nodes, thus mirroring the concept of a schema.

In the field of data science and machine learning, a contrary trend is visible: Instead of using highly structured data following the principles of OOM, simpler formats are often preferred, such as comma-separated-values (CSV) for representing tabular data. The background lies in the fact that most neural network architectures are based on manipulating and transforming matrix and tensor-like structures. Based on these observations, some companies[4] promote the conversion of object-oriented BIM data into these flat data structures for further processing using standard tools of Machine Learning, Big Data Analytics and Business Intelligence. It must be noted, however, that the required de-normalization typically either creates many redundancies or destroys relationships between data records.

Such approaches might be suitable for special, rather simple use cases. In the remainder of the paper, however, we will show that for the management of Digital Twin information of complex construction projects, well-structured and clearly defined data structures are required.

[4] E.g. https://opendatabim.io/

5 Information modeling for AEC digital twins

For the discussion, we clearly distinguish between information models (sometimes also referred to as data models) and file formats. When done properly, the information model is defined independently of any concrete file or transport formats using modeling languages such as UML. OGC, for example, uses the notion of “conceptual models” for information models that are defined in an implementation-independent manner on the one hand, and provides concrete encodings using different data formats such as XML on the other hand.

5.1 Requirements of digital twin data management

In this paper we focus on application of digital twins for construction management. The digital twin in this sense is a replica of the construction project including the as-performed processes as well as the as-built physical objects.

The requirements for the DT data management are derived from this purpose and include the following aspects:

- A product breakdown structure – a hierarchy of objects describing the facility under construction at various levels of granularity
  - differentiation between as-designed and as-built building information
- A work breakdown structure – a hierarchy of objects with interdependencies describing the construction processes to be performed at various levels of granularity
  - differentiation between as-planned and as-performed construction processes
  - notion of quality of the performed processes
- A resource breakdown structure -a hierarchy of objects representing construction resources (equipment, workers, and materials) that serve as input flows for the processes
- A location breakdown structure - a hierarchy of working zone objects that describe the locations where the processes are executed.
- linkages among these four breakdown structures that express the way in which the project is designed and built and how the processes are planned and executed, all at various levels of granularity

From this description, it follows that a graph structure is most suited to modeling the data because any given building model and digital twin is an inherently complex network of objects from each of the four structures and the relationships between them. Relationships may occur between objects of different levels of granularity, and ad-hoc aggregations are needed. The information structure must allow easy access and navigation in the complex network of connected entities by both human end-users and machines and algorithms. Most importantly, the information structure must support computation of the Key Performance Indicators that provide the basis for high-level decision making in the construction project (aka the knowledge layer, see Section 3).

5.2 Monolithic models: Industry Foundation Classes

A more traditional approach towards data modeling is the concept of “monolithic data models” which implies the goal of developing an all-encompassing information model covering all aspects of a domain. One example is the IFC data model, which is described in more detail below. The advantage of this approach is the semantic rigidness and the “inner compatibility” that can only be achieved with one data model. The disadvantage, however, lies in the big size and high complexity of such data models

The Industry Foundation Classes (IFC) developed and maintained by buildingSMART International (2021) is a well-known and internationally used standard for data exchange of construction-related information. The IFC data model implements the paradigm of object-oriented modeling, however with a number of particularities, such as the usage of objectified relationships, the usage of proxy elements and the inclusion of dynamically definable properties. Originally, the IFC schema was directly encoded using EXPRESS, while property set templates were kept outside of the schema and defined by separated XML documents.

Recently, bSI changed the development process and now relies on UML for defining the information model. It supports multiple serializations, with the most common one being the EXPRESS format for the definition of the IFC schema and STEP physical files for instantiation. However, mappings from EXPRESS to XML-Schema (ifcXML), to OWL (Pauwels and Terkaj, 2016) as well as to JSON (Afsari et al. 2017) have been developed, thus opening the richness of the IFC data model to these specific technological worlds.

The IFC data model aims at covering the full range of the built environment, including buildings but also infrastructure assets such as road and railways, while considering all relevant domains, including architecture, HVAC, electrical etc. This naturally results in very big data model with more than 900 classes in the most recent version 4.3.

While very powerful and comprehensive, the IFC data model is regularly criticized for its extent and complexity (Amor et al. 2007). Specific problems arise from the original use of EXPRESS as primary modeling language, which has found only little widespread adoption in the open-source developers community.

Although being a common misunderstanding, the IFC data model is not bound to the usage of files for exchanging information. Instead, the underlying STEP standard foresees the data model being used in databases etc. Parts 23, 24, 27 define bindings for the programming languages C, C++ and Java, respectively.

However, when it comes to concrete solutions for web-based data management of IFC models, there are a few options that natively support the EXPRESS format, among them the Open BIM server and Jotne EDMserver. However, as these are rather bespoke implementations they suffer from limited support and use by the wider community. Mapping to relational databases is possible, but is plagued by the well-known object-relational impedance mismatch (Ireland et al. 2009).

Graph databases on the other hand seem to be a natural option for storing and managing networks of objects as an IFC instance model. Indeed, a commonly proposed option is to use “SPARQL endpoints” to manage and query RDF instance graphs following the ifcOWL ontology (Zhang et al. 2018, Krijnen & Beetz 2018, Guo et al. 2020).

Another option is the usage of property graph databases. The conversion of the IFC model into graph meta-models for management of IFC instances in graph databases is hindered by the particularities mentioned above, in particular the notion of objectified relationships (Borrmann et al. 2018). Future versions of IFC will aim to overcome these deficiencies (van Berlo et al. 2021).

5.3 Linked Data Approach

An approach contrary to the IFC approach of a monolithic data model is taken by the Linked Data Community following the concepts of the Semantic Web and Ontology modeling. Here, typically, much smaller data models are defined that focus on specific aspects of the built environment. In this context, the Building Topology Ontology, Building Product Ontology and the BRICK ontology have been published (Pauwels et al. 2018). A key aspect of this approach is the re-usage of existing ontologies.

In the context of digital twins for construction management, some relevant ontologies are Damage Topology Ontology, Digital Construction Ontologies, Building Element Ontology (Bonduel 2021)] but also the ifcOWL.

As discussed before, RDF data is stored in RDF databases which essentially are databases specialized on RDF data, accessible by means of SPARQL endpoints, providing standardized access to the data through the web.

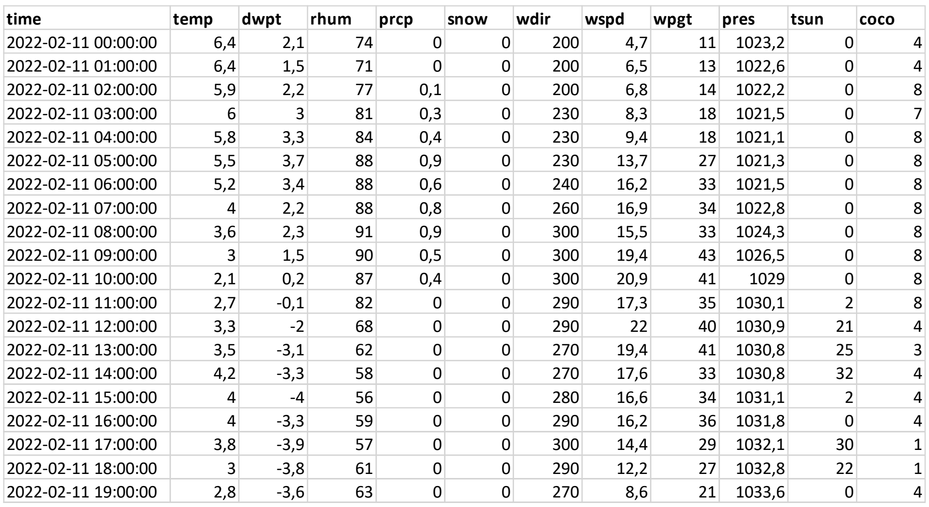
The linked data approach is very relevant for the concept of digital twins as indeed many different aspects have to be covered, which are not available in one comprehensive monolithic data model, but in various ontologies with smaller scopes. A major challenge however lies in the correct linking of the individual ontologies and their objects. Nevertheless, thanks to its expressive power the linked data approach provides both flexibility but also rigidness when it comes to the specification of well-defined data structures.

On the data hosting side, ontologies are increasingly supported not only by dedicated RDF databases, but also by other graph databases. Although conceptual differences exist (Angles et al. 2019), a mapping from RDF graphs onto Property Graphs is possible (Angles 2020). A key aspect, however, is not to use the flexibility of graph databases in terms of node-specific extensions and particularities, but stick to the concept of schemas (here: ontologies or graph meta-models) as this provides the necessary reliability in terms of agreed content for any application accessing the database.

5.4 Flat data approaches for digital twins

A few researchers have proposed a flat data approach for digital twins (El-Diraby 2021, Koch et al. 2021). In this case the data is represented in a mere tabular structure consisting of a large number of data records. Typically, this kind of bulk data is stored and exchanged using the CSV format. The notion of objects does not exist, neither the concept of a data schema describing the meaning of the columns, their data types and units. This allows a high degree of flexibility, however at the cost of interoperability, as the correct interpretation of the received data is the responsibility of the human programmer. Given the vagueness and imprecision involved, this easily leads to misinterpretation.

The flat data approach is appropriate for the data layer of the digital twin, where bulk data is (temporally) stored, processed and analyzed to feed the higher information levels of the DIKW pyramid. Typical examples are time series of temperature and humidity, location protocols of equipment and workers, object detection results of images or even raw point clouds (see **Fig. 2**).

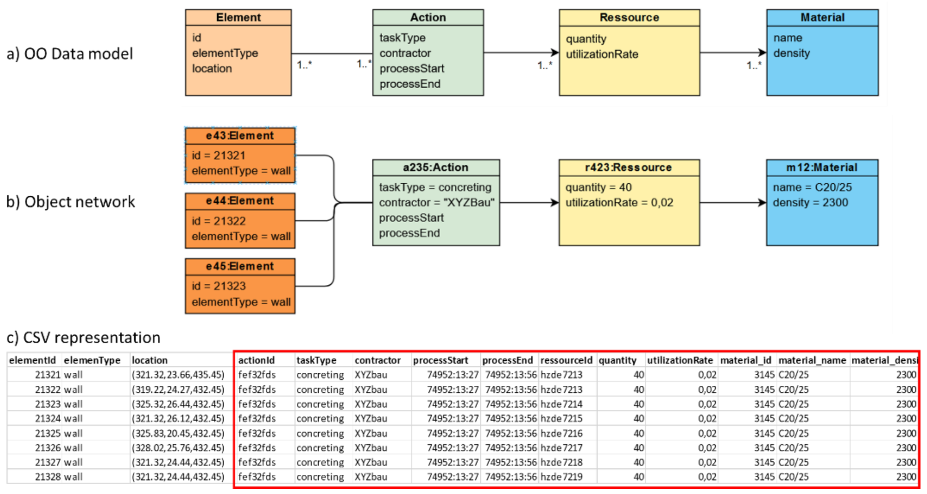


time	temp	dwpt	rhum	prcp	snow	wdir	wspd	wpgt	pres	tsun	coco
2022-02-11 00:00:00	6,4	2,1	74	0	0	200	4,7	11	1023,2	0	4
2022-02-11 01:00:00	6,4	1,5	71	0	0	200	6,5	13	1022,6	0	4
2022-02-11 02:00:00	5,9	2,2	77	0,1	0	200	6,8	14	1022,2	0	8
2022-02-11 03:00:00	6	3	81	0,3	0	230	8,3	18	1021,5	0	7
2022-02-11 04:00:00	5,8	3,3	84	0,4	0	230	9,4	18	1021,1	0	8
2022-02-11 05:00:00	5,5	3,7	88	0,9	0	230	13,7	27	1021,3	0	8
2022-02-11 06:00:00	5,2	3,4	88	0,6	0	240	16,2	33	1021,5	0	8
2022-02-11 07:00:00	4	2,2	88	0,8	0	260	16,9	34	1022,8	0	8
2022-02-11 08:00:00	3,6	2,3	91	0,9	0	300	15,5	33	1024,3	0	8
2022-02-11 09:00:00	3	1,5	90	0,5	0	300	19,4	43	1026,5	0	8
2022-02-11 10:00:00	2,1	0,2	87	0,4	0	300	20,9	41	1029	0	8
2022-02-11 11:00:00	2,7	-0,1	82	0	0	290	17,3	35	1030,1	2	8
2022-02-11 12:00:00	3,3	-2	68	0	0	290	22	40	1030,9	21	4
2022-02-11 13:00:00	3,5	-3,1	62	0	0	270	19,4	41	1030,8	25	3
2022-02-11 14:00:00	4,2	-3,3	58	0	0	270	17,6	33	1030,8	32	4
2022-02-11 15:00:00	4	-4	56	0	0	280	16,6	34	1031,1	2	4
2022-02-11 16:00:00	4	-3,3	59	0	0	290	16,2	36	1031,8	0	4
2022-02-11 17:00:00	3,8	-3,9	57	0	0	300	14,4	29	1032,1	30	1
2022-02-11 18:00:00	3	-3,8	61	0	0	290	12,2	27	1032,8	22	1
2022-02-11 19:00:00	2,8	-3,6	63	0	0	270	8,6	21	1033,6	0	4

Fig. 2. Weather data (from meteostat.net) as a typical example for flat data which can be represented by the CSV format.

However, flat data approaches are not suitable for representing complex information of entire construction projects. The simple concept of a data record cannot replace the notion of an object having properties and associations with other objects. If an information requires a non-basic data type (float, integer, Boolean), a multitude of columns is necessary, thus increasing complexity. Unlike the relational model, the flat data approach typically does not address normalization, leading to frequent redundancies and thus inconsistencies.

This is illustrated by the example shown **Fig. 3** where (a) a simplified class diagram for representing the as-performed processes is depicted along with (b) an UML instance diagram showing the resulting object network when employing OO or graph databases, and (c) the corresponding flat data CSV representation. The lack of higher-level concepts in flat data, in particular object associations, results in excessive repetition of data that remains unchanged. In the shown example, this refers to action, resource and material data that must be repeated for every wall instance. This redundancy results in a massive increase in storage footprint and creates the risk of inconsistencies.



elementId	elemenType	location	actionId	taskType	contractor	processStart	processEnd	ressourceId	quantity	utilizationRate	material_id	material_name	material_densi
21321	wall	(321.32,23.66,435.45)	fef32fds	concreting	XYZbau	74952:13:27	74952:13:56	hzde7213	40	0,02	3145	C20/25	2300
21322	wall	(319.22,24.27,432.45)	fef32fds	concreting	XYZbau	74952:13:27	74952:13:56	hzde7213	40	0,02	3145	C20/25	2300
21323	wall	(325.32,26.44,432.45)	fef32fds	concreting	XYZbau	74952:13:27	74952:13:56	hzde7214	40	0,02	3145	C20/25	2300
21324	wall	(321.32,26.12,432.45)	fef32fds	concreting	XYZbau	74952:13:27	74952:13:56	hzde7215	40	0,02	3145	C20/25	2300
21325	wall	(325.83,20.45,432.45)	fef32fds	concreting	XYZbau	74952:13:27	74952:13:56	hzde7216	40	0,02	3145	C20/25	2300
21326	wall	(328.02,25.76,432.45)	fef32fds	concreting	XYZbau	74952:13:27	74952:13:56	hzde7217	40	0,02	3145	C20/25	2300
21327	wall	(321.32,24.44,432.45)	fef32fds	concreting	XYZbau	74952:13:27	74952:13:56	hzde7218	40	0,02	3145	C20/25	2300
21328	wall	(321.32,24.44,432.45)	fef32fds	concreting	XYZbau	74952:13:27	74952:13:56	hzde7219	40	0,02	3145	C20/25	2300

Fig. 3: The example shows (a) an object-oriented data model, (b) a network of instances as provided by object-oriented approaches and graph databases, and (c) the same data in a flat-data representation using CSV. The red-boxed part resembles repeated data that must be created in CSV and results in redundancy and larger data footprint.

In relational database design, this effect is well known and can be overcome by *database normalization*, resulting in separate tables for *Elements*, *Actions*, *Resources*, and *Materials* in the shown example, to avoid duplication and redundancy (Kent 1983). The authors point out that this example is only a small section of a real DT data model, which typically has much more classes and associations (see next Section), rendering the flat data approach even less suitable.

Indeed, the complex interlinked nature of the information representing a digital twin of construction projects can hardly be reflected by flat tables, but deserves a navigable graph structure. It has been shown that graphs, as well as hierarchical aggregation structures, facilitate the navigation of complex information by both, end-users and application programmers.

6 The BIM2TWIN data model

For the reasons discussed above, the BIM2TWIN project has decided to use the linked data approach for implementing the information layer of the DIKW pyramid. A number of existing ontologies for covering specific aspects (such as the building structure) are re-used and integrated. At the same time, however, it was necessary to define a core ontology that reflects the precise requirements of the Digital Twin of construction projects as listed in Section 5.1.

6.1 Key model characteristics

As a starting point, the BIM2TWIN Core data model was developed as UML diagrams, as shown in Figures 4 and 4. This leaves the model file format agnostic and allows various implementation strategies. One of its main characteristics is the separation of the project intent (Fig. 4) and the project status, representing the current situation on the construction site (Fig. 5). These two sides of the model can be understood as two containers that use a set of classes specific to the model side but also classes that both have in common. Even though the two sides are visualized separately, there are clear connections between them. The intention is to link every node from one side to its corresponding node on the other side. This allows direct comparison of the project plan to its actual realization and results in precise information about the deviation between them.

The Core data model classes can be grouped into four categories. These are the construction processes, the resources that are their input parameters, the working zones where they are executed, and finally, the elements of the building structure, which are their output. All four categories are explained in more detail in the following section.

Processes: The processes are the main aspect of the model and can be found in the center of Figures 2 and 3. We differentiate between three different process levels. The most general level of the processes is the *work package*. It holds information about the used construction method and can be seen as an aggregation of more detailed processes. Every *work package* consists of multiple *activities*, representing the individual construction steps that are part of the *work package*. The *activities* are broken down even further into *tasks*. While an activity can refer to a construction step applied to a group of construction elements, the *task*s have one-to-one relations to the elements. Additionally, p*reconditions* can be connected *to any processes on any of the three levels.* They describe the initial requirements to allow a process to be started. Where the as-planned processes hold details about long-term averaged performance factors dependent on the construction company, the construction method, and the project's particularities (Hofstadler, 2007), the as-performed processes (*construction*, *operation*, and *action*) need to support short-term performance evaluation. Fine-grained insight into performance variation and process disruption allows the development of timely countermeasures to improve the overall construction performance.

Products: The building elements are organized according to the spatial structure of the building. From high-level to low-level, the construction project is broken down into the *site*, one or multiple *buildings*, their *storeys*, and their *spaces*. Depending on their type, the *elements*, like walls, slabs, and columns, can be associated with any of these levels. Since there are no significant differences between the project intent and status regarding the building structure, both model parts use the same set of classes.

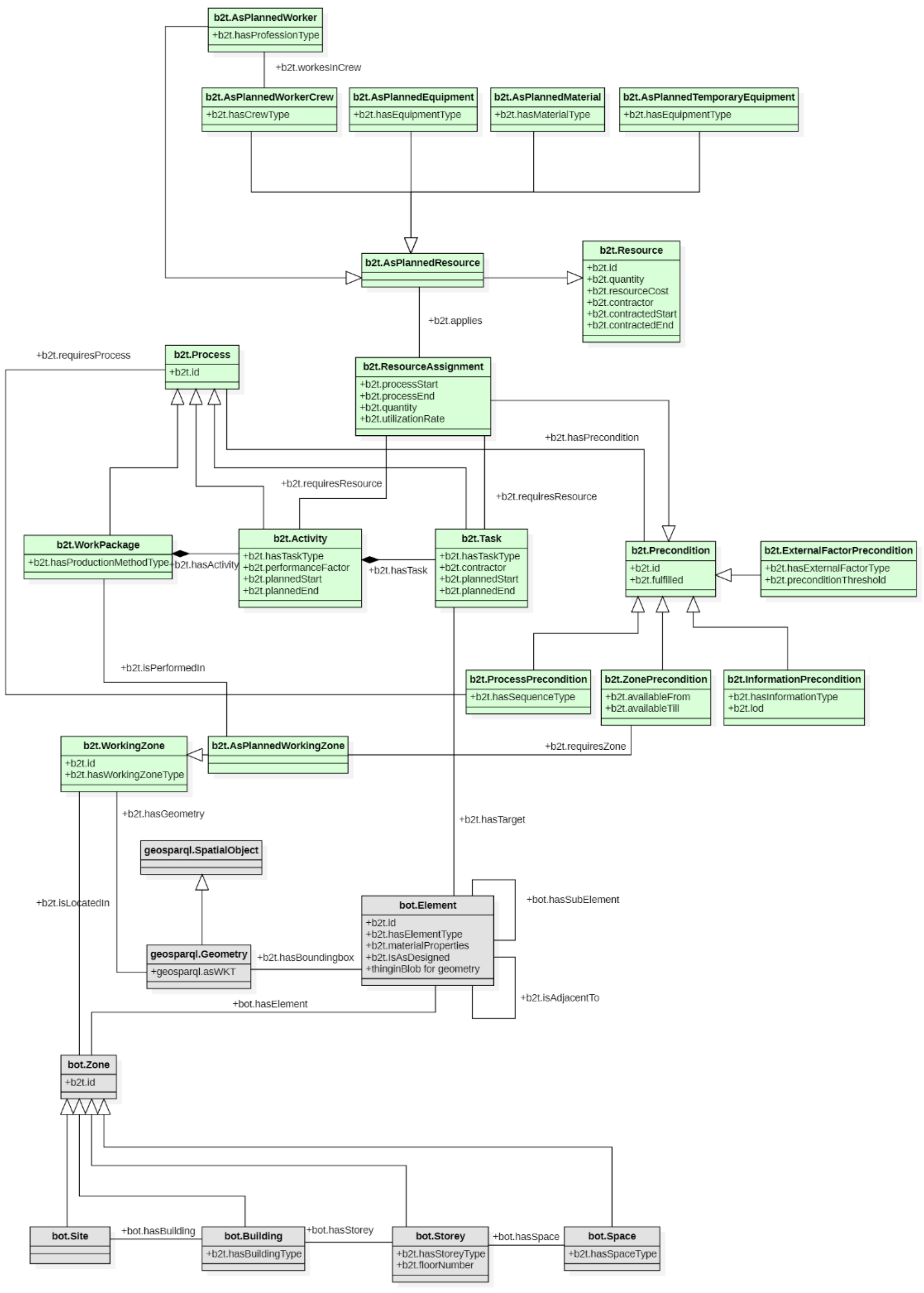


Fig. 4. UML model of the project intent information (as-designed and as-planned).

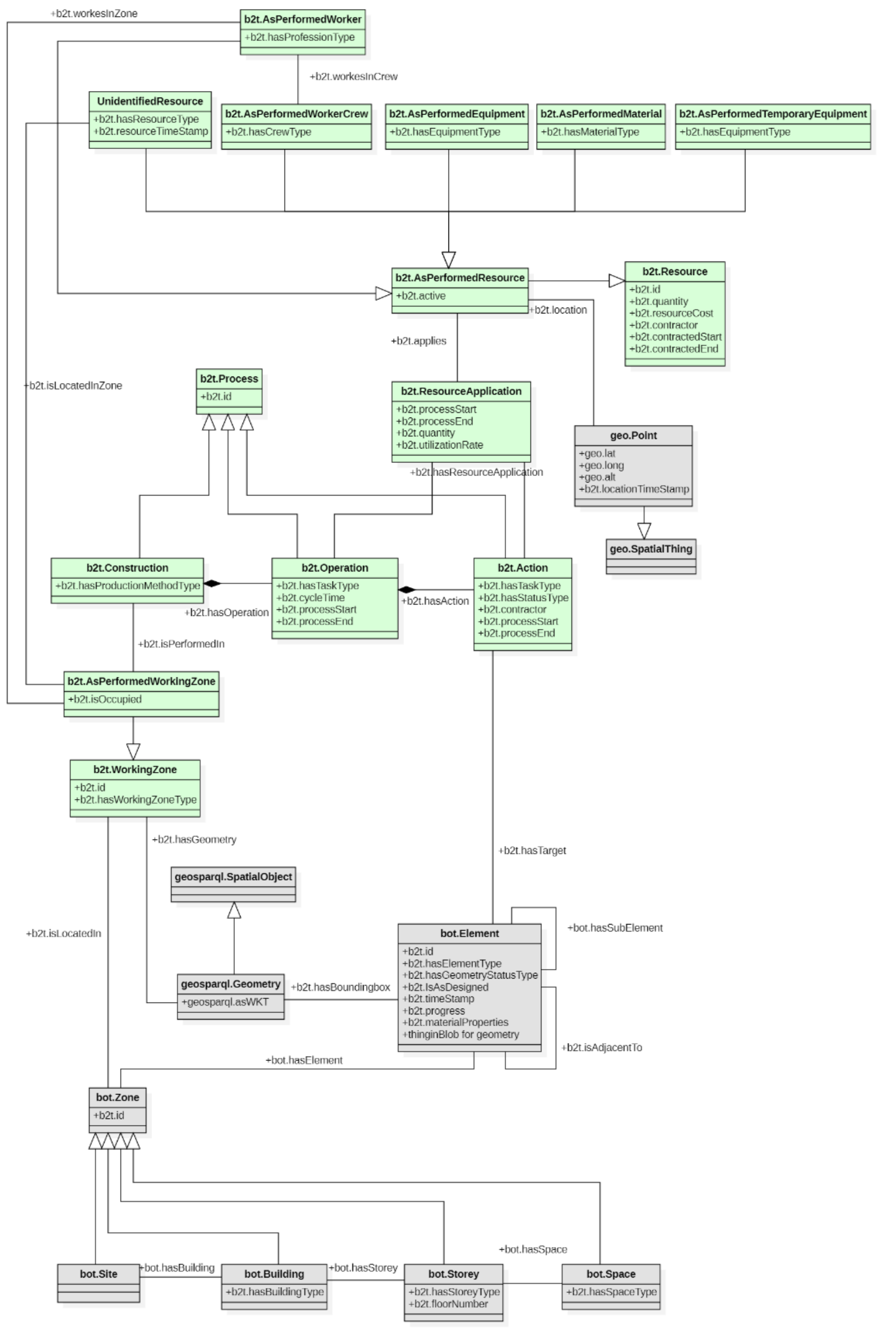


Fig. 5. UML model of the project status information (as-built and as-performed).

Resources: Various types of *resources* are modeled in the Core data model. They describe the input flows of the construction processes, which are essential for successful process execution. The included types of resources are the construction workers and worker crews representing the labor force, *equipment* like heavy machinery and small tools, *materials*, and *temporary equipment* like formwork and guardrails. In Figures 2 and 3, they can be found on the upper half of the diagram. While the *resource* classes are foreseen to model the resources available on the construction site, the *resource assignment* class is used to specify the amount and time frame during which a resource is assigned to a specific process. The *resources* differ between project intent and project status because, during execution, exact location and current activity/ inactivity can be monitored. At the same time, this information is not planned ahead of time.

Zones: Finally, the *zones* enable modeling of the location breakdown structure of the construction project, describing dedicated zones where processes are executed, represented through a direct link between them. Unlike the zones related to the building structure, they do contain geometric information. *Zones* can be equivalent to, e.g., a *storey* or a *space*, but a direct relationship is not always given. Overall, the *zones* are an essential indicator of the construction flow because flow can be judged on the occupation rate of worker crew and flow of materials but also on the occupancy of working locations (Sacks, 2016).

6.2 BIM2TWIN Core Ontology

Based on the UML diagrams above the BIM2TWIN Core Ontology was implemented by translating the diagrams into the corresponding ontology classes, object properties, and data properties. Additionally, only domains and ranges of object and data properties were defined to improve reusability.

The lightweight BOT ontology is reused in the BIM2TWIN Core ontology for the classes related to the building structure. For the data layer of the DIKW-pyramid, SOSA/SSN and QUDT are reused to represent sensor data from the construction site and their units. The data layer is not shown in the UML diagrams above. The status of the BIM2TWIN data model and ontology presented here are the first stable version but will be further refined through testing on dedicated pilot projects. Once thoroughly tested, the ontology will be published online and freely accessible.

6.3 BIM2TWIN platform

To provide access to a large number of applications and services providing and retrieving digital twin data, the described ontology is used as underlying schema of a central platform. In the BIM2TWIN project this platform is formed by the commercial product Thing'In by Orange, which in turn is based on the property graph database Arango DB. The platform allows to upload ontologies in the OWL format and transform its content into a graph meta model. By supporting various rule-checking languages like SHACL and ShEx, data manipulation requests can be checked for compliance with the used ontologies and further ensure the data structure. Instances of the ontology classes are represented by graph nodes with respective properties and edges to related nodes. Access to individual nodes is provided through a dedicated REST API.

7 Conclusion

In this paper, we have discussed the relevance of well-defined data structures for handling the complexity of digital twin information. We use the notion of the data-information-knowledge-wisdom pyramid to clearly distinguish between the different layers of abstraction. We show that simplistic approaches such as flat CSV structures can be suitable for handling bulk data such as temperature time series, but have clear limitations when it comes to the information level. Here, well-defined data structures are required providing clear and unambiguous specifications of the relevant information. Object-oriented modeling has been proven to be the gold standard of information modeling for many years now, providing powerful concepts such as encapsulation, inheritance and associations.

These concepts are also very suitable for representing digital twins of construction projects, which are dominated by a complex network of objects reflecting processes and products on various levels of granularity. We see ontology modeling as a suitable implementation of the object-oriented paradigm and presented the core ontology of the BIM2TWIN platform to underline this statement. In the project, the ontology is mapped to a property graph which is hosted by a dedicated cloud database, allowing fine-grained access for the distributed digital twin ecosystem.

With this paper, the authors hope to contribute to the discussion on suitable data structures for digital twins. We emphasize that in our view, digital twin technologies are a natural evolution of the BIM technologies, as both concepts are based on well-defined data structures. In this sense, many of the elaborated information models laid down in standards such as the Industry Foundation Classes remain absolutely valid and provide a solid foundation to build on. At the same time, however, a more flexible combination with small-scope ontologies is essential for digital twins, as well as a much more fine-grained data access that must replace conventional file-based data exchanges.

Acknowledgements

The research presented in this paper has been partially funded by the European Union’s Horizon 2020 research and innovation programme under grant agreement no. 958398, “BIM2TWIN: Optimal Construction Management & Production Control”, as well as Transregio 277 “Additive Manufacturing in Construction – The Challenge of Large Scale” funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) - project number 414265976 - TRR277.

References

1. Amor, R., Jiang, Y., & Chen, X. (2007). BIM in 2007–are we there yet?. In Proceedings of CIB W78 conference on Bringing ITC knowledge to work, Maribor, Slovenia (pp. 26-29).
2. Ackoff, R. (1989): From Data to Wisdom, Journal of Applied Systems Analysis. 16: 3–9.
3. Agocs, A., & Le Goff, J. M. (2018). A web service based on RESTful API and JSON Schema/JSON Meta Schema to construct knowledge graphs. In *2018 International Conference on Computer, Information and Telecommunication Systems (CITS)* (pp. 1-5). IEEE.
4. Afsari, K., Eastman, C. M., & Castro-Lacouture, D. (2017). JavaScript Object Notation (JSON) data serialization for IFC schema in web-based BIM data exchange. *Automation in Construction*, *77*, 24-51.
5. Angles, R. (2012). A comparison of current graph database models. In 2012 IEEE 28th International Conference on Data Engineering Workshops (pp. 171-177). IEEE.
6. Angles, R., Thakkar, H., & Tomaszuk, D. (2019). RDF and Property Graphs Interoperability: Status and Issues. AMW, 2369.
7. Angles, R., Thakkar, H., & Tomaszuk, D. (2020). Mapping RDF databases to property graph databases. IEEE Access, 8, 86091-86110.
8. Aragao, R. R., & El-Diraby, T. E. (2019). Using network analytics to capture knowledge: three cases in collaborative energy-oriented planning for oil and gas facilities. *Journal of Cleaner Production*, *209*, 1429-1444.
9. Aragao, R., & El-Diraby, T. E. (2021). Network analytics and social BIM for managing project unstructured data. *Automation in Construction*, *122*, 103512.
10. Berners-Lee, T., Hendler, J., & Lassila, O. (2001). The semantic web. *Scientific american*, *284*(5), 34-43.
11. Beetz, J., Van Leeuwen, J., & De Vries, B. (2009). IfcOWL: A case of transforming EXPRESS schemas into ontologies. Ai Edam, 23(1), 89-101.
12. Bonduel, M. (2021): A Framework for a Linked Data-based Heritage BIM, PhD Thesis, KU Leuven, Belgium
13. Bizer, C., Heath, T., & Berners-Lee, T. (2008). Linked data: Principles and state of the art. In World wide web conference (Vol. 1, p. 40).
14. Boje, C., Guerriero, A., Kubicki, S., & Rezgui, Y. (2020). Towards a semantic Construction Digital Twin: Directions for future research. Automation in Construction, 114, 103179.
15. van Berlo, L., Tauscher, H., Liebich, T., van Kranenburg, A., & Paasiala, P. (2021). Future of the Industry Foundation Classes: towards IFC 5. In Proc. of the Conference CIB W78 2021, pp. 11-15
16. Borrmann, A.; Beetz, J.; Koch, C.; Liebich, T.; Muhic, S. (2018): Industry Foundation Classes: A Standardized Data Model for the Vendor-Neutral Exchange of Digital Building Models, In: Borrmann, A.; König, M.; Koch, C.; Beetz, J. (Eds): Building Information Modeling, Springer, 2018
17. Brilakis, I. and Pan, Y. and Borrmann, A. and Mayer, H. and Rhein, F. and Vos, C. and Pettinato, E. and Wagner, S. (2020): Built Environment Digital Twinning, Report of the International Workshop of Built Environment Digital Twinning, Institute for Advanced Study, Technische Universität München
18. Buil-Aranda, C., Hogan, A., Umbrich, J., & Vandenbussche, P. Y. (2013). SPARQL web-querying infrastructure: ready for action?. In *International Semantic Web Conference* (pp. 277-293). Springer, Berlin, Heidelberg.
19. Codd, E. F. (1970). A relational model of data for large shared data banks. *Communications of the ACM*, *13*(6), 377-387.

20. Damesha, H. S. (2015). Object Oriented Database Management Systems-Concepts, Advantages, Limitations and Comparative Study with Relational Database Management Systems. Global Journal of Computer Science and Technology 15 (3) 11-18
21. El-Diraby, T.E. (2021): Can IFC mentality be the basis of digital twins? NO. Keynote presentation, CIB-W78 conference, Luxembourg
22. Grieves, M., & Vickers, J. (2017). Digital twin: Mitigating unpredictable, undesirable emergent behavior in complex systems. In *Transdisciplinary perspectives on complex systems* (pp. 85-113). Springer, Cham.
23. Guo, D., Onstein, E., & Rosa, A. D. L. (2020). An approach of automatic SPARQL generation for BIM data extraction. Applied Sciences, 10(24), 8794.
24. Hartig, O., & Champin, P. A. (2021): Metadata for RDF Statements: The RDF-star Approach. In Lotico.
25. Kritzinger, W., Karner, M., Traar, G., Henjes, J., & Sihn, W. (2018). Digital Twin in manufacturing: A categorical literature review and classification. *IFAC-PapersOnLine*, *51*(11), 1016-1022.
26. Ireland, C., Bowers, D., Newton, M., & Waugh, K. (2009). A classification of object-relational impedance mismatch. In *2009 First International Confernce on Advances in Databases, Knowledge, and Data Applications* (pp. 36-43). IEEE.
27. Jacobsen, I.; Magnus C.; Patrik J.; Gunnar O. (1992). Object Oriented Software Engineering. Addison-Wesley ACM Press
28. Junghanns, M., Petermann, A., Teichmann, N., Gómez, K., & Rahm, E. (2016). Analyzing extended property graphs with Apache Flink. In Proceedings of the 1st ACM SIGMOD Workshop on Network Data Analytics (pp. 1-8)
29. Koch, J., Lotzing, G., Gomse, M., & Schüppstuhl, T. (2021). Application of Multi-Model Databases in Digital Twins Using the Example of a Quality Assurance Process. Lecture Notes in Mechanical Engineering, 364–371
30. Kent, W. (1983) *A Simple Guide to Five Normal Forms in Relational Database Theory*, Communications of the ACM, vol. 26, pp. 120–125
31. Krijnen, T., & Beetz, J. (2018). A SPARQL query engine for binary-formatted IFC building models. Automation in Construction, 95, 46-63.
32. Mafipour, M. S.; Vilgertshofer, S.; Borrmann, A. (2022): Digital twinning of bridges from point cloud data by deep learning and parametric models, In: Proc. of European Conference on Product and Process Modeling 2022, Trondheim, Norway, 2022
33. Mellor, S. J., Scott, K., Uhl, A., & Weise, D. (2002). Model-driven architecture. In: International Conference on Object-Oriented Information Systems (pp. 290-297). Springer, Berlin, Heidelberg.
34. Pauwels, P., & Terkaj, W. (2016). EXPRESS to OWL for construction industry: Towards a recommendable and usable ifcOWL ontology. *Automation in construction*, *63*, 100-133.
35. Pauwels, P., & Roxin, A. (2017). SimpleBIM: From full ifcOWL graphs to simplified building graphs. In eWork and eBusiness in Architecture, Engineering and Construction (pp. 11-18). CRC Press.
36. Pauwels, P., McGlinn, K., Törmä, S., & Beetz, J. (2018). Linked data. In: A. Borrmann, M. König, C. Koch, J. Beetz (Eds): Building information modeling (pp. 181-197). Springer, Cham.
37. Peng, D., Cao, L., & Xu, W. (2011). Using JSON for data exchanging in web service applications. *Journal of Computational Information Systems*, *7*(16), 5883-5890.
38. Portele, C. (2012). OGC Geography Markup Language (GML)–Extended schemas and encoding rules. In *Open Geospatial Consortium Inc*.

39. Robie, J., & Bartels, D. (1994). A comparison between relational and object oriented databases for object oriented application development. *POET Software Corporation*, 800-950.
40. Rowley, Jennifer (2007): The wisdom hierarchy: representations of the DIKW hierarchy, Journal of Information and Communication Science. 33 (2): 163–180
41. Sacks, R., Brilakis, I., Pikas, E., Xie, H., Girolami, M. (2020). ‘Construction with Digital Twin Information Systems’, Data-Centric Engineering, Vol. 1, pp. 1-26.
42. Schlenger, J., Yeung, T., Vilgertshofer, S., Martinez, J., Sacks, R., & Borrmann, A. (2022): A Comprehensive Data Schema for Digital Twin Construction. 29th International Workshop on Intelligent Computing in Engineering (EG-ICE), Aarhus, Denmark
43. Zhang, C., Beetz, J., & de Vries, B. (2018). BimSPARQL: Domain-specific functional SPARQL extensions for querying RDF building data. Semantic Web, 9(6), 829-855.