2025 European Conference on Computing in Construction

CIB W78 Conference on IT in Construction

Porto, Portugal
July 14-17, 2025

# INTRODUCTION OF THE ONTOLOGY FOR CHRONOLOGICAL CONSTRUCTION PROCESSES (OCCP)

Albrecht Vaatz[1,2], Al-Hakam Hamdan[3], Martin Wogan[1,4], Nidhal Al-Sadoon[1], Karsten Menzel[1]
[1]Technical University, Dresden, Germany
[2]DB Bahnbau Gruppe GmbH, Berlin, Germany
[3]A+S Consult GmbH, Dresden, Germany
[4]B/M Consult GmbH, Braunschweig, Germany

## Abstract

The Ontology for Chronological Construction Processes (OCCP) provides a semantic foundation for managing time-related information in the AECO sector. Based on OWL-Time (Cox et al., 2006), OCCP introduces a lifecycle-oriented concept for BIM workflows, enhancing precise timestamp management with phases, cycles, milestones, and transitions to enable traceability throughout an asset's lifetime. This paper describes the concept and composition of OCCP, its SHACL-based validation rules, and provides an example to demonstrate its practical application.

## Introduction

The AECO sector faces growing challenges in managing complex project information effectively. While Building Information Modeling (BIM) provides a structured approach, current workflows lack robust mechanisms for integrating and utilizing time-sensitive data. Existing standards like Industry Foundation Classes (IFC) (ISO 16739-1) and Information Container for Linked Document Delivery (ICDD) (ISO 21597-1) enable spatial and semantic interoperability, but their support for temporal data remains fragmented. This results in disconnected records of planned and actual timelines, leading to errors, inefficiencies, and reduced traceability (Autodesk, 2018). Current approaches, such as XML and relational databases, provide only basic timestamp storage, failing to implement or capture semantic relationships between lifecycle stages. They also struggle with scalability, consistency, and integration within BIM and multimodel frameworks, issues that graph-based ontologies address more effectively (Berners-Lee et al., 2001). The W3C Time ontology (Cox et al., 2006) provides a foundational model for representing time-related information, but lacks domain-specific constructs needed for chronological consistency, overlapping phases, iterative cycles, and project-specific adaptations. The OCCP introduces a structured approach for managing construction lifecycles by defining phases, cycles, transitions, and instants, with SHACL shapes enabling validation, logical sequencing, and data integrity. OCCP enables the structured representation of lifecycle-relevant states and their temporal dependencies within BIM workflows and data containers, including ICDD, to support long-term consistency and traceability in chronological models (cMod) by providing a chronological framework applicable to both general datasets and ICDD containers. This paper presents its core structure, logic, and applications, demonstrating its potential to transform time data management in the AECO sector.

## State of the art

The AECO sector increasingly relies on digital tools and standards to manage complex building projects. IFC, developed by buildingSMART International, has become the de facto standard for BIM interoperability, supporting geometric, semantic, and topological data exchange across disciplines (Sacks et al., 2018; buildingSMART International, 2023). While IFC provides detailed lifecycle management, it lacks robust support for dynamic temporal and process-oriented data, despite features like IfcOwnerHistory for versioning and IfcWorkPlan for scheduling, which remain underutilized and contextually limited (Pauwels et al., 2017). OCCP addresses this by structuring lifecycle phases semantically.

Multimodels emerged to address the limitations of single, centralized models (Scherer & Schapke, 2011). These integrate multiple specialized data sources (e.g., cost, schedule, energy models) into a unified project representation (Borrmann et al., 2018; Gilo et al., 2010). By linking different data layers, multimodel containers enhance information accessibility (Xie et al., 2024), but their lack of semantic integration and validation mechanisms can lead to inconsistencies and reduced reliability (Singh et al., 2011). Similarly, while ICDD (ISO 21597-1) improves traceability and version control for static document delivery, it does not inherently support semantic reasoning or structured chronological data management (Pauwels et al., 2016). OCCP complements this by enabling chronological structuring within and beyond ICDD containers.

Ontologies, expressed using the Resource Description Framework (RDF) (Lassila and Swick, 1999), provide a powerful approach for structuring, validating, and reasoning BIM data (Beetz et al., 2009). They improve

data consistency and interoperability, making them suitable for semantic integration. The OWL-Time Ontology is widely used for temporal data representation, supporting instants, intervals, and relationships such as *time:Before* and *time:After*. It enables integration with IfcOWL, which is an OWL representation of the IFC schema, and ICDD for semantic temporal modeling (Pauwels et al., 2016) and facilitates advanced temporal queries, such as detecting overlapping project phases (Car et al., 2024). The OWL-Time Ontology provides a foundation for temporal modeling, but lacks domain-specific constructs for AECO workflows, such as phases, iterative cycles, or distinctions between planned and actual timestamps (Volk et al., 2014). It also does not inherently validate chronological consistency, complicating the detection of conflicts like overlapping timelines (Karlapudi et al., 2021). While DiCon (Törmä, 2022) models time-based events with Instant and Interval classes for construction workflows, OCCP introduces a unique combination of flexible phase and cycle definitions. By leveraging the Shapes Constraint Language (SHACL) (Knublauch et al., 2017; Debruyne et al., 2020), OCCP enables validation of RDF-based temporal structures, enforcing logical constraints like the correct ordering of phases, cycles, and related instants in AECO contexts.

However, current IFC implementations still lack structured temporal representations, leading to fragmented data across multiple tools (Volk et al., 2014; Pauwels, 2017). Similarly, ICDD and multimodel containers structure heterogeneous data, but lack inherent mechanisms for ensuring temporal consistency (Singh et al., 2011). Versioning systems, such as Global Information Tracker (Git), Apache Subversion, and Ontology-based Programming Models, are widely used for tracking changes in software development and ontologies. However, these methods are not inherently designed for structured chronological modeling in AECO workflows, where updates must be linked to specific lifecycle events. The OCCP addresses this gap by integrating chronological records directly within the semantic model. Using IFC-linked timestamps, OCCP captures the evolution of a component across different versions in context of the construction-specific lifecycle, providing a chronology that could potentially be used in combination with Git-based repositories to design a bi-directional version tracking system. This would enable a fine-grained traceability approach, allowing stakeholders to track changes within both the IFC model and its associated temporal records, enabling transparency in model evolution.

## The core concept of OCCP

The OCCP is based on the OWL-Time ontology and extends its classes and object properties to provide a semantic representation of the building lifecycle. This ontology is part of the concept of chronological models (Vaatz et al., 2023), which aims to combine IFC-based models with the OCCP and other ontologies, such as the Bridge Topology Ontology (BROT) (Hamdan et al., 2020) or the Building Topology Ontology (BOT) (Rasmussen et al., 2020), and model-related datasets using ICDD. In this concept, OCCP provides the temporal structure for capturing lifecycle-relevant states and their timestamps, assigning them to phases or milestones within the cMod framework, focusing on long-term traceability rather than exhaustive model versioning.

The general temporal structure of the OCCP, as shown in Figure 1, serves as a flexible template demonstrating a possible lifecycle to illustrate the ontology's principles. It is designed to ensure applicability to a wide range of AECO projects by not adapting any national-specific construction process definitions such as the German HOAI (Honorarordnung für Architekten und Ingenieure), while allowing customization to represent diverse construction lifecycles tailored to specific project needs.

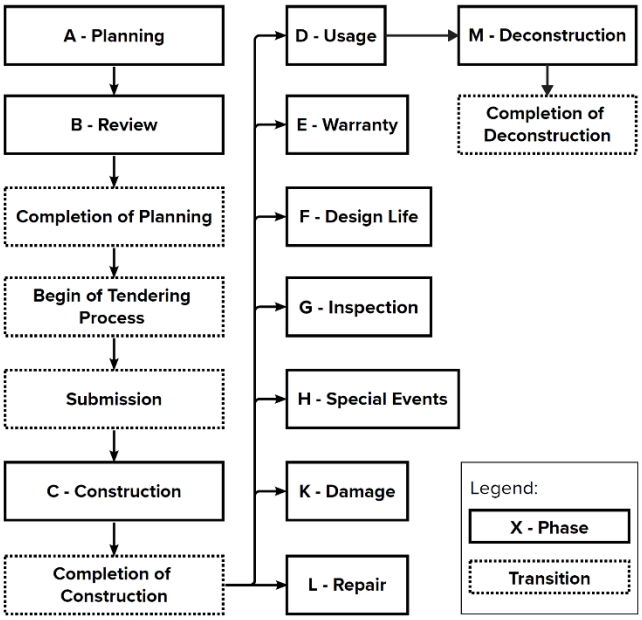


*Figure 1: Overview of the temporal structure of the OCCP*

### Methodology of development

The development of the OCCP followed a structured methodology based on domain-specific requirements, expert insights, and semantic validation. Given the complexity of AECO workflows, the ontology was designed to represent real-world temporal constructs, lifecycle transitions, and iterative review cycles while supporting adaptability across different project structures by extending the OWL-Time ontology with specialized classes and properties, enabling dependency tracking, logical sequencing, and event recording. Industry expert input further refined OCCP's classifications, ensuring it accommodates iterative planning, review processes, and real-world deviations such as delays, rework cycles, or phased repairs.

OCCP was designed to provide a semantic approach to create component- and building-model-specific

chronologies across the entire lifecycle, enabling a structured representation of temporal data that adds a contextual dimension to lifecycle information. This supports addressing key challenges in AECO workflows: (1) tracking iterative processes like review cycles and their timeline impacts, (2) documenting component states across lifecycle phases, and (3) reconstructing event histories with temporal precision. These capabilities can be achieved using SPARQL queries on RDF datasets, offering expressive temporal reasoning beyond SHACL constraints. Together, they highlight OCCP's potential to track changes, compare planned versus actual schedules, and maintain structured, machine-readable chronological records within BIM and multimodel environments.

**Classes**

The OWL-Time ontology defines temporal entities, distinguishing between *time:Instant* (specific points in time without duration) and *time:Interval* (spans between two instants). The OCCP extends the OWL-Time ontology by introducing *occp:Phase, occp:Cycle,* and *occp:Process* as subclasses of *time:Interval* to provide a flexible, semantically enriched lifecycle framework. *occp:Phase* (e.g., *occp:PhaseD_Usage*) represents main chronological blocks of a building's lifecycle, such as planning or usage, serving as a customizable base structure.

*occp:Cycle* (e.g., *occp:CycleA_PlanningReview*) captures iterative processes, like multiple review cycles, occurring within or across phases. *occp:Process* combines phases and transitions into complete or partial workflows, enabling nested modeling (e.g., a repair process within *occp:PhaseD_Usage*).

To structure milestone events, OCCP extends *time:Instant* by defining *occp:Transition* and phase-related instants.

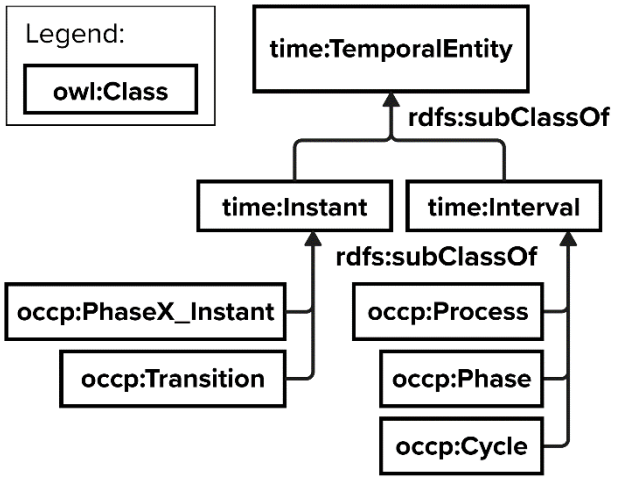


*Figure 2: General class hierarchy of OCCP*

Transitions mark key milestones between phases or trigger subsequent project steps, such as *occp:Submission*, which signals the transition from planning to construction. Similarly, *occp:CompletionOfPlanning* defines the finalization of planning, ensuring proper sequencing of activities. Phase-related instants, such as *occp:EditBeforeSubmissionToReview* or *occp:BeginningOfPlanning*, provide precise temporal markers within each phase. Following this pattern, OCCP introduces dedicated event classes for each defined phase. Figure 2 illustrates the OCCP class hierarchy, showcasing its extension of the OWL-Time ontology, while Figure 3 provides a detailed representation of the temporal framework for the first two phases, their transitions, and associated instants.

Figure 3 highlights OCCP's structured approach to temporal modeling, demonstrating how the instants related to *occp:PhaseA_Planning_Instant* and *occp:PhaseB_Review_Instant* organize phase-specific events within project lifecycles. The *occp:BeginningOfPlanning* serves as a foundational event marking the start of planning, while *occp:SubmissionToReview* signifies the handoff between planning and review.

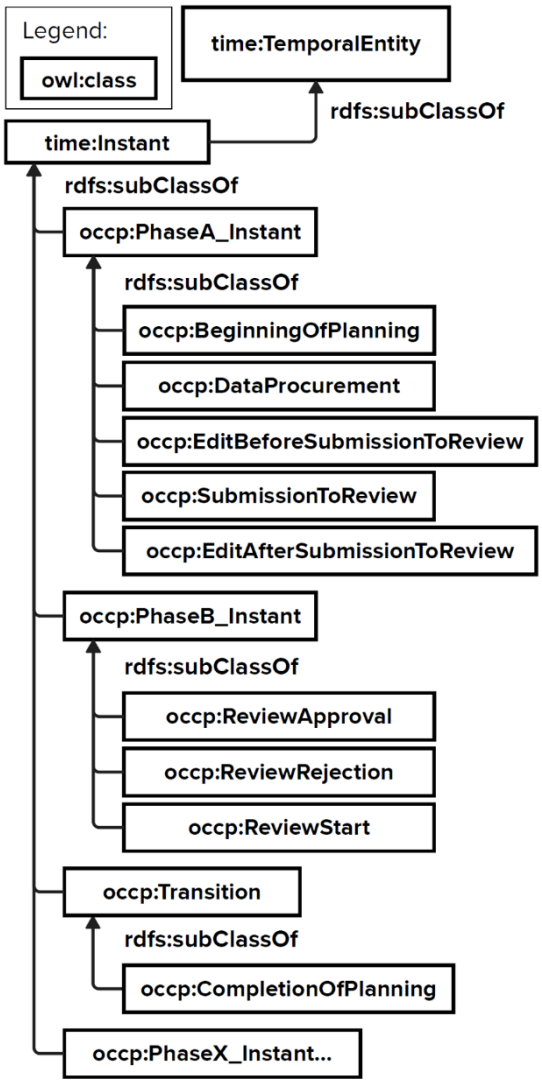


*Figure 3: Extract of phase-related instant classes of OCCP*

Transitions such as *occp:CompletionOfConstruction* define clear boundaries between phases, supporting chronological consistency and milestone validation.

The subclass relationships in Figure 3 show how OCCP extends the OWL-Time Ontology by introducing domain-specific instants and transitions. The ontology's modular structure enhances semantic clarity, distinguishing between different temporal entities through explicit object properties. This modular approach makes the OCCP adaptable to diverse project requirements, supporting both high-level lifecycle planning and granular activity tracking.

### Object and datatype properties

The OWL-Time ontology provides fundamental object properties for describing general relationships between *time:Instant* and *time:Interval,* but lacks the expressiveness needed for structured lifecycle management in AECO projects. The OCCP extends these capabilities by introducing additional object properties, as shown in Figure 4, refining both the *owl:topObjectProperty* and *time:hasTime* (including *time:hasBeginning* and *time:hasEnd*) to define chronological relationships between phases, cycles, milestones, and instances. To establish clear temporal associations, OCCP introduces *occp:hasPhase* and *occp:isInPhase*. The *occp:hasPhase* property embeds events within structured lifecycle phases, ensuring temporal instances are correctly classified. For example, *occp:SubmissionToReview*, which represents the submission of a completed plan for review, is explicitly linked to Phase A (Planning), maintaining a logically

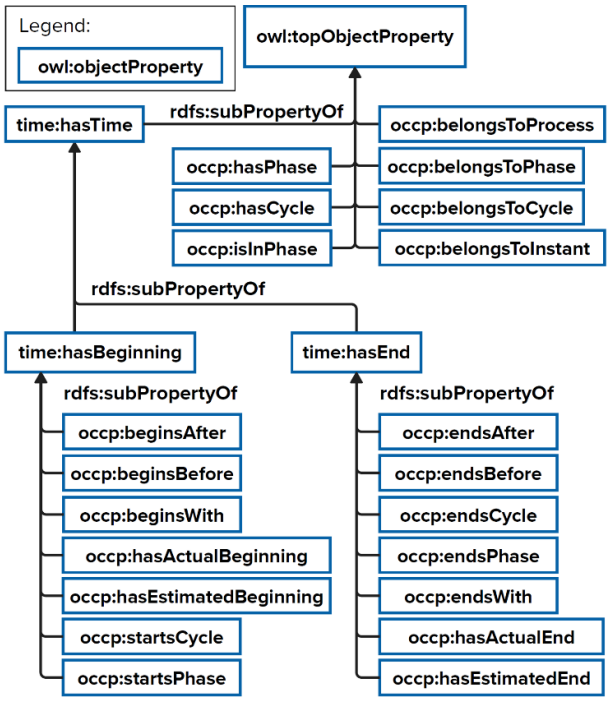


*Figure 4: Object properties of the OCCP*

ordered structure for querying and validation. This enables the tracking of events and automated reasoning over project lifecycles. The *occp:isInPhase* property, in contrast, captures nested or overlapping processes within phases. A practical example is the repair of a damaged component during the Usage phase, where the repair lifecycle must be contextualized within the broader phase. The *occp:isInPhase* relationship enables the representation of concurrent activities, acknowledging the reality of complex, interwoven construction processes.

Beyond these structural properties, OCCP also refines temporal granularity by distinguishing between actual and estimated time values. The *occp:hasActualTime* and *occp:hasEstimatedTime* datatype properties differentiate between verified timestamps and planned or uncertain events. Similarly, the object properties *occp:hasActualBeginning* and *occp:hasEstimatedEnd* extend *time:hasBeginning* and *time:hasEnd*, allowing precise modeling of both anticipated and confirmed project timelines. Additional datatype properties, such as *occp:hasCycleNumber* and *occp:hasProcessID*, track cycle iterations and process identifiers, enhancing traceability. OCCP also introduces relational properties like *occp:beginsBefore*, *occp:beginsAfter*, and *occp:beginsWith* to enable flexible sequencing of phases, cycles, and processes. Unlike *time:intervalIn*, which seems to assume fixed interval relationships, these properties support modeling incomplete or evolving processes, ensuring adaptability for unfinalized phase boundaries until completion. Additionally, hierarchical lifecycle properties (*occp:startsCycle*, *occp:endsCycle*, *occp:startsPhase*, and *occp:endsPhase*) structure iterative processes, supporting the modular definition of project milestones, transitions, and recurrent events. OCCP's semantic framework allows users to track time-related data and to structure, validate, and integrate it within complex lifecycle models, supporting flexibility, consistency, and traceability in construction project management using SHACL shapes.

### SHACL shapes

OCCP's SHACL validation shapes, maintained separately from the OWL-based ontology, enforce temporal and semantic constraints on RDF data. These shapes are flexible: if no prior phase is defined, no check is enforced, allowing partial use (e.g., starting from *occp:CompletionOfPlanning*). By defining and enforcing structural and semantic constraints on RDF data, SHACL ensures that relationships and properties within OCCP adhere to predefined logical rules. This validation layer bridges the gap between theoretical modeling and practical implementation, enabling robust verification of temporal and semantic correctness in an OCCP-enhanced framework.

A primary function of the SHACL validation shapes is to maintain the integrity of OCCP's temporal structure by governing relationships between phases, transitions, and instants. SHACL shapes describe sequential and

hierarchical order, ensuring that each phase has a well-defined beginning and end, validated using properties such as *occp:hasActualBeginning* and *occp:hasActualEnd*. This mechanism prevents erroneous overlaps or gaps between phases, an essential requirement in multi-stakeholder construction projects, where precise chronology is crucial for lifecycle management. A distinct feature of OCCP's SHACL validation shapes is their ability to validate iterative and nested temporal structures. Within the planning phase, for instance, iterative cycles such as *occp:CycleA_PlanningReview* regulate review processes. SHACL shapes ensure that each iteration only begins or ends upon a defined trigger event, such as *occp:ReviewRejection*, and concludes with either another iteration or *occp:ReviewApproval*. This enables logical sequencing and enhances traceability by embedding semantic rules directly into the data model. Beyond internal consistency, SHACL supports cross-contextual validation, enabling OCCP to accurately represent nested lifecycles and parallel workflows. For example, when a construction component undergoes a local repair within an ongoing broader construction phase, SHACL shapes validate that the repair process, including its planning, execution, and completion phases, aligns with the overarching temporal constraints of the primary phase.

In addition, SHACL enforces temporal constraints between milestones and transitions. Shapes tied to properties such as *occp:startsPhase* and *occp:endsPhase* ensure that key events occur in correct sequence, e.g., *occp:CompletionOfPlanning* must precede the start of subsequent phases. Furthermore, these shapes verify that estimated timestamps (*occp:hasEstimatedTime*) and actual timestamps (*occp:hasActualTime*) maintain logical consistency, preventing misalignment between planned and actual project timelines. A key strength of OCCP's SHACL framework is its modularity, which allows for the separation of constraints and validation results via reusable shapes. In this regard, project-specific shapes can be applied for model validation without compromising the core ontology structure. Users can extend OCCP's validation mechanisms to comply with regional planning standards, industry-specific constraints, or unique project milestones. This adaptability extends OCCP's applicability across diverse AECO workflows.

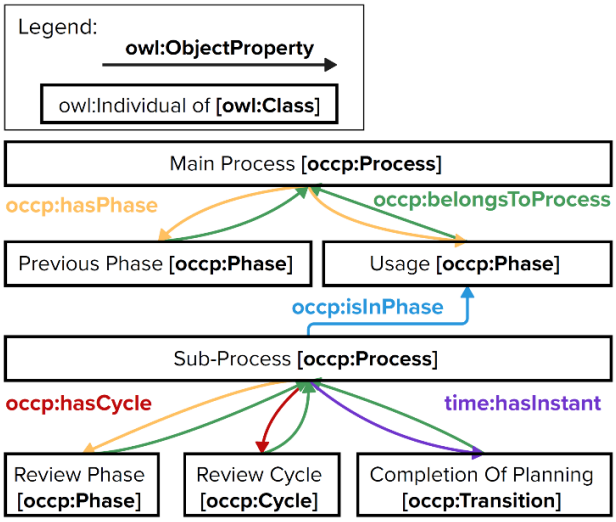


*Figure 5: Macro view of an example of a multilayered process*

In practice, SHACL enhances data validation workflows by providing explicit violation reports when data fails to meet predefined constraints. If a phase violates temporal boundaries or a transition lacks proper phase alignment, SHACL generates clear diagnostic feedback. This automated validation is particularly valuable in dynamic BIM and multi-model environments, where continuous updates necessitate rigorous and consistent verification to maintain data accuracy and reliability.

## Examples for OCCP application

Figure 5 provides a macro view of a multilayered process, illustrating OCCP's ability to structure nested lifecycles. A Main Process orchestrates top-level phases (Previous Phase, Usage), while a Sub-Process within Usage integrates a Review Phase, an iterative Review Cycle, and a transition (Completion of Planning). This hierarchical flexibility, enabled by properties like *occp:hasPhase* and *occp:isInPhase*, reflects OCCP's philosophy of adaptable, semantically rich temporal modeling across diverse AECO workflows. Figure 6 presents a multi-layered example demonstrating how OCCP structures time records and links them to phase-related instances. Layer 1 illustrates the sequential order of phases (Block 1), while Block 2 shows how a component individual (semantic representation) connects to phase-related instances via *occp:hasInstant*, linking components to *occp:Phase(X)_Instants*. Each instant is assigned a timestamp (e.g., via *time:hasTime*) as indicated in Legend 2. The main diagram depicts six blue events within the planning phase and four red events within the review phase. Event 1 (Ind. 1) marks the beginning of planning, which automatically initiates a new planning phase (IND:PhaseA_Planning) and a new planning-review cycle (IND:CycleA_PlanningReview). After the first submission for review (Ind. 4), the review phase starts (IND:PhaseB_Review) through an OCCP SHACL shape, which ensures that *occp:startsPhase* properly initiates the review phase. A rejection (Ind. 6) ends the current cycle (Ind. I.), prompting a new iteration (Ind. II.) with IND:PhA_Edit_ASTR (edit after submission to review). Once the review is approved (Ind. 7), the 2nd cycle (Ind. II.) and phases A and B (Ind. A & B) conclude, marking the transition to construction (Ind. C).

This example demonstrates how OCCP records and connects temporal events across phases, ensuring that their relationships and sequential logic are semantically structured. The same approach applies to subsequent phases, such as construction (Phase C) or usage (Phase D), as well as transitions like submission or completion of construction. Combining the approaches shown in Figures

5 and 6 demonstrates OCCP's flexibility to represent multilayered scenarios, such as a repair process nested within a broader lifecycle. Consider a bridge bearing replacement during *occp:PhaseD_Usage*: a Main Process orchestrates top-level phases (Previous Phase, Usage), while a Sub-Process embeds a repair lifecycle. This Sub-Process, linked via *occp:isInPhase* to Usage, includes a Review Phase, an iterative Review Cycle (e.g., *occp:CycleA_PlanningReview*), and a milestone like Completion of Planning. Inspections record damage, triggering planning and review iterations until approval, impacting bridge operation. OCCP also distinguishes estimated versus actual timestamps using datatype properties (*occp:hasEstimatedTime*, *occp:hasActualTime*) for instants (e.g., review start), linked to phase boundaries *via occp:hasEstimatedEnd* and *occp:hasActualBeginning*. This enables tracking delays, e.g., when reviews exceed estimates, while SHACL shapes ensure sequential logic. By structuring such interdependencies semantically, OCCP delivers traceable, adaptable chronological records across AECO workflows. The ontology's SHACL shapes validate compliance with defined chronological relationships, ensuring logical consistency while allowing flexibility for diverse project requirements.

Figures 5 and 6 illustrate OCCP's ability to structure lifecycle processes using phases, cycles, and transitions within a semantic framework. Figure 5 shows a multilayered process with nested Sub-Process in Usage, while Figure 6 details sequential phases (PhaseA_Planning, PhaseB_Review) and cycles (CycleA_PlanningReview). OCCP's design tackles the challenges mentioned in the methodology of development by leveraging its semantic framework: (1) Iterative cycles (e.g., *occp:CycleA_PlanningReview*) track review iterations and compare planned (*occp:hasEstimatedEnd*) versus actual (*occp:hasActualEnd*) timelines; (2) Component states are semantically linked to phase-specific instants (e.g., *occp:PhaseA_Instant*) via *occp:hasInstant*, capturing lifecycle context; (3) Event histories are reconstructed using timestamps (e.g., *time:inXSDDate* or *occp:hasActualTime*) within a semantically enriched model. These examples illustrate OCCP's ability to create machine-readable chronological records in BIM and multimodel environments.

## Discussion

OCCP's development highlights both its strengths and avenues for refinement. Its structure, designed as a flexible template (Figure 1), captures a sample AECO lifecycle while allowing customization for diverse project needs, addressing concerns about incomplete phase representations. This adaptability stems from a modular design—properties like *occp:isInPhase* and

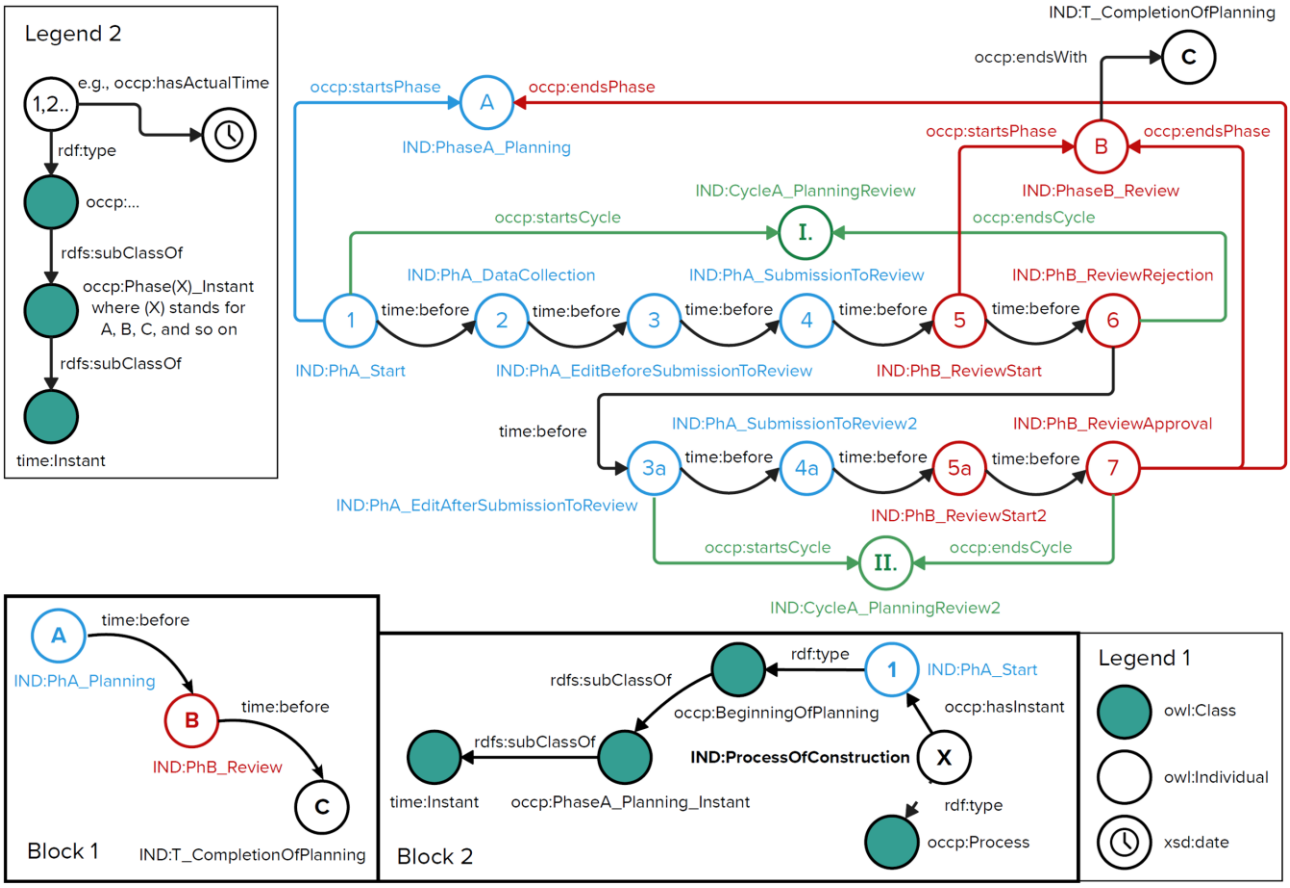


*Figure 6: Detailed view of a process - showing the connection of instants and phases*

*occp:hasCycle* enable nested modeling (Figure 5)—although its complexity could deter users unfamiliar with semantic tools. A broader vision emerges: An Ontology for Chronological Processes (OCP) could extend OWL-Time generically, omitting predefined phases or AECO ties. OCCP would then layer atop this, adding domain-specific constructs and SHACL shapes, enhancing reusability across fields.

Reflecting on its validation, OCCP's key challenges replaced competency questions to focus on practical needs (e.g., tracking iterations), though their derivation from expert input warrants further elaboration in future work. The ontology prioritizes lifecycle structuring over versioning, distinguishing it from IFC's underused mechanisms (e.g., IfcOwnerHistory). SPARQL queries validate these capabilities, offering expressive power beyond SHACL's constraint checking—e.g., querying cycle durations—yet practical examples remain limited here. Similarly, OCCP's flexibility supports partial use (e.g., starting from *occp:CompletionOfPlanning*), but this assumes user-defined SHACL adjustments, a potential standardization hurdle.

Practical integration poses a challenge. While OCCP delivers machine-readable records (Figure 6), linking them to BIM tools or schedules requires automation, currently manual. Publishing OCCP's TBox and shapes (available for examination as work-in-progress here: https://github.com/DigitalizeMe/OCCP/tree/main) is a start, but real-world adoption demands case studies testing scalability, e.g., adapting to regional workflows or simplifying *occp:Process* if nested phases suffice. Future work could split OCP as a core ontology, refine OCCP's AECO focus, and develop software bridging semantic data to industry practice. This evolution, rooted in cMod's vision, could transform chronological management, balancing flexibility with usability.

## Outlook and conclusion

The OCCP is nearing public release, including its source code and documentation, marking a significant milestone in its development. The next phase focuses on its practical implementation within chronological models, designed to unify lifecycle-based temporal data with broader lifecycle information management systems. The work on the cMod framework focuses on operationalising OCCP's semantic structure, enabling real-time validation, version tracking, and dataset analysis linked to IFC models while supporting graph-based reasoning; cMod will facilitate automated consistency checks and lifecycle simulations. A dedicated software interface is under development to support cMod creation, management, and model-stage visualization. OCCP extends the OWL-Time ontology with domain-specific constructs such as phases, cycles, transitions, and phase-specific instants, providing a structured framework for integrating time-based data into BIM workflows, while contributing to advancing lifecycle-oriented construction informatics by adding flexibility for designing unfinished processes through additional object properties. Its SHACL-based validation enables logical consistency, minimizing errors and improving data reliability. By supporting iterative cycles, overlapping phases, and complex lifecycle structures, OCCP enhances timeline visibility, traceability, and automated consistency checks, enabling structured temporal information management. However, its practical impact depends on (semi-)automated workflow and BIM tool integration, a focus of future work. Furthermore, OCCP addresses the identified lifecycle challenges, demonstrating its capability to retrieve temporal insights, compare planned versus actual schedules, and track iterative processes like review cycles. This approach, supported by SPARQL queries and SHACL validation shapes, enables the OCCP to store and structure time-related data while supporting the active verification of chronological consistency and logical sequencing. While further refinement is needed to optimize its integration with industry tools, its structured approach to time management in BIM and multimodel workflows provides a solid foundation for improving consistency, traceability, and automation of chronological records. Future work will focus on expanding its application, enhancing interoperability, and evaluating its real-world impact through practical case studies and software implementations.

## Declaration of generative AI and AI-assisted technologies in the writing process

During the preparation of this work, the authors used Grok 3 and Deepl in order to improve readability and language of the work. After using these services, the authors reviewed and edited the content as needed and take full responsibility for the content of the publication.

## References


Autodesk (2018) Construction Disconnected: The High Cost of Poor Data and Miscommunication. https://www.autodesk.com/blogs/construction/construction-disconnected-fmi-report/ . Accessed date: 30th January 2025.

Beetz, J., van Leeuwen, J. P., & de Vries, B. (2009) IfcOWL: A case of transforming EXPRESS schemas into ontologies. Artificial Intelligence for Engineering Design, Analysis and Manufacturing, 23(1), 89-101. DOI: 10.1017/S0890060409000122

Berners-Lee, T., Hendler, J. & Lassila, O. (2001) The Semantic Web. Scientific American, 284(5), 34-43. https://doi.org/10.1038/scientificamerican0501-34

Borrmann, A., König, M., Koch, C., & Beetz, J. (2018) Building Information Modeling: Technology Foundations and Industry Practice. Springer International Publishing. DOI: 10.1007/978-3-319-92862-3

buildingSMART International (2023) Industry Foundation Classes (IFC) – Standard. https://www.buildingsmart.org/standards/bsi-standards/industry-foundation-classes/ . Accessed date: 30th January 2025.

Car, N.J., Homburg, T., Perry, M., Knibbe, F., Cox, S.J.D., Abhayaratna, J., Bonduel, M., Cripps, P.J., & Janowicz, K. (2024). OGC GeoSPARQL - A Geographic Query Language for RDF Data (Version 1.1). Open Geospatial Consortium (OGC). https://docs.ogc.org/is/22-047r1/22-047r1.html. Accessed date: 29th January 2025.

Cox, S. & Little, C. (2006) Time Ontology in OWL. https://www.w3.org/TR/owl-time/ . Accessed date: 29th January 2025.

Debruyne, C. & McGlinn, K. (2020). Reusable SHACL Constraint Components for Validating Geospatial Linked Data. In: GeoLD@ESWC 2020 – Proceedings of the 1st International Workshop on Geospatial Linked Data. Heraklion, Greece. https://chrdebru.github.io/papers/2021-geold-preprint.pdf . Accessed date: 30th January 2025.

Fuchs, S., Kaddolsky, M. & Scherer, R. J. (2011) Formal description of a generic multi-model. In 2011 IEEE 20th International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, 205-210.

Grilo, A., & Jardim-Goncalves, R. (2010) Value proposition on interoperability of BIM and collaborative working environments. Automation in Construction, 19(5), 522-530. DOI: 10.1016/j.autcon.2009.11.003

Hamdan, A. H. & Scherer, R. J. (2020) Integration of BIM-related bridge information in an ontological knowledgebase. In: Proceedings of the 8th Linked Data in Architecture and Construction Workshop (LDAC)

Karlapudi, J., Valluru, P., & Menzel, K. (2021). Ontology Approach for Building Lifecycle Data Management. Proceedings of the ASCE International Conference on Computing in Civil Engineering (i3CE2021). Orlando, Florida, USA.

Knublauch, H. & Kontokostas, D. (2017). Shapes Constraint Language (SHACL). https://www.w3.org/TR/shacl/ . Accessed date: 30th January 2025.

Lassila, O., & Swick, R. R. (1999). Resource Description Framework (RDF) Model and Syntax Specification. W3C Recommendation. https://www.w3.org/TR/1999/REC-rdf-syntax-19990222/ . Access date: 30th January 2025.

Pauwels, P., Zhang, S., & Lee, Y.-C. (2017). Semantic web technologies in AEC industry: A literature overview. Automation in Construction, 73, 145–165. https://doi.org/10.1016/j.autcon.2016.10.003

Pauwels, P., & Terkaj, W. (2016) EXPRESS to OWL for construction industry: Towards a recommendable and usable ifcOWL ontology. Automation in Construction, 63, 100-133. DOI: 10.1016/j.autcon.2015.12.003

Rasmussen, M. H., Lefrançois, M., Schneider, G. & Pauwels, P. (2020) BOT: The Building Topology Ontology of the W3C Linked Building Data Group. In: Semantic Web. DOI: 10.3233/SW-200385.

Sacks, R., Eastman, C., Lee, G., & Teicholz, P. (2018). BIM Handbook: A Guide to Building Information Modeling for Owners, Designers, Engineers, Contractors, and Facility Managers (3rd ed.). John Wiley & Sons. DOI: 10.1002/9781119287568

Scherer, R.J., & Schapke, S.-E. (2011). A distributed multi-model-based management information system for simulation and decision-making on construction projects. Advanced Engineering Informatics, 25(4), 582–599.

Singh, V., Gu, N. & Wang, X. (2011) A Theoretical Framework of a BIM-Based Multi-Disciplinary Collaboration Platform. Automation in Construction, 20(2), 134-144. DOI: 10.1016/j.autcon.2010.09.011

Törmä, S. (2022) Digital Construction Works: Vers. 0.5. https://digitalconstruction.github.io/v/0.5/index.html. Accessed date: 3rd April 2025

Vaatz, A., Hamdan, A.-H., Al-Sadoon, N., Wogan, M., Menzel, K., (2023). Integration of semantic temporal information in BIM using ontologies. European Conference on Computing in Construction. Crete, Greece, July 10-12. DOI: 10.35490/EC3.2023.281

Volk, R., Stengel, J., & Schultmann, F. (2014). Building Information Modeling (BIM) for existing buildings — Literature review and future needs. Automation in Construction, 38, 109–127. https://doi.org/10.1016/j.autcon.2013.10.023

Xie, Y., Zhan, N., Zhu, Q., Zhan, J., Guo, Z., Qiao, C., Zhu, J., & Xu, B. (2024). Multimodal data visualization method for digital twin campus construction. International Journal of Digital Earth, 17(1), 145–165. https://doi.org/10.1080/17538947.2024.2431624