An Ecosystem of DSMLs for Building Commissioning

Philipp Zech*, Emanuele Goldin*, Christoph Zallinger*, Sascha Hammes†, Philipp Pobitzer*, Judith Michael‡, Ruth Breu*
*Department of Computer Science, University of Innsbruck, Tyrol, Austria
†Unit of Energy Efficient Buildings, University of Innsbruck, Tyrol, Austria
‡University of Regensburg, Programming and Software Engineering, Germany
Emails: {philipp.zech, emanuele.goldin, christoph.zallinger, philipp.pobitzer, sascha.hammes, ruth.breu}@uibk.ac.at, judith.michael@ur.de

***Abstract*—Building Information Modeling (BIM) has advanced the construction industry from simple 2D/3D CAD to semantically rich, object-oriented, and visually intuitive modeling. However, unlike, e.g., the UML or SysML, BIM's modeling formalisms lack universal applicability and full life cycle coverage, confining its use largely to architectural design. Key trades like building control remain insufficiently integrated, leading to inefficiencies and misaligned design and operational requirements. This paper presents an ecosystem of domain-specific modeling languages (DSML) integrated with BIM, to expand its capabilities to additional trades such as building control. By embedding operational requirements (cf. the building control system installation) during the design phase by integrating them into the BIM model, the subsequent application of model-driven deployment ensures alignment with operational objectives. Validation in a Living Lab at the University of Innsbruck highlights the approach's benefits, including faster commissioning and improved energy efficiency, representing a step toward comprehensive, BIM-based building design and operation.**

***Index Terms*—Model-driven Engineering, Domain-specific Modeling Languages, Building Information Modeling, Building Programming, Building Commissioning**

I. Introduction

Building Information Modeling (BIM) has redefined the architecture, engineering, and construction (AEC) industries by introducing a model-based, data-driven approach to managing buildings throughout their life cycle [1]. Unlike traditional Computer-Aided Design (CAD) systems, BIM *aims* at creating a shared, multidimensional digital representation of a structure that integrates geometric, semantic, and operational data. BIM's goal is to improve efficiency, reduce waste, and enhance collaboration across project phases—from design and construction to operation and eventual decommissioning [2]. Recent studies emphasize its evolving role in integrating technologies like digital twins (DT) [3], enabling real-time simulations and smart decision-making for sustainable building management [4]. DTs are virtual representations of physical systems utilized for a range of applications, such as monitoring, optimization, and predictive analytics [5]. They provide real-time insights into system behavior, thereby facilitating informed decision-making and anomaly detection across various domains, including cyber-physical systems (CPS) like modern buildings [5], [6], [7].

This research has received funding from the Austrian Research Promotion Agency (FFG) under Grant Agreement No.: 898708, TwinLight.

By *similarly* embedding performance and life cycle considerations into the design phase, BIM can *facilitate* tasks such as energy analysis, material optimization, and life cycle assessments [8], [9]. Despite challenges in interoperability and standardization, BIM's widespread adoption demonstrates its potential to streamline project execution, improve sustainability, bridge gaps between stakeholders, and eventually the gestation of BIM-based DTs [10], [11], cementing its position as a cornerstone of modern construction methodologies [12].

To fully realize the potential of BIM across a building's entire life cycle, from design to operation, a BIM model must evolve to integrate the modeling, programming, and configuration of building control systems (BCS). This integration would allow the operational requirements of a building to be embedded directly into the design phase, ensuring that the BIM model reflects the building's operational model, ready for deployment. However, a fully integrated, model-driven workflow for both design and deployment of BCS is not yet in place [13], leaving a gap in the seamless application of BIM across all stages of building development and operation [10].

We suggest the application of model-driven engineering (MDE) for expanding BIM towards BCS. MDE is beneficial for managing complexity [14], improving automation, and excels at tailoring engineering processes to different domains (or trades such as BCS) by capitalizing on domain- and platform-specific modeling [15], [16]. Specifically, the application of domain specific modeling languages (DSML) [16] for extending existing modeling formalisms, e.g., BIM, towards novel domain concepts such as BCS, is worth investigating. Following this reasoning, in this paper, we introduce a novel ecosystem of DSMLs for *modeling*, *programming*, and *configuring* of BCS devices, culminating in an approach to model-driven development and deployment of BCS. This DSML-based ecosystem is designed to enhance BIM's capabilities and offer a streamlined workflow that bridges the design and operational phases. Our work also delivers the necessary tooling environment and is evaluated in the context of a Living Lab, i.e., a designated office used as a user-centered research environment at the University of Innsbruck (cf. Section VI) where we demonstrate faster commissioning cycles and improved energy efficiency by facilitating quicker adaptation to evolving operational requirements.

II. Background

The application of MDE for developing advanced CPS has gained much traction over the last years [17]. Modern buildings alike describe such CPS [17], yet the application of MDE as an integrated part of the development process so far has rather focused on dedicated *sub*-CPS (e.g., elevators [17]). With the advent of BIM however, a new cornerstone technology for creating unified models of built assets has emerged, promising enhanced collaboration and life cycle management across design, construction, and operation phases. However, its adoption in building commissioning — particularly for integrating BCS and DTs — remains challenging. Current workflows often rely on manual configurations and proprietary tools, leading to inefficiencies, information silos, and suboptimal building performance [10]. These issues are further compounded by the lack of interoperability between trades, such as heating, ventilation and air conditioning (HVAC), lighting, and construction, as well as inadequate feedback loops between operational data (e.g., the current building state or its occupancy and corresponding usage) and the BIM environment [18].

The interdisciplinary research project TwinLight [19] addresses these gaps by introducing a BIM-based approach to automate building commissioning. By extending BIM with BCS and introducing corresponding DSMLs, TwinLight facilitates collaborative workflows and seamless data exchange across stakeholders. Its emphasis on pre-configuring BCS directly within BIM models reduces commissioning errors, improves energy efficiency, and supports dynamic, short-cycle (i.e., accelerated or streamlined), re-commissioning through DT integration. By ensuring that the BIM model – next to architectural and structural details – also integrates a comprehensive model of the BCS, it readily facilitates the establishment of a DT's connection to physical devices in a building for bidirectional data exchange. Consequently, the DT functions as both a real-time monitoring center and a decision-making tool, continuously synchronizing with the physical twin through live data streams from the BCS. The DT enables seamless automation, allowing changes made in the virtual model such as optimizing HVAC settings or adjusting lighting schedules to be immediately reflected in the physical twin [20]. The project's innovations, such as leveraging MDE for BIM, promise significant advances in automating commissioning processes and ensuring the scalability of these solutions to different building types (e.g., residential, commercial, industry or infrastructure).

A. Challenges and Contributions

Despite recent advancements, we have identified several challenges in realizing model-driven building commissioning [13]:

- Inadequate integration between BIM, BCS, and operational feedback loops impairs collaboration among design, construction, and operation stakeholders,
- lack of adequate support for the BIM-based pre-configuration of BCS and subsequent building commissioning,
- reliance on closed systems and protocols limits interoperability and reuse of control logic across buildings, and
- limited support for incorporating operational data back into BIM models for iterative optimization.

Commensurate with these, in this paper, we deliver the following contributions:

1) An ecosystem of DSMLs and tooling, integrated with BIM tools to support modeling, programming, and configuration of BCS,
2) model-driven automation for automated commissioning by generating runtime artifacts directly from enriched BIM models,
3) feedback loops for operational data to refine control systems and short-cycle building performance optimization, and
4) improved collaboration through a shared modeling environment, enhanced transparency and reduced inefficiencies in commissioning processes.

In solving these challenges, we address the following research questions:

RQ1 How can DSMLs extend BIM capabilities to effectively model, program, and commission BCS?

RQ2 To what extent can model-driven automation improve the efficiency and accuracy of building the commissioning process, particularly through runtime artifact generation from enriched BIM models?

RQ3 How can operational feedback loops enhance building performance optimization during the life cycle, particularly in the context of energy efficiency and occupant comfort?

Our work follows the Design Science Research (DSR) [21] paradigm and produces a prototype as an artifact [22]. The development of our artifact follows a systematic process, starting with requirements engineering and ends with implementing a prototype and its evaluation using a case study. Our artifact is implemented as a solution to the following design science problem, outlined using the DSR template [21]:

Expand *BIM* (context)
by designing *an ecosystem of DSMLs with tool support* (artifact)
that satisfies *building control system modeling, programming, and configuration* (requirement)
to deliver *model-driven building commissioning.* (goal)

DSR usually refers to an *artifact* as a prototype at Technology Readiness Level 3 (TRL3) [23], representing a conceptual solution at an early stage of technology development. Using a prototype implemented in a Living Lab which compares to an operational environment, our proposal achieves TRL6 (cf. Section VI).

III. Requirements for Model-driven Building Commissioning

TwinLight highlights the transformative potential of integrating BIM with advanced modeling of BCS. Central to this is the development of modeling languages that enable seamless transitions between design and operational phases. Modern buildings operate as complex CPS, necessitating modeling tools that address the growing complexity of interconnected components while promoting collaboration among diverse

TABLE I
REQUIREMENTS FOR OUR GRAPHICAL (GR*X*), TEXTUAL (TR*X*), AND THE CONFIGURATION DSML (CR*X*).

Graphical DSML Requirements
GR1: *Platform independence*: Enable high-level, domain-specific modeling abstracted from hardware or software dependencies.
GR2: *Expressiveness*: Represent the topology and connectivity of BCS, including sensors, actuators, and controllers in an intuitive visual format.
GR3: *Scalability*: Handle complex system hierarchies and extensive building projects.
GR4: *Interoperability*: Integrate seamlessly with BIM tools and enable bi-directional data exchange for cross-disciplinary collaboration.
GR5: *Validation Support*: Include mechanisms to check model validity and ensure compliance with domain-specific constraints.
Textual DSML Requirements
TR1: *Platform Independence*: Enable high-level, domain-specific programming abstracted from hardware or software dependencies.
TR2: *Automation*: Facilitate the generation of runtime artifacts, including control algorithms and configurations through model-to-text transformations.
TR3: *Readability*: Provide a clear and concise syntax to promote adoption among non-programming domain experts.
TR4: *Extensibility*: Support customization for evolving use cases and integration of novel control paradigms.
Configuration DSML Requirements
CR1: *Platform Independence*: Enable high-level, domain-specific configuration abstracted from hardware or software software dependencies.
CR2: *Dynamic Adaptability*: Allow on-the-fly updates to system configurations to accommodate changes in building usage or occupancy.
CR3: *Integration with Runtime Systems*: Ensure smooth deployment of configurations into live operational environments.
CR4: *Simplicity*: Minimize complexity to facilitate configuration by facility management teams without requiring deep technical expertise.

stakeholders. Table I outlines the language requirements for our proposed DSMLs for facilitating model-driven building commissioning.

Graphical DSMLs [24] can provide intuitive representations of system topologies, ensuring that components such as sensors, actuators, and controllers are well-integrated within BIM models. These languages also enable stakeholders to collaboratively explore and validate configurations during early design stages, mitigating errors and reducing performance gaps. Textual DSMLs [24] complement this by supporting platform-independent definition of control logic, enabling automated generation of runtime artifacts and promoting consistency across diverse system implementations. Finally, configuration languages [25] must support continuous commissioning by enabling real-time adaptation to changing operational conditions, ensuring buildings meet evolving energy and comfort targets throughout their life cycle. The requirements for these modeling languages, as motivated from TwinLight, are essential to advancing sustainable and efficient building operations, bridging the current gap between static design and dynamic operational performance. By addressing interoperability, scalability, and adaptability, our proposed DSMLs contribute substantially to buildings meeting their operational requirements by meriting these already during early design phases.

IV. AN ECOSYSTEM OF DSMLS FOR BUILDING COMMISSIONING

In the following we outline our contribution from a conceptual point of view, followed by introducing our three DSMLs for (i) graphical modeling BCS components, (ii) textual modeling of related control logic by leveraging the modeled BCS components, and (iii) BCS components' configurations by low-level interface specifications. Figure 1 outlines the interrelations of our ecosystem of DSMLs along their purpose. Specifically, the *BCS DSML* is used for modeling BCS components. The *Control DSML* enables programming of the modeled BCS and generating executable runtime artifacts using model-2-text generation (cf. Section IV-A). Finally, the *Configuration DSML* - during model-2-model generation on the grounds of both graphical and textual models - is used for emitting the BCS configuration for the middleware (e.g., device IDs, device endpoint descriptions, etc. cf. Section IV-B).

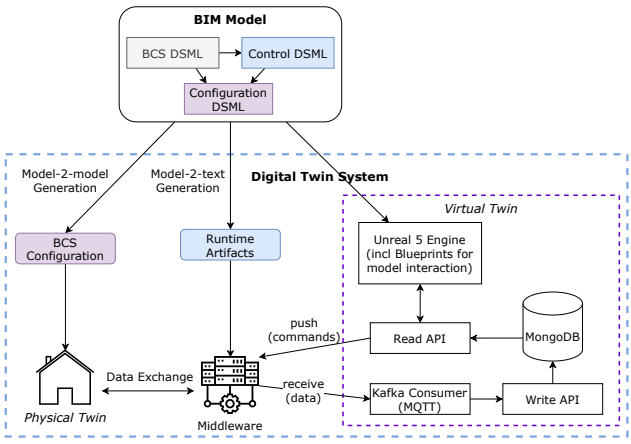

Fig. 1. Our proposal augments the BIM modeling formalism with tailored DSMLs for modeling, programming, and configuring BCS. The resulting BIM model is employed for model-2-model and model-2-text generation of BCS configuration and runtime artifacts for the middleware [26]. The BIM model further establishes the virtual representation (cf. twin) to complete the DT. Please refer to Section V-B for a detailed description of the virtual twin.

Our DSML for modeling BCS and an early prototype for modeling control logic have already been introduced in earlier works and were well received, as evidenced by a Technology Acceptance Model (TAM)-based evaluation [18], [26], [27]. In this work, we focus on a redesign of the *Control DSML*, the newly developed *Configuration DSML*, model generation

of the *Configuration DSML*, and the integration in a DT framework based on Unreal Engine 5 (UE5) [28] as shown in Figure 1. Please see our artifact for a metamodel of the *BCS DSML* [22].

A. Control DSML

The left side of Figure 2 illustrates the textual DSML for BCS device programming, which is developed atop the *BCS DSML* by leveraging the devices that are modeled therein. The DSML embodies a systematic framework that enables the interaction among diverse elements. At the core of the model are entities including `Action`, `Device`, and `Property`, each distinguished by attributes that delineate their characteristics and functions. For example, the `Property` encompasses attributes like `type` and `unit`, which facilitate accurate definitions of the processable values available from the BCS. Furthermore, the DSML utilizes constructs such as `SetStatement`, `GetStatement`, and `ConditionalStatement` to articulate the logic that directs device functionality and interactions, integrating intricate expressions and conditions to facilitate significant automation.

In addition, the DSML incorporates logical conditions denoted by `Condition`, along with particular variants including `OrCondition` and `AndCondition`. These conditions facilitate the formulation of advanced decision-making processes within a BCS. This DSML functions as a robust instrument for developers, facilitating automation, optimizing device control, and improving the efficiency of the built environment through clearly articulated programming constructs.

B. Configuration DSML

The right side of Figure 2 provides the last building block by the *Configuration DSML* that builds atop the *BCS* and *Control DSML*s to generate configurations for a middleware which connect to BCS devices [26]. This configuration metamodel maps devices from the *BCS DSML* to middleware-specific settings and uses elements from the *Control DSML* to define behavior and communication. Central to this model is the `Provider` entity, which connects devices to specific providers via a `DeviceToProviderMap`. Providers encapsulate device details, including `host`, `port`, `protocol` (HTTP/HTTPS), and authentication properties like `tokens`. Each device is uniquely identified (linked to the graphical DSML) and assigned a `Provider` for middleware interaction. The metamodel also includes abstractions for modeling device requests (`AbstractRequest`, `Translation` and `AbstractNameToTranslation`), and necessary property mappings (`PropertyMap`) to customize communication and behavior. By combining graphical device definitions and textual behavior specifications, the configuration metamodel facilitates the seamless setup of the middleware, ensuring proper device provisioning and interaction capabilities tailored to the system's hardware (e.g., BCS devices) requirements.

Observe that a configuration model based on this latter DSML usually is not established manually but inferred from both the graphical and textual models pertaining to the modelled BCS using model-2-model transformations (cf. Section V-B).

V. Modeling Infrastructure

Our DSMLs enable an integrated modeling methodology from (i) initial establishment of the BIM model, to (ii) modeling and programming the BCS, to finally (iii) generating the BCS configuration and runtime artifacts. Observe that for establishing the BIM model we employ third-party tools (e.g., Autodesk Revit [29]).

Scope: The modeling methodology covers the modeling, configuring and programming of BCS in BIM models which includes the definition of devices and their connections, and corresponding control algorithms.

Modeling languages and tools: Eclipse Ecore [30] is used for formalizing the abstract syntaxes of our DSMLs, whereas their concrete syntaxes, i.e., the graphical and textual representations of our DSMLs, are implemented using Revit families and Eclipse Xtext [31]. Model-2-model and model-2-text transformations are implemented using Eclipse Xtend [32] atop Xtext. Please see our artifact for the concrete implementations [22].

A. Modeling Methodology

Our modeling methodology comprises three steps, viz. (i) establishing the BIM model, (ii) augmenting the BIM model with BCS devices and control logic, and (iii) generation of the BCS configuration and executable runtime artifacts (cf. BCS control algorithms).

(1) Creating the BIM model In a first step, the BIM model of the built asset is established. During this, we rely on common third-party tools, e.g., Autodesk Revit, for creating this initial model.

(2) Augmenting the BIM model With the BIM model established, next, it is augmented with BCS devices using the *BCS DSML* by modeling each BCS device in the BIM model using the custom families, thereby also spatially locating the various devices in the building. This step further comprises modeling of device properties (e.g., IDs, endpoints, ...) and their interconnections. With all devices modeled and the BIM, as a last activity, BCS control algorithms are implemented in the model using the *Control DSML* to fully augment the model with the BCS. For providing such an integrated development environment we leverage a custom language server [33] that implements our *Control DSML* for our Revit plugin. Figure 3 shows an augmented BIM model (left) with control code (right). Although the code in Figure 3 is schematic, such simplistic rule-based control currently is state of the art in modern BCS [34], [13].

(3) Model-driven Generation Finally, with the BIM model fully augmented with BCS components and corresponding control algorithms, the BCS configuration model and the runtime artifacts are generated using Xtend. The BCS configuration which is emitted using the *Configuration DSML* subsequently is loaded into the middleware (cf. Section V-B); runtime artifacts are generated as executable Lua code for execution in a dedicated runtime environment as part of the middleware (cf. Section V-B). Listings 1, 2 and 3 show a BCS configuration and generated Lua code for the model from Figure 3. Please see our artifact for the actual implementation of the code generators [22].

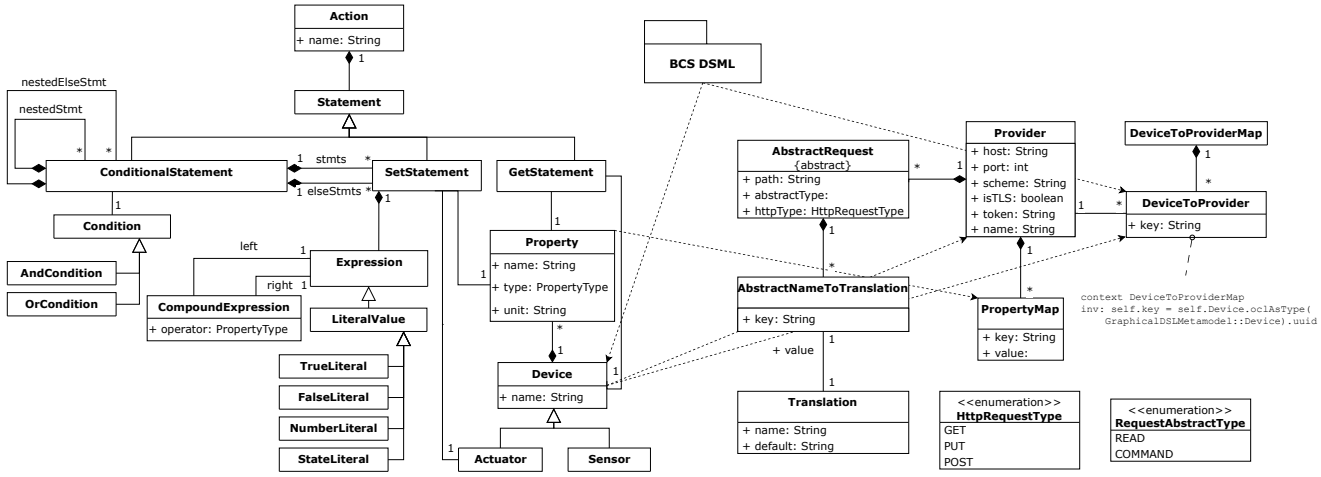

Fig. 2. Metamodels of our *Control DSML* for programming BCS devices (left) and the *Configuration DSML* for configuring the middleware (right). Dotted arrows indicate language dependencies, i.e., the `Device` entity from the *BCS DSML* is leveraged in the *Control DSML*; in turn, entities from the *Control DSML* are leveraged in the *Configuration DSML*, i.e., the `Device` for establishing the `DeviceToProviderMap`.

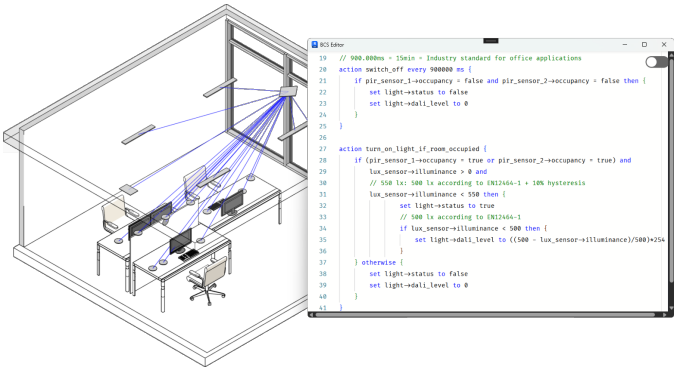

Fig. 3. BIM model of the Living Lab with modeled BCS devices (lozenges denote controllers, circles sensors, and squares actuators) alongside two actions to ensure energy-efficient operation depending on occupant presence (top) and in accordance with sensor-specific switch-off times of 15 minutes for artificial lighting (bottom). Best viewed in color on a computer screen.

```
[
  {
    "vendor": "homeassistant",
    "abstractRequests": [
      {
        "abstractRequestType": "COMMAND",
        "requestType": "POST",
        "pathTemplate": "/api/services/light/turn_on",
        "parameterTranslations": [
          {
            "abstractParameter": "lighting_intensity",
            "concreteParameter": "brightness_pct",
            "defaultValue": "0"
          },
          {
            "abstractParameter": "UUID",
            "concreteParameter": "entity_id",
            "defaultValue": "0"
          }
        ]
      },
      ...
  }
]
```

Listing 1: Excerpt of the BCS configuration for the Living Lab from Figure 3. As to space limitations, we do not show the full configuration (as indicated by the ...).

B. Model-driven Digital Twinning

Following our modeling methodology which capitalizes on the earlier introduced DSMLs, our approach enables an "extended" concept of model-based twinning using the configuration model for the middleware and the building's BCS, as well as the executable runtime artifacts in the form of Lua code. A configuration as shown in Listing 1 is uploaded to the middleware [26] which then automatically connects to the devices in the building as specified in the configuration. In addition, the middleware also provides the runtime environment for the Lua code [26]. Once uploaded, the runtime environment takes care of either executing the scripts once (i) a scheduled period has passed (e.g., every 30mins), or (ii) as soon as a specified value has dropped below or surpassed a threshold (e.g., a room's illuminance has dropped below 500 lx). This allows for the scripts to remain simple, as they do not contain complex control flow logic like loops, making them easier to maintain and execute reliably in real-time environments. The necessary services are both provided by the middleware via gRPC [36] to allow for easy client integration.

The virtual twin of our DT capitalizes on UE5, a powerful, open-source game engine. As outlined in the right of Figure 1, the BIM model is uploaded into UE5 (cf. Figure 4 for the resulting rendering) to create an interactive 3D visualization. For the purpose of achieving the spike solution within the context of the Living Lab, the relevant components of the virtual twin were implemented manually. This comprises a set of Blueprint (a UE5-specific C++-based scripting language) scripts for model interaction, an instance of Apache Kafka [37] for receiving data (via MQTT [38]) from the middleware which is then forwarded into a MongoDB [39] via the `Write`

```lua
mw = require("middleware")
stringtoboolean {["true"] = true, ["false"] = false}

local pir_sensor_1_occupancy =
    stringtoboolean[string.lower(mw.request("pir_sensor_1",
    {"occupancy"}))]
local pir_sensor_2_occupancy =
    stringtoboolean[string.lower(mw.request("pir_sensor_2",
    {"occupancy"}))]
if not pir_sensor_1_occupancy and not pir_sensor_2_occupancy
    then
    mw.command("light", {"status" = tostring(false),
        "dali_level" = tostring(0)})
end
```

Listing 2: Lua code for conformance to 15 minutes switch-off times [35]. Execution scheduling (e.g., every 900000ms, cf. line 20 in Figure 3) is taken care of by the middleware.

```lua
mw = require("middleware")
stringtoboolean {["true"] = true, ["false"] = false}

local pir_sensor_1_occupancy =
    stringtoboolean[string.lower(mw.request("pir_sensor_1",
    {"occupancy"}))]
local lux_sensor_illuminance =
    tonumber(mw.request("lux_sensor", {"illuminance"}))
local pir_sensor_2_occupancy =
    stringtoboolean[string.lower(mw.request("pir_sensor_2",
    {"occupancy"}))]
if (pir_sensor_1_occupancy or pir_sensor_2_occupancy) and
    lux_sensor_illuminance > 0 and lux_sensor_illuminance < 550
    then
        mw.command("light", {"status" = tostring(true)})
        if lux_sensor_illuminance < 500 then
                mw.command("light", {"dali_level" =
                    tostring(((500 - lux_sensor_illuminance) /
                    500) * 254)})
        end
else
        mw.command("light", {"status" = tostring(false),
            "dali_level" = tostring(0)})
end
```

Listing 3: Lua code for regulating artificial lighting at 500 lux to EN 12464-1 for workplace lighting including 10% hysteresis (line 14).

API. Finally, the Read API takes care of forwarding data to UE5 for visualization as well as pushing commands, which result from pushing buttons in the visualization, towards the middleware. Figure 4 shows a screenshot of our virtual twin of the Living Lab from Figure 3.

It is noted that the (i) Blueprint scripts, (ii) value and command parameterizations for the Read and Write API (cf. Figure 1), and (iii) data schemata for the Kafka instance and MongoDB can also be generated from the BCS-augmented BIM model using model-2-text generation. This would necessitate a significant refactoring of the virtual twin's Read and Write APIs. As this contribution does not prioritize the DT, this is regarded as a subject for future work. As our implementation of the DT in UE5 was conducted manually and in an ad hoc manner, it does not constitute part of our artifact.

Our model-driven approach to establish a bi-directional connection between virtual and physical twins via our middleware

Fig. 4. Screenshot of the Living Lab rendered inside UE5 with embedded sensor readings; the *Toggle Lights* button allows turning on/off the light in the actual office space. Red boxes display sensor readings.

enables an agile and continuous style of working akin to modern software engineering, yet in a domain where digitalization efforts only slowly establish themselves [13]. By supporting model-based abstractions it facilitates an efficient workflow by bridging current gaps in building commissioning [13], thereby fostering collaboration and aligning with agile engineering practices to enhance both design and operation of buildings.

VI. Evaluation and Discussion

An office space of 22 m^2 at the University of Innsbruck was established as a Living Lab. Modern energy-efficient lighting systems regulate artificial light depending on occupant presence to a minimum illuminance of 500 lx at the workplace according to EN12464-1 [40] (cf. Listings 2 and 3, and Figure 3). This approach significantly conserves energy relative to manual lighting [41]. Each workstation is equipped with one horizontal illuminance sensor to accomplish this sort of control. Artificial lighting is also regulated based on occupancy to decrease energy consumption relative to manual control [42]. Passive infrared sensors (PIR) are implemented in the Living Lab at each workstation, with detection limited to the respective workstation's range. PIR sensors are prevalent in this application because of their cost-efficiency and simple commissioning. Sensor-related incorrect absence detections can occur during stationary activities, such as PC work. Switch-off times are used to avoid discomfort impairments due to incorrect switching-off [43].

A switch-off time of 15 minutes is implemented in the study object in accordance with industry standards [35] (cf. Listing 2). Glare and overheating protection are essential comfort criteria. The shading system in the office is regulated based on the time of day, following standard market practices for protection. Although automatic controls are more energy efficient, occupants can use manual controls for acceptance, which are valid for 90 minutes before reverting to automatic mode.

The proposed modeling methodology from Section V-A was implemented for the Living Lab (cf. Listing 2 and Figure 3) to facilitate efficient and transparent continuous system improvement. This encompasses continuous commissioning, viz., (i) evaluating actual conditions, (ii) conducting post-occupancy evaluations, and (iii) identifying measures to enhance energy efficiency (cf. De-Graft et al. [44]). In the building design

phase, limited information on occupant behavior necessitates assumptions, resulting in energy performance gaps [45]. Consequently, cyclical post-occupancy evaluations and system adaptation, as facilitated by our proposal, are necessary. To collect the necessary data, all sensor and actuator information is logged in high resolution and made available for post-occupancy evaluations. The occupancy status is recorded for evaluation in accordance with data protection regulations.

The 15 minute switch-off time is a common industry standard for lighting control based on passive infrared sensor technology. However, it does not account for individual occupancy patterns, which typically emerge after commissioning. Adjusting the switch-off time based on actual usage can improve both comfort and energy efficiency [46].

Flexible working hours, the option to work from home and a high level of participation in meetings ensure a high level of occupancy dynamics. Our evaluation revealed considerable variability in attendance times. This leads to sensor-related switch-off times, which result in numerous intervals of artificial lighting operation without occupancy (28.8% of the total operating times during the measurement period from Oct 22, 2024 to Dec 20, 2024). Following Hammes et al. [46], high resolution presence data enables the switch-off times to be adapted to individual presence patterns. For workstation 1 and 2, the switch-off time could be reduced from 15 minutes (industry standard [35]) to 6 minutes without increasing sensor-related false-off rates, which can have a negative impact on comfort (cf. line 20 in Figure 3). These derived switch-off times can be integrated using the proposed modeling method and transmitted to the BCS. The facility manager, as the person responsible for operations, bears sole responsibility for authorization and monitoring, so that no complex reconfiguration of the BCS is required. For the adjustment on Nov 01, 2024, there is a 16.1% reduction in energy consumption for artificial lighting for the subsequent measurement days compared to the reference with a 15-minute switch-off time (cf. Figure 5).

As the occupancy dynamics at the workplace are a variable that can vary greatly depending on the season and work activity, a dynamic adaptation of the switch-off times would prove to be advantageous [47]. The system architecture created in combination with the methodology from Section V-A can provide such a basis.

A. Answering the Research Questions (RQs)

As to **RQ1**, our DSMLs provide domain-aligned graphical and textual representations of BCS components, enhancing stakeholder communication and collaboration. By facilitating model-driven automation to generate runtime artifacts from BIM models, we minimize commissioning time and errors. Our DSMLs enable dynamic adaptability, facilitating real-time updates to system configurations for enhanced energy efficiency and occupant comfort. In summary, DSMLs significantly extend BIM's capabilities, leading to more efficient and effective BCS operations. On the contrary, the use of SysMLv2 (or UML, for analogous reasons) is not recommended: BIM focuses on detailed architectural and spatial modeling, while SysMLv2 targets systems engineering. Integrating BIM data into SysMLv2 leads to unnecessary complexity and inefficiency, complicating established workflows.

As to **RQ2**, automating the commissioning workflow by capitalizing on MDE to reduce the time needed for commissioning tasks results in quicker project completion. This high degree of automation reduces human errors linked to manual configurations, improving the accuracy of the commissioning process. Figure 5 demonstrates that adapting BCS logic led to a 16.1% decrease in artificial lighting energy consumption following the implementation of the proposed modeling methodology. This illustrates that generating runtime artifacts from BIM models allows to meet operational requirements. In synopsis, model-driven automation streamlines commissioning and enhances building operations, evidenced by notable energy savings.

As to **RQ3**, our DSMLs facilitate real-time adjustments to BCS through post-occupancy evaluations and continuous monitoring, thus addressing actual occupancy patterns and environmental conditions. The resulting iterative process maintains alignment between occupant needs and operational goals. Figure 5 illustrates this by reporting on a 16.1% reduction in artificial lighting energy consumption following amendments to the BCS logic. This reduction demonstrates the efficacy of operational feedback loops in refining control strategies using real-world data from the post-occupancy phase. Thus, the integration of operational feedback mechanisms via DSMLs is crucial for continuous performance optimization in building operations as evidenced by our results.

B. Performance Evaluation

The performance evaluation looks at the scalability of Lua script generation and middleware configuration file export. Figure 6 presents the results. Lua script generation experiments show that execution time grows with the number of devices, actions, and nesting levels. For example, writing scripts for 1000 devices with 100 actions and five levels of nesting takes 1758.6 ms, versus 36.4 ms for a simple configuration (two devices, one action, no nesting). The rise is nonlinear, with increasing complexity affecting performance. The first test showed a chilly start effect, which was most likely caused by parser startup. Middleware configuration file export tests show that execution time scales with the number of devices and controllers while remaining more efficient than script creation, with 708 ms required for 1000 devices and 500 controllers. The tests were run on a Windows 11 Enterprise system with 16GB of RAM and an AMD Ryzen 7 PRO 5850U processor. The experiments were not run independently of other processes, ensuring realistic performance conditions.

C. Discussion and Positioning Within Related Work

Using MDE to improve BCS interoperability, development efficiency, and cost-effectiveness is not novel [48]. DSMLs offer specialized abstractions and frameworks to handle unique construction commissioning difficulties. Albataineh and Jarrah [49] used DSMLs to develop IoT-enabled models for energy optimization in smart buildings, enabling simulation and analysis of device interactions. Fazel and Wainer [50] presented a DSML for IoT-enabled infrastructure management, which focuses on managing dynamic systems and building automation components. These approaches highlight the importance of DSMLs in gathering domain-specific needs

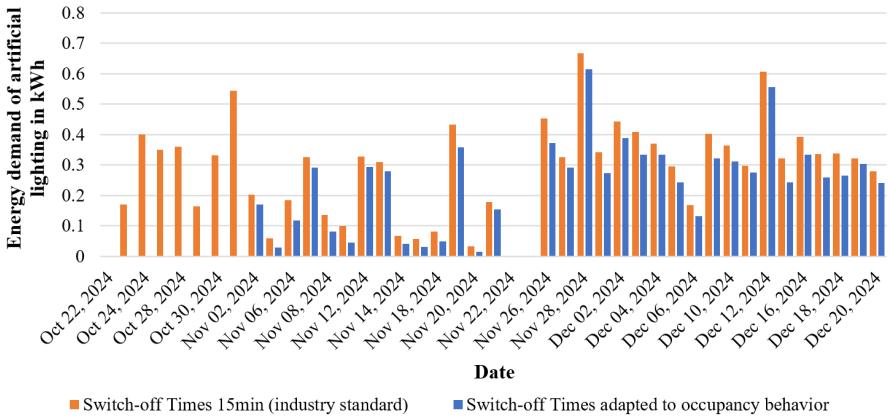

Fig. 5. Changes in energy consumption in the Living Lab after amending the logic of the BCS for meriting occupant patterns using the proposed modeling methodology and its tooling. The gap on Nov 22 is due to nobody having been at the office (cf. the Living Lab).

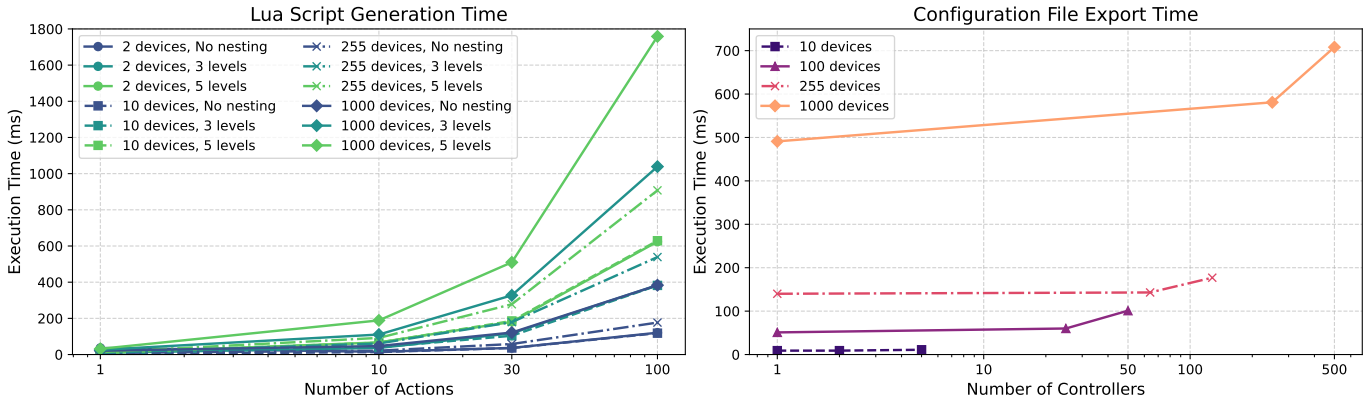

Fig. 6. Durations (in ms) for both generating Lua scripts and exporting middleware configurations. *Nesting* denotes the levels of nested conditional statements in control logic.

and facilitating organized model management during pre-commissioning. Integrating DSMLs with BIM can significantly improve building commissioning processes and operational efficiency. Traditional BIM tools are limited in their ability to manage the intricacies of BCS and dynamic building operations. Our DSML ecosystem incorporates operational requirements into design, creating a unified framework that addresses these limitations.

MDE frameworks can link building commissioning processes with DT workflows. Domingo et al. [51] showed that MDE approaches can increase consistency and minimize mistakes during commissioning by automating repetitive operations and aligning models with operational data. Zech et al. [18] developed a DSML to automate BCS commissioning within BIM. Their solution incorporates BCS data into BIM models, allowing for seamless collaboration, automatic deployment, and DT development. Lu and Olofsson [52] used MDE concepts and discrete-event simulations (e.g., DEVS formalism) to improve commissioning efficiency. The challenges of the commissioning process, such as inefficiencies and information silos (cf. Section II), highlight the importance of a model-driven approach. Our methodology leverages DSMLs to improve data flow and collaboration among stakeholders, reducing errors and misalignment between design and operational objectives. According to Chan et al. [12], successful BIM deployments rely on increased interoperability and stakeholder participation across the project and building life cycle. The importance of operational feedback loops is crucial. Using real-time data to refine BCS is vital for optimizing building performance, including energy efficiency and comfort. Our findings support Galasiu et al.'s findings on the effectiveness of sophisticated control systems, which show significant energy savings through feedback mechanisms [42]. Continuous improvement through feedback improves building efficiency and promotes sustainability by reducing energy usage.

MDE is particularly useful for developing IoT systems due to their complicated needs and diverse platform technologies. Using MDE to include non-functional requirements, such as interoperability and context-awareness, helps lessen disputes during early design and development [53]. MDE can enhance software quality evaluation for Low-Code platforms by tailoring current standards to meet IoT-specific requirements [54]. Kirchof et al. presented MontiThings, a model-driven infrastructure that improves IoT application dependability and efficacy by generating executable container images [55]. Corradini et al. [56] introduced Floware, a model-driven toolchain for de-

veloping IoT applications, especially for smart home contexts. Incorporating DTs into MDE frameworks offers new ways to analyze system behavior and improve troubleshooting [57], [58]. MDE has the potential to reduce development complexity for IoT applications, making it a viable area for future research.

MDE methodologies have also helped to improve the integration of BIM and DTs. Mittal and Martin's [59] framework aligns MDE with net-centric systems, focusing on automating and ensuring data consistency across complex infrastructures. Di Biccari et al. [60] investigated how MDE might improve BIM interoperability by facilitating data interchange and consistency checks during commissioning. These studies highlight the significance of MDE in bridging the gap between static BIM models and dynamic operational DTs during the commissioning stage. The inclusion of DTs in our proposal improves the overall functionality and adaptability of BCS. Our major goal is to enhance modeling, programming, and commissioning procedures. However, incorporating DTs provides valuable real-time monitoring and simulation of building performance. Using DTs helps bridge the gap between static design models and dynamic building operations, resulting in more efficient and responsive settings. DTs play a crucial role in improving the commissioning and operational efficiency of modern buildings, highlighting the importance of interoperability and collaboration among construction stakeholders [10].

Despite these developments, there are still numerous limitations in the use of DSMLs and MDE for building commissioning. Existing DSMLs frequently focus on individual parts of commissioning, like energy management or device control, without integrating models and data throughout the life cycle. MDE frameworks offer automation and consistency, but their use in collaborative commissioning workflows – at least for now – is limited, especially for buildings with BCS integration [10].

D. Sustainability Impact

The sustainability impacts can be measured through life cycle optimization and enhancements in energy efficiency. Projections suggest that the adoption of our modeling methodology may result in at least a 15% reduction in energy consumption in commercial buildings, consistent with ISO 50001 energy management standards [61]. The anticipated increase in the lifespan of equipment, including lighting systems, is attributed to reduced usage and optimized operational schedules, thus naturally extending their lifespan. This reduces replacement frequency and minimizes waste and resource consumption linked to manufacturing and disposal.

Aligning our approach with the European Green Building Council standards substantiates our sustainability claims, ensuring that the ecological benefits realized through our framework are acknowledged within the wider context of sustainable building practices. Quantifying these impacts allows for a strong argument in favor of our proposal, illustrating its capacity to deliver ecological advantages while improving the overall performance of building systems.

E. Practical Impact

As early as the design phase, assumptions have to be made about subsequent usage behavior, which can later lead to discrepancies between planning and operation. These so-called energy performance gaps often only become apparent after commissioning, as tests usually only ensure basic functionality but do not include energy performance checks. As user behavior has a significant influence on system performance, subsequent adjustments are just as important as initial commissioning in order to ensure long-term energy efficiency. This is illustrated by the adjustments to the switch-off times in the lighting in the study object. In view of the fact that inefficient building operation is currently a key challenge in the construction industry, the sector responsible for one third of global energy demand [62], supporting methods for subsequent system improvements are essential.

However, the large number of proprietary systems with specific protocols currently makes system adaptations difficult in practice and increases time and costs. The system architecture created with the middleware and the DSMLs offers the possibility of reducing the effort for executing instances such as facility management. A system-independent user interface for programming and configuring the control components proves to be valuable.

Commissioning is also often a complex, error-prone process in which not all influencing factors can be taken into account. The multiple transfer of control responsibility from the specialist planner to the contractor to facility management poses further challenges.

By integrating BCS into BIM, transparency and responsibilities can be improved. This facilitates energy-efficient adaptations and reduces expensive system changes. Supporting tools for automated adaptation to user behavior are essential to sustainably optimize comfort and efficiency.

VII. Lessons Learned

When realizing our approach, we learned the following lessons about modeling and organizational aspects.

A. Modeling Aspects

Integration Challenges. The integration of DSMLs with BIM tools remains problematic due to mismatched data structures and varying levels of support for automation across platforms. BIM models primarily serve architectural and structural purposes, while BCS models require more detailed operational logic. This mismatch leads to inefficiencies, such as manual remodeling, inconsistencies between design and implementation, and difficulties in synchronizing model updates [13], [10].

We recommend *adopting* a standardized, open data exchange format, such as Industry Foundation Classes (IFC), to ensure compatibility between BIM tools. This will reduce manual remodeling efforts and improve consistency across different phases of the building life cycle. In addition, we propose leveraging tool interoperability standards like the Open Services for Lifecycle Collaboration (OSLC) [63] to enable seamless integration across heterogeneous modeling and life cycle management tools.

Scalability. As building complexity increases, DSML-based models become harder to manage, leading to slow processing times and difficulties in maintaining system consistency. Larger building projects introduce exponentially more devices, connections, and operational dependencies, which can overwhelm

traditional modeling approaches. Without proper scalability measures, generation times (cf. Figure 6) increase significantly, making real-time updates and iterative optimization impractical.

We recommend *implementing hierarchical modeling* techniques to break large systems into modular, trade-specific sub-models, allowing different components to be managed independently while maintaining overall system coherence. This approach improves performance and makes it easier to scale DSML-based commissioning workflows.

Automation Efficiency. Model-driven automation promises efficiency gains, but errors in automatically generated runtime artifacts (e.g., configuration scripts, control scripts) might require manual intervention. The causing errors stem from inconsistencies between high-level models and the actual control system requirements, leading to misconfigured settings or incomplete logic. This reduces the expected benefits of automation and increases commissioning time.

We recommend *introducing automated validation* pipelines that check generated artifacts for correctness before deployment [64]. This reduces the need for manual corrections, ensuring that models align with operational expectations from the outset.

Feedback Loop Implementation. BIM and DSMLs often function as static representations, while real-world building operations are dynamic. Without an efficient mechanism for integrating live operational data back into the models, design assumptions remain unverified, and potential efficiency improvements (e.g., energy optimizations, occupancy-based adjustments) are missed.

We recommend *developing automated feedback* ingestion mechanisms that continuously update BIM models based on real-time operational data. This ensures that commissioning and control strategies remain adaptable to actual building performance.

B. Organizational Aspects

Stakeholder Collaboration. A major barrier to effective model-driven commissioning is the disconnect between architects, engineers, and facility managers [13]. Architects and engineers primarily focus on design and construction, while facility managers deal with operational realities. The lack of a shared framework for integrating operational concerns into early design decisions leads to misaligned priorities and costly post-construction modifications.

We recommend *establishing collaborative platforms* [65] where stakeholders can co-develop, exchange, and review BCS-augmented BIM models, ensuring alignment between design intent and operational requirements.

Training & Adoption Barriers. The adoption of DSMLs for building commissioning is slowed by a steep learning curve. Many industry professionals are unfamiliar with MDE concepts, making it difficult to implement new workflows. Additionally, the perceived complexity of DSML tools discourages users from transitioning away from manual methods.

We recommend *providing targeted training* programs and user-friendly tooling enhancements to lower the entry barrier and encourage wider adoption of model-driven commissioning approaches.

Regulatory & Standardization Gaps. Lack of standardization in how DSMLs integrate with BIM and BCS models makes it difficult to ensure compliance with industry regulations. Without clear regulatory frameworks, organizations risk creating proprietary solutions that are difficult to scale or share across projects.

We recommend *working towards industry-wide* standardization efforts that define best practices for model-based commissioning, ensuring regulatory compliance and cross-project compatibility.

Operational Resistance. Facility managers and building operators are often resistant to transitioning from traditional manual commissioning methods to automated, model-driven workflows. This resistance is driven by concerns over loss of control, lack of transparency in automated workflows, and fear of system failures due to incorrect model-generated artifacts.

We recommend *incorporating user-driven override* mechanisms within automated workflows, allowing facility managers to retain control while gradually building confidence in model-driven automation.

VIII. Conclusion

The integration of DSMLs with BIM represents a significant advancement in building commissioning and operational efficiency. Our research addresses gaps in current methodologies by providing a structured framework that improves BCS modeling, programming, and configuration while embedding operational requirements in the design phase.

Our results demonstrate potential for enhanced energy efficiency, reduced commissioning errors, and faster adaptation to changing operational needs. The ecological benefits -— including energy consumption reductions and extended equipment lifespans -— highlight our framework's contribution to sustainable building practices.

While our prototype implementation shows promise, we acknowledge limitations in BCS technology compatibility and scalability challenges. Future improvements focus on *(i)* addressing scalability concerns for large-scale implementations with numerous devices and complex control logic; *(ii)* integrating our framework with IoT platforms to enhance automated data collection, enable dynamic operational adjustments, and support a multitude of communication protocols; *(iii)* expanding DSML functionality to include additional use cases such as water management, waste management, and renewable energy systems; *(iv)* exploring the link to asset administration shells [66], [67] to reuse existing descriptions in the building context; *(v)* modularization of our DSMLs for application in further domains, e.g., manufacturing; and *(vi)* further TAM-based [18], [27] evaluations of our complete ecosystem of DSMLs and tooling support to assess their usability, acceptance, and efficiency and effectiveness.

This research establishes a foundation for future innovations in building design and operation, providing a framework for sustainable and efficient practices adaptable to evolving occupant and environmental requirements. As digital transformation continues in the construction industry, insights from this study will be crucial for advancing building management and operational performance.

References

[1] B. Succar, "Building information modelling framework: A research and delivery foundation for industry stakeholders," *Automation in construction*, vol. 18, no. 3, pp. 357–375, 2009.

[2] R. Sacks, C. Eastman, G. Lee, and P. Teicholz, *BIM handbook: A guide to building information modeling for owners, designers, engineers, contractors, and facility managers*. John Wiley & Sons, 2018.

[3] P. Zech, T. Clark, and R. Breu, "An Empirical Analysis of Digital Twin Adoption," in *Proceedings of 58th Hawaiin Internation Conference on System Sciences (HICSS'58)*, 2025, pp. 6253–6262.

[4] X. Kuai, Y. Liu, M. Bi, and Q. Luo, "Deciphering Building Information Modeling Evolution: A Comprehensive Scientometric Analysis across Lifecycle Stages," *Buildings*, vol. 13, no. 11, p. 2688, 2023.

[5] W. Kritzinger, M. Karner, G. Traar, J. Henjes, and W. Sihn, "Digital Twin in manufacturing: A categorical literature review and classification," *Ifac-PapersOnline*, vol. 51, no. 11, pp. 1016–1022, 2018.

[6] F. Tao, H. Zhang, A. Liu, and A. Y. Nee, "Digital twin in industry: State-of-the-art," *IEEE Transactions on industrial informatics*, vol. 15, no. 4, pp. 2405–2415, 2018.

[7] J. Michael, L. Cleophas, S. Zschaler, T. Clark, B. Combemale, T. Godfrey, D. Khelladi, V. Kulkarni, D. Lehner, B. Rumpe, M. Wimmer, A. Wortmann, S. Ali, B. Barn, I. Barosan, N. Bencomo, F. Bordeleau, G. Grossmann, G. Karsai, O. Kopp, B. Mitschang, P. Muñoz Ariza, A. Pierantonio, F. Polack, M. Riebisch, H. Schlingloff, M. Stumptner, A. Vallecillo, M. van den Brand, and H. Vangheluwe, "Model-Driven Engineering for Digital Twins: Opportunities and Challenges," *INCOSE Systems Engineering*, 2025, in press.

[8] M. K. Najjar, K. Figueiredo, A. C. J. Evangelista, A. W. Hammad, V. W. Tam, and A. Haddad, "Life cycle assessment methodology integrated with BIM as a decision-making tool at early-stages of building design," *International Journal of Construction Management*, vol. 22, no. 4, pp. 541–555, 2022.

[9] J. Michael, J. Blankenbach, J. Derksen, B. Finklenburg, R. Fuentes, T. Gries, S. Hendiani, S. Herlé, S. Hesseler, M. Kimm, J. C. Kirchhof, B. Rumpe, H. Schüttrumpf, and G. Walther, "Integrating models of civil structures in digital twins: State-of-the-Art and challenges," *Journal of Infrastructure Intelligence and Resilience*, vol. 3, no. 3, September 2024. [Online]. Available: https://doi.org/10.1016/j.iintel.2024.100100

[10] M. Hauer, S. Hammes, P. Zech, D. Geisler-Moroder, D. Plörer, J. Miller, V. van Karsbergen, and R. Pfluger, "Integrating Digital Twins with BIM for Enhanced Building Control Strategies: A Systematic Literature Review Focusing on Daylight and Artificial Lighting Systems," *Buildings*, vol. 14, no. 3, p. 805, 2024.

[11] P. Zech, C. Nardin, S. Ristov, M. Flora, and R. Breu, "Digital-Twins-as-a-Service in Construction Engineering," in *2024 IEEE 20th International Conference on Automation Science and Engineering (CASE)*. IEEE, 2024, pp. 3004–3010.

[12] D. W. Chan, T. O. Olawumi, and A. M. Ho, "Critical success factors for building information modelling (BIM) implementation in Hong Kong," *Engineering, Construction and Architectural Management*, vol. 26, no. 9, pp. 1838–1854, 2019.

[13] S. Hammes, D. Geisler-Moroder, J. Weninger, P. Zech, and R. Pfluger, "Market Demands vs. Scientific Realities: A Comparative Analysis in the Context of BIM-based and user-centred Lighting Control," *Developments in the Built Environment*, vol. 19, p. 100526, 2024.

[14] M. Völter, T. Stahl, J. Bettin, A. Haase, S. Helsen, and K. Czarnecki, *Model-Driven Software Development: Technology, Engineering, Management*. Wiley, 2013.

[15] A. W. Wymore, *Model-based systems engineering*. Boca Raton, FL, USA: CRC press, 2018, vol. 3.

[16] B. Combemale, R. France, J.-M. Jézéquel, B. Rumpe, J. Steel, and D. Vojtisek, *Engineering Modeling Languages: Turning Domain Knowledge into Tools*. Chapman & Hall/CRC Innovations in Software Engineering and Software Development Series, November 2016.

[17] M. A. Mohamed, M. Challenger, and G. Kardas, "Applications of model-driven engineering in cyber-physical systems: A systematic mapping study," *Journal of computer languages*, vol. 59, p. 100972, 2020.

[18] P. Zech, E. Goldin, S. Hammes, D. Geisler-Moroder, R. Pfluger, and R. Breu, "Model-Based Auto-Commissioning of Building Control Systems," in *Proceedings of the 26th International Conference on Enterprise Information Systems (ICEIS 2024)*, 2024, pp. 121–128.

[19] J. Beiter, "TwinLight - BIM-based implementation of daylight and artificial lighting controls," Online Access May 2024: https://nachhaltigwirtschaften.at/en/sdz/projects/twinlight.php, 2022.

[20] G. B. Ozturk, "Digital twin research in the AECO-FM industry," *Journal of Building Engineering*, vol. 40, p. 102730, 2021.

[21] R. Wieringa, *Design Science Methodology for Information Systems and Software Engineering*. Springer, 2014.

[22] P. Zech, "Artifacts for MODELS'25 Paper: An Ecosystem of DSMLs for Building Commissioning ," Aug. 2025. [Online]. Available: https://doi.org/10.5281/zenodo.16738343

[23] J. C. Mankins, *TECHNOLOGY READINESS LEVELS: A White Paper*. Nasa, 1995.

[24] A. Wąsowski and T. Berger, *Domain-Specific Languages*. Springer, 2023.

[25] A. Dreiling, M. Rosemann, W. Van Der Aalst, L. Heuser, and K. Schulz, "Model-based Software Configuration: patterns and languages," *European Journal of Information Systems*, vol. 15, no. 6, pp. 583–600, 2006.

[26] P. Zech, S. Hammes, E. Goldin, D. Geisler-Moroder, R. Breu, and R. Pfluger, "From BIM to Digital Twin: A transformation process through advanced control modeling and automated commissioning using daylight and artificial lighting as examples," *Energy and Buildings*, p. 115184, 2024.

[27] F. D. Davis *et al.*, "Technology Acceptance Model: TAM," *Al-Suqri, MN, Al-Aufi, AS: Information Seeking Behavior and Technology Adoption*, vol. 205, no. 219, p. 5, 1989.

[28] "Unreal Engine 5," https://www.unrealengine.com/unreal-engine-5, 2025, accessed: 2025-07-18.

[29] "Autodesk Revit," https://www.autodesk.com/products/revit/overview, 2025, accessed: 2025-07-18.

[30] "Eclipse Modeling Framework (EMF) Ecore," https://www.eclipse.org/modeling/emf/, 2025, accessed: 2025-07-18.

[31] "Xtext: Language Development Made Easy!" https://www.eclipse.org/Xtext/, 2025, accessed: 2025-07-18.

[32] "Xtend: Modernized Java for Eclipse Xtext," https://www.eclipse.org/xtend/, 2025, accessed: 2025-07-18.

[33] D. Bork and P. Langer, "Language Server Protocol: An Introduction to the Protocol, Its Use, and Adoption for Web Modeling Tools," *Enterprise Modelling and Information Systems Architectures (EMISAJ)*, vol. 18, pp. 9–1, 2023.

[34] A. Soleimanijavid, I. Konstantzos, and X. Liu, "Challenges and opportunities of occupant-centric building controls in real-world implementation: A critical review," *Energy and Buildings*, p. 113958, 2024.

[35] V. Garg and N. Bansal, "Smart occupancy sensors to reduce energy consumption," *Energy and Buildings*, vol. 32, no. 1, pp. 81–87, 2000.

[36] "gRPC: A high performance, open-source universal RPC framework," https://grpc.io/, 2025, accessed: 2025-07-18.

[37] "Apache Kafka," https://kafka.apache.org/, 2025, accessed: 2025-07-18.

[38] "MQTT: MQ Telemetry Transport," https://mqtt.org/, 2025, accessed: 2025-07-18.

[39] "MongoDB NoSQL Database," https://www.mongodb.com/, 2025, accessed: 2025-07-18.

[40] "EN 12464-1:2021 - Light and lighting – Lighting of work places – Part 1: Indoor work places," European Committee for Standardization (CEN), 2021, brussels, Belgium.

[41] A. Pandharipande and D. Caicedo, "Smart indoor lighting systems with luminaire-based sensing: A review of lighting control approaches," *Energy and Buildings*, vol. 104, pp. 369–377, 2015.

[42] C. S. Anca D. Galasiu, Guy R. Newsham and D. M. Sander, "Energy Saving Lighting Control Systems for Open-Plan Offices: A Field Study," *LEUKOS*, vol. 4, no. 1, pp. 7–29, 2007.

[43] M. S. R. Dorene Maniccia, Burr Rutledge and W. Morrow, "Occupant Use of Manual Lighting Controls in Private Offices," *Journal of the Illuminating Engineering Society*, vol. 28, no. 2, pp. 42–56, 1999.

[44] D.-G. J. Opoku, S. Perera, R. Osei-Kyei, M. Rashidi, K. Bamdad, and T. Famakinwa, "Digital twin for indoor condition monitoring in living labs: University library case study," *Automation in Construction*, vol. 157, p. 105188, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S092658052300448X

[45] H. Yoshino, T. Hong, and N. Nord, "IEA EBC annex 53: Total energy use in buildings—Analysis and evaluation methods," *Energy and Buildings*, vol. 152, pp. 124–136, 2017.

[46] S. Hammes, J. Weninger, D. Geisler-Moroder, R. Pfluger, and W. Pohl, "Reduction of the use of artificial light by adapting the switch-off times to individual presence patterns," *Bauphysik*, vol. 43, no. 1, pp. 50–64, 2021.

[47] S. Hammes, J. Weninger, and P. Zech, "Closer to Realtime: A Comparison of Machine Learning Methods to Reduce Artificial Switch-Off Times," in *2024 IEEE Sustainable Smart Lighting World Conference & Expo (LS24)*, 2024, pp. 1–4.

[48] B. Butzin, F. Golatowski, C. Niedermeier, N. Vicari, and E. Wuchner, "A model based development approach for building automation systems," in *Proceedings of the 2014 IEEE Emerging Technology and Factory Automation (ETFA)*. IEEE, 2014, pp. 1–6.

[49] M. Albataineh and M. Jarrah, "DEVS-based IoT management system for modeling and exploring smart home devices," in *2019 Sixth International Conference on Internet of Things: Systems, Management and Security (IOTSMS)*. IEEE, 2019, pp. 73–78.

[50] I. A. Fazel and G. Wainer, "A DEVS-Based Methodology for Simulation and Model-Driven Development of IoT," in *International Conference on Simulation Tools and Techniques*. Springer, 2023, pp. 3–17.

[51] Á. Domingo, J. Echeverría, O. Pastor, and C. Cetina, "Evaluating the benefits of model-driven development: empirical evaluation paper," in *Advanced Information Systems Engineering: 32nd International Conference, CAiSE 2020, Grenoble, France, June 8–12, 2020, Proceedings 32*. Springer, 2020, pp. 353–367.

[52] W. Lu and T. Olofsson, "Building information modeling and discrete event simulation: Towards an integrated framework," *Automation in construction*, vol. 44, pp. 73–83, 2014.

[53] M. R. Tabassum, "Addressing non-functional requirements of adaptive IoT systems: a model-driven approach," in *Proceedings of the 25th International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings*, 2022, pp. 195–200.

[54] F. Ihirwe, D. Di Ruscio, S. Gianfranceschi, and A. Pierantonio, "Assessing the Quality of Low-Code and Model-driven Engineering Platforms for Engineering IoT Systems," in *2022 IEEE 22nd International Conference on Software Quality, Reliability and Security (QRS)*. IEEE, 2022, pp. 583–594.

[55] J. C. Kirchhof, B. Rumpe, D. Schmalzing, and A. Wortmann, "Montithings: Model-driven development and deployment of reliable iot applications," *Journal of Systems and Software*, vol. 183, p. 111087, 2022.

[56] F. Corradini, A. Fedeli, F. Fornari, A. Polini, and B. Re, "FloWare: an approach for IoT support and application development," in *International Conference on Business Process Modeling, Development and Support*. Springer, 2021, pp. 350–365.

[57] J. C. Kirchhof, L. Malcher, and B. Rumpe, "Understanding and improving model-driven IoT systems through accompanying digital twins," in *Proceedings of the 20th ACM SIGPLAN International Conference on Generative Programming: Concepts and Experiences*, 2021, pp. 197–209.

[58] J. C. Kirchhof, J. Michael, B. Rumpe, S. Varga, and A. Wortmann, "Model-driven Digital Twin Construction: Synthesizing the Integration of Cyber-Physical Systems with Their Information Systems," in *23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems*. ACM, October 2020, pp. 90–101.

[59] S. Mittal and J. L. R. Martin, "Model-driven systems engineering for netcentric system of systems with DEVS unified process," in *2013 Winter Simulations Conference (WSC)*. IEEE, 2013, pp. 1140–1151.

[60] C. Di Biccari, F. Calcerano, F. D'Uffizi, A. Esposito, M. Campari, and E. Gigliarelli, "Building information modeling and building performance simulation interoperability: State-of-the-art and trends in current literature," *Advanced Engineering Informatics*, vol. 54, p. 101753, 2022.

[61] "ISO 50001:2018 - Energy Management Systems – Requirements With Guidance for Use," International Organization for Standardization, Geneva, Switzerland, 2018, accessed: 2025-07-18. [Online]. Available: https://www.iso.org/standard/69426.html

[62] U. Berardi, "A cross-country comparison of the building energy consumptions and their trends," *Resources, Conservation and Recycling*, vol. 123, pp. 230–241, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0921344916300489

[63] "Open Services for Lifecycle Collaboration (OSLC)," https://open-services.net/, 2025, accessed: 2025-07-18.

[64] P. Zech, P. Burger, S. Hammes, D. Geisler-Moroder, and R. Breu, "BIMReason: Validating BIM model correctness," *Bauphysik*, vol. 46, no. 6, pp. 332–339, 2024.

[65] P. Zech, P. Pobitzer, G. Fröch, and R. Breu, "A Proposal for a Models-Meet-Data Repository For Digital Twins in Construction Engineering," in *2024 IEEE 21st International Conference on Software Architecture Companion (ICSA-C)*. IEEE, 2024, pp. 111–118.

[66] "IEC 63278-1 Asset Administration Shell for industrial applications – Part 1: Asset Administration Shell structure," 2023. [Online]. Available: https://www.vde-verlag.de/iec-normen/252417/iec-63278-1-2023.html

[67] J. Zhang, C. Ellwein, M. Heithoff, J. Michael, and A. Wortmann, "Digital Twin and the Asset Administration Shell: An Analysis of 3 AASs Types and their Feasibility for Digital Twin Engineering," *Journal Software and Systems Modeling (SoSyM)*, 2025.