buildings

MDPI

Article

Development of an FMI-Based Data Model to Support a BIM-Integrated Building Performance Analysis Framework

ByungChan Kong and WoonSeong Jeong *

Department of Architectural Engineering, Chungbuk National University, Cheongju 28644, Republic of Korea; skckwdk@chungbuk.ac.kr
* Correspondence: wsjeong@chungbuk.ac.kr; Tel.: +82-43-261-2432

Abstract

The lack of modularity in building design information within multi-domain building performance analysis environments impedes efficient multidisciplinary analysis during the building design process. This study proposes a Functional Mock-up Interface (FMI)-based data model to facilitate the translation of building design information into a Building Information Modeling (BIM)-integrated building performance analysis framework that can be seamlessly integrated with object-oriented physical models. The proposed data model employs both FMI and BIM to decouple the design information required for physics-based analysis from existing Building Information Models. It then generates a physical BIM-based Functional Mock-up Unit (PBIM-FMU), which encapsulates the necessary building design information and can operate independently within a multi-domain building performance analysis environment. The PBIM-FMU can be readily interfaced with object-oriented physical modeling (OOPM)-based analysis models, as demonstrated in this study through its integration with an OOPM-based thermal analysis model for estimating annual building energy demand. To validate the proposed framework, simulation results from a manually constructed thermal analysis model were compared with those from a model integrated with the PBIM-FMU. The results were consistent, confirming that the data model supports accurate data exchange between BIM and multi-domain building performance simulation platforms.

Keywords: Building Information Modeling; Functional Mock-up Interface; building performance analysis; Modelica; object-oriented modeling

check for updates

Academic Editor: Ahmed Senouci

Received: 8 August 2025
Revised: 29 August 2025
Accepted: 3 September 2025
Published: 5 September 2025

Citation: Kong, B.; Jeong, W. Development of an FMI-Based Data Model to Support a BIM-Integrated Building Performance Analysis Framework. *Buildings* **2025**, *15*, 3200. https://doi.org/10.3390/buildings15173200

1. Introduction

The construction industry plays a pivotal role in national economic development through infrastructure and urban expansion; however, it has long faced criticism for its lack of environmental sustainability. According to the International Energy Agency [1], the global construction sector accounts for approximately 30% of final energy consumption and contributes nearly 40% of energy-related carbon emissions. Moreover, energy use and carbon emissions in the construction industry are projected to continue rising through 2050 [2]. Against this backdrop, green and low-energy building technologies have emerged as effective strategies to reduce environmental burdens—such as energy consumption, greenhouse gas emissions, and waste generation—and to promote sustainable development in the construction sector [3]. Extensive research and development efforts have been devoted to applying these technologies from the early design stages of buildings [4].

In this context, a variety of building performance analysis tools are now essential in the design phase to evaluate design alternatives and optimize building performance [5]. However, there remains a lack of integrated methods that can efficiently support multidisciplinary performance analyses during the design stage. As a result, significant time and effort are often required to reflect comprehensive performance analysis results in the building design.

To address this challenge, numerous studies have focused on improving the exchange of information between building design data and analytical models. These efforts have facilitated the efficient input of design information required for model generation and supported the rapid and accurate development of analytical models [6,7]. Although extensive literature reviews and technical developments on data exchange methods have been conducted [8,9], most existing approaches have focused on exchanging information for a single performance domain—primarily energy performance. In future building projects, however, integration of information across diverse performance domains such as structural integrity and cost estimation will be increasingly necessary. Yet studies that holistically address such comprehensive integration remain scarce.

Currently, over 500 energy performance analysis tools are utilized during the design phase [5], and this number is expected to grow substantially when extended to other domains such as structural performance [10]. Accordingly, considerable time and effort are required to define appropriate data exchange methods tailored to the specific data processing requirements and rules of each analysis tool. At the same time, the mathematical and computational representations of analytical models continue to evolve with advancements in mathematics, computer science, and engineering [11], necessitating continuous updates and maintenance of existing data exchange frameworks as models are revised.

Conventional data exchange approaches are limited in their ability to effectively integrate design information with new performance domains and require substantial resources for maintenance and management. Therefore, there is a pressing need for research that enables sustainable data exchange between building design information and multidisciplinary analytical models—taking into account the expansion of performance domains and the ongoing evolution of analytical tools. Such efforts will ultimately facilitate the efficient extraction and integration of comprehensive performance analysis information during the design phase.

The objective of this study is to develop a framework that enables efficient data exchange between building design information and multidisciplinary building analysis models. The proposed framework transforms building design information into a simulation model using a simulation data exchange standard, thereby facilitating the seamless transmission of information to various analysis models. The design data, once converted according to the standard, can be readily integrated with analysis models within the simulation environment. The framework was validated by evaluating whether the building design information could be accurately delivered to an object-oriented Building Energy Model (BEM). The resulting BEM model, generated through the proposed framework, successfully produced accurate simulation results such as building energy consumption.

2. Literature Reviews

To support efficient multidisciplinary building performance analysis during the design phase, an integrated methodology has been proposed that leverages Building Information Modeling (BIM), Object-Oriented Physical Modeling (OOPM), and simulation data exchange standards. OOPM techniques and simulation exchange standards were developed to facilitate efficient performance analysis across multiple domains and have been integrated with BIM to enhance the effectiveness of performance evaluation during the design stage [12].

A methodology for integrated building performance analysis based on data exchange between Building Information Models (BIM models) and Object-Oriented Physical Models (OOPM models) was presented as one of the key outcomes of the Annex 60 project [12]. The objective of the Annex 60 project was to develop next-generation simulation tools for the integrated design and operation of buildings and district energy systems. Additionally, a similar approach utilizing BIM and simulation exchange standards was discussed in the FMI4BIM project [13], which was compared with the Annex 60 framework to analyze the strengths and limitations of each methodology.

This study conducts a comprehensive review of the data exchange mechanisms proposed in these projects and other related prior research. The analysis reveals that existing performance analysis methods using OOPM and simulation exchange standards lack tight integration with BIM, which hinders efficient multidisciplinary performance evaluation in the early design stages. To address this limitation, this section also outlines a novel approach to data exchange aimed at improving integration between BIM and OOPM-based performance models.

2.1. Object-Oriented Physical Modeling and Simulation Data Exchange Standards for Sustainable Design and Building Energy Analysis

To enable efficient multidisciplinary building performance analysis, integrated simulation methodologies and standardized data exchange protocols have been widely adopted [14,15]. While these two approaches operate via different mechanisms, they both support the implementation of diverse domain-specific models and the execution of integrated simulations within a unified simulation environment.

OOPM is a modeling technique that abstracts physical systems into organized component-based structures [16]. The Modelica language, an open-standard, object-oriented language, is specifically designed to represent equation-based dynamics of complex physical systems [17]. OOPM models developed using Modelica express system topologies through interconnected components represented by differential-algebraic equations [14]. This approach facilitates model comprehension, supports component reuse, and enables flexible system development [14,18]. These models can be developed and simulated in Modelica-based platforms such as Dymola [19] and are widely used across engineering disciplines—including electrical, mechanical, thermal, and control systems. In the building domain, they are applied to thermodynamic modeling of individual buildings and district-level energy systems [9,11], with Modelica libraries such as Buildings [20] and IDEAS [21] being developed.

While Modelica-based modeling is well-suited for multidisciplinary physical simulation, converting all existing tools into the Modelica ecosystem is limited by economic and licensing constraints [12]. Furthermore, Modelica is not designed to represent partial differential equation-based models [22]. To overcome these limitations, real-time co-simulation approaches have been proposed that integrate heterogeneous simulation tools [12,22,23].

Co-simulation enables real-time data exchange between simulation components during execution. The Functional Mock-up Interface (FMI) standard defines an interface for such exchanges among heterogeneous tools [15]. Through FMI, a Functional Mock-up Unit (FMU)—an independent simulation module—can be generated and used across different simulation platforms. FMUs facilitate the integration of Modelica models with non-Modelica systems. In the building domain, FMI has been successfully used for co-simulation of building envelope and HVAC systems [11]. This hybrid simulation approach combining FMI and Modelica enables efficient and flexible coupling of multidisciplinary simulation tools [13,23].

2.2. Integrated Building Performance Analysis Using BIM, OOPM, and FMI Standards

For OOPM and FMI-based building performance analysis to be effectively applied during the design phase, seamless integration with the building design process is essential [12,24]. Accordingly, Modelica-based OOPM models have been developed to mirror the object-oriented structure of building elements [14,20], enabling potential interoperability with BIM models [14,24,25].

In this context, various studies have proposed methods for generating Modelica models directly from BIM and automating or semi-automating the exchange of information between the two [7,24–31]. Additionally, other studies have investigated the application of the FMI standard to facilitate information exchange between BIM and domain-specific simulation tools [13,23].

Table 1 classifies the reviewed literature according to their data exchange approach, simulation tools used, representation format of building information, target simulation model (including Modelica-based tools), and the scale of development.

Table 1. Summary of data exchange methods between BIM and multidomain building performance.

Author.	Year	Building Data Format/Input	Data Exchange Method/Tool	Target Modelica Library or Fledged Tool/Output	Development Level		
					Building Geometry	Material	HVAC
Kim et al. [25]	2015	Revit Model	ModelicaBIM	Buildings	Developed	Developed	N/A
Remmen et al. [26]	2015	IFC	SimModel	AixLib	Developed	N/A	Developed
Thorade et al. [27]	2015	IFC	SimModel	AixLib, BuildingSystems	Developed	N/A	Developed
Jeong et al. [24]	2016	Revit Model	ModelicaBIM	Buildings	Developed	Developed	N/A
Nicolai and Paepcke [23]	2017	Implicitly BIM-based	N/A	Exported FMU from NANDRAD	N/A	N/A	N/A
Andriamamonjya et al. [28]	2018	IFC	Ifc2Modelica	IDEAS	Developed	Developed	Developed
Nytsch-Geusen et al. [29]	2019	IFC	BIM2Modelica	BuildingSystems	Developed	Developed	N/A
Seidenschnur et al. [30]	2022	Revit Model	VC CDE (FSC exporter)	FSC object model	N/A	N/A	N/A
Jansen et al. [31]	2023	IFC	BIM2SIM	AixLib, HKESim	N/A	N/A	Developed
Visby Fjerbæk et al. [7]	2023	Revit Model	VC CDE	Customized Buildings	N/A	N/A	Developed

2.3. Limitations of Existing Data Exchange Methods

2.3.1. Classification by Input Building Information Type

As summarized in Table 1, existing data exchange methods can be categorized into two types based on the format of input building information: (1) methods utilizing the Industry Foundation Classes (IFC) standard, and (2) methods directly using BIM authoring tools such as Revit [32].

The IFC-based approach involves exporting IFC files from a BIM authoring tool, extracting the relevant object properties needed for performance analysis, and mapping them directly to the parameters of a Modelica model [28]. Another example includes reading IFC files directly in a simulation tool to generate an FMU for performance analysis [13,23].

However, these methods suffer from several limitations. First, the exported IFC files may be incomplete due to limitations in the authoring tool. Second, updates to the IFC standard are not always promptly implemented by software vendors. Third, the IFC schema itself may not fully support the representation of all building components and their associated information [31,33]. As a result, effective and reliable data exchange remains a challenge.

In contrast, methods that directly utilize BIM authoring tools are more dependent on specific vendors but offer significant advantages. These methods can preserve the object relationships defined within the BIM environment and enable bidirectional data exchange with Modelica models [24,25]. In other words, they not only allow for the efficient transfer of building design information into analytical models but also enable simulation results to be reintegrated into the BIM environment with ease.

2.3.2. Dependency on Modelica Libraries in Data Exchange Methods

Data exchange methods based on the IFC standard have been criticized for their lack of flexibility when the underlying Modelica libraries are updated or structurally modified [30,31]. Similarly, FMI-based methods are sensitive to changes in the data structure that may result from updates to domain-specific simulation tools. This issue also arises in BIM authoring tool-based approaches, as observed in studies by Kim et al. [25] and Jeong et al. [24].

For instance, Jeong et al. [24] developed a method for exchanging data between Revit and a Modelica-based building energy model using the Buildings library. In their framework, Revit's architectural room objects were mapped to the *Buildings.Rooms.MixedAir* model, which was available in version 1.3.0 of the library. However, starting with version 4.0.0, this model was restructured and renamed to *Buildings.ThermalZones.Detailed.MixedAir*, which now inherits from *Buildings.ThermalZones.Detailed.BaseClasses.RoomHeatMassBalance* and *MixedAirHeatMassBalance* [34].

Such structural changes in the Modelica library significantly affect the parameter mapping and require a complete redefinition of the data linkage mechanism. Given the unpredictable nature of library evolution, data exchange frameworks that heavily rely on specific Modelica libraries face inherent limitations. Therefore, there is a growing need for flexible, library-independent data exchange mechanisms that can adapt to changes in modeling environments without requiring extensive rework.

2.3.3. Incompatibility of Intermediate Data Models Between BIM and Modelica Libraries

Remmen et al. [26] and Thorade et al. [27] proposed a method for extracting building geometry and HVAC system properties required for performance simulation from IFC files, converting them into an intermediate data model called SimModel, and then generating flexible Modelica libraries using a template-based code generation tool called CoTeTo. CoTeTo enables users to add new Modelica model content without altering the internal structure of the tool [26]. However, generating new model content requires the definition of additional mapping rules, which necessitates a deep understanding of both the SimModel structure and the Modelica language.

Although the SimModel was designed to be independent of specific Modelica libraries, the absence of predefined mapping rules in CoTeTo limits its usability. In such cases, the building design information embedded in SimModel cannot be transferred to a Modelica-based simulation environment, since SimModel is not a Modelica model nor inherently compatible with Modelica's structure. Furthermore, the semantic content of the design data in SimModel is insufficient for direct use in analytical model generation. These limitations have been reiterated in follow-up studies such as Nytsch-Geusen et al. [29] and Jansen et al. [31].

Similarly, Seidenschnur et al. [30] and Visby Fjerbæk et al. [7] extracted HVAC system information from Revit and converted it into a data model called the FSC object model, which was then transmitted via a web-based Common Data Environment (CDE) to support microservices. These microservices performed tasks such as HVAC integrity checks, airflow calculations, and statistical analysis. A template-based code generation tool was also used to convert the FSC object model into Modelica-compatible content [7,35]. However, like SimModel, the FSC object model is not inherently compatible with Modelica. Implementing new Modelica components through this approach requires prior knowledge of the FSC object model structure and programming concepts.

In conclusion, while intermediate data models independent of Modelica libraries are necessary for flexible data exchange between BIM and simulation environments, current models lack compatibility with Modelica's modular architecture. As a result, changes in Modelica libraries often necessitate the complete redefinition of mapping structures. The lack of modularity and interoperability in these intermediate data models poses a significant barrier to the seamless and efficient transfer of building design information into multidisciplinary simulation models.

2.3.4. Development of a Data Exchange Method Between BIM and Modelica Libraries Using the FMI Standard

This study proposes a framework that enables flexible integration between BIM models and Modelica libraries by converting the physical building information in a BIM model into a Functional Mock-up Unit (PBIM-FMU) in accordance with the FMI standard. The PBIM-FMU encapsulates building design information—such as walls, roofs, floors, windows, and rooms—which is transformed into simulation-domain entities such as exterior/interior surfaces, constructions, and boundary conditions. Importantly, this information retains the original data structure and semantics of the BIM model, thereby allowing for intuitive linkage between BIM and Modelica libraries.

The proposed framework defines mapping rules between the BIM model (authored in Revit) and the Modelica library based on the FMI standard, enabling automated generation of PBIM-FMUs. As a result, the PBIM-FMU functions not only as a data model but also as a stand-alone component that can be called within a Modelica-based simulation environment. It also supports seamless data exchange with heterogeneous FMUs and other Modelica models.

In this study, the PBIM-FMU was integrated with the Buildings library in a Modelica-based simulation environment and was used to estimate annual building energy consumption.

3. Framework Development

3.1. Methodology and Tools and Data for the Development of the Framework

This section presents a framework designed to efficiently support multidisciplinary building performance analysis during the design phase by generating a PBIM-FMU from a BIM model and integrating it with a Modelica library. The development process of the framework consists of two main stages: (1) process modeling and (2) system implementation. In the first stage, the process model is defined by organizing all required activities in a structured sequence, along with their respective input/output data and data flow. In the second stage, each activity is implemented as a system interface based on the defined process model, and a user interface (UI) is developed to enable user interaction. Additionally, the framework specifies data exchange mechanisms and algorithms required for integration across heterogeneous systems.

The following tools and datasets were used to develop the framework. First, the study assumes that all design data required for physics-based building performance analysis (e.g., building envelop geometry, material properties, thermophysical attributes) are pre-defined in the BIM model. Autodesk Revit 2024 was used as the BIM authoring tool to input the building envelope geometry, material properties, and thermophysical attributes. If such data are incomplete in the BIM model, the framework can incorporate a preprocessing stage to enrich the model [24]. This may include user-defined assignment of additional parameters to material objects and automatic calculation of derived values such as the window-to-frame area ratio.

The BIM model was designed based on the concept of Physical Building Information Modeling (PBIM), which extends the standard BIM data structure to incorporate information needed for physical simulation [24]. The PBIM data structure was implemented using the Revit Application Programming Interface (API), and the C# programming language [36] was employed to access the parametric BIM model directly and generate the PBIM-FMU. This approach minimizes the transformation steps between the BIM model and the analytical model [24].

Since Modelica-based BEM models reflect the physical system structure, intuitive data mapping is possible between BIM elements and Modelica components. This study defines the mapping rules between BIM models and the Buildings library based on the FMI standard, referencing multiple versions of the library. For example, building components representing architectural spaces in the BIM model are mapped to boundary conditions that define thermal zones in Modelica.

The PBIM-FMU was designed as a FMU of the Model Exchange type. This format supports discrete equations, algebraic equations, and differential equations involving time, state, and discrete events [37]. Although it does not include its own numerical solver, it relies on the simulation environment's solver, ensuring consistent simulation results [13]. Importantly, the Model Exchange FMU allows not only input/output variables but also parameter values to be exchanged with other simulation models during execution [13]. In building simulation models, building design information used as input data generally remains constant during simulation and can be specified by the user prior to execution. In certain cases—such as when employing a Model Exchange FMU—these values may also be reassigned by external systems during simulation. The building design information is defined in the simulation model as a variable type called a parameter, which represents the characteristics of the building. Accordingly, the proposed framework extracts design parameters from the BIM model and converts them into parameters of the Model Exchange FMU (PBIM-FMU). Because both the PBIM-FMU and the Buildings library define these variables as parameters, their data types are compatible. This compatibility allows the PBIM-FMU's parameter values to be transferred to the parameters of the Buildings library during simulation, where the numerical solver of the Buildings library performs the simulation.

3.2. Development of the Process Model

The process model defines the procedural flow and the essential information required for the development of the proposed framework. To construct this model, a series of activities (termed Activities) were specified to represent system functions, and the information required for and generated by each activity—as well as their interrelationships—was systematically organized.

The process model was developed using the IDEF0 (Integration Definition for Function Modeling) methodology, as defined by the International Organization for Standardization (ISO) [38]. In IDEF0, each system function is represented as a labeled box, with inputs shown on the left, outputs on the right, constraints on the top, and mechanisms on the bottom. Arrows indicate the direction of data or control flow. The sequential order and dependency among activities are also expressed using connecting arrows. Figure 1 presents the proposed process model designed using the IDEF0 formalism.

The process model consists of the following three major activities: (A1) extraction of building design information for energy simulation, (A2) generation of PBIM-FMU, and (A3) generation of an FMI-based OOPM model.

- **A1: Extraction of Building Design Information**
  This activity extracts the physical design information required for generating a BEM model from the BIM model (Revit). To do so, physical properties not originally defined in the BIM data schema were additionally specified, and functionality was developed to input and access these properties within the BIM model.
- **A2: Generation of PBIM-FMU**
  Based on the data obtained in A1, this activity generates a PBIM-FMU. The PBIM-FMU structure is designed to preserve the semantics and data structure of the original BIM model. Its internal variables are defined according to the FMI standard so that they are semantically compatible with the corresponding variables in the Modelica library (in terms of variable type, causality, variability, etc.).
- **A3: Generation of an FMI-Based OOPM Model**
  In this activity, the PBIM-FMU and the Modelica library are coupled using a Modelica-based simulation environment to create an FMI-compliant OOPM model, which serves as the integrated building performance simulation model. This model enables the estimation of building energy performance through simulation execution.

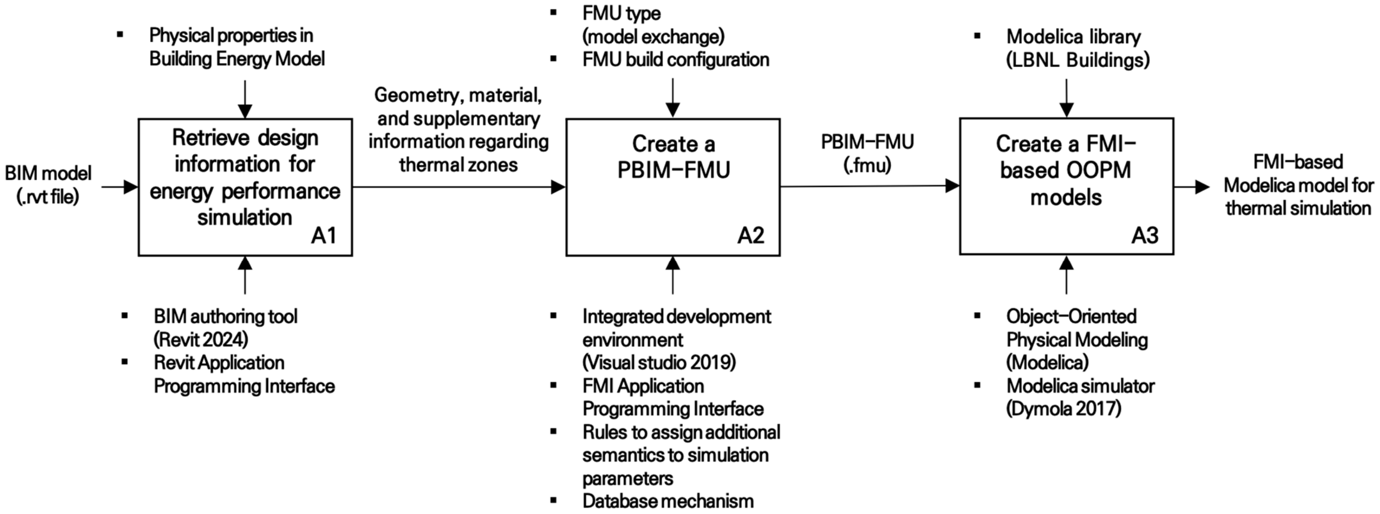

▪ Physical properties in Building Energy Model
BIM model (.rvt file)
Retrieve design information for energy performance simulation
A1
▪ BIM authoring tool (Revit 2024)
▪ Revit Application Programming Interface
Geometry, material, and supplementary information regarding thermal zones
▪ FMU type (model exchange)
▪ FMU build configuration
Create a PBIM−FMU
A2
▪ Integrated development environment (Visual studio 2019)
▪ FMI Application Programming Interface
▪ Rules to assign additional semantics to simulation parameters
▪ Database mechanism
PBIM−FMU (.fmu)
▪ Modelica library (LBNL Buildings)
Create a FMI−based OOPM models
A3
▪ Object−Oriented Physical Modeling (Modelica)
▪ Modelica simulator (Dymola 2017)
FMI−based Modelica model for thermal simulation

Figure 1. IDEF0-based process model illustrating the overall workflow of the proposed framework.

4. Framework Implementation

The functionality of the proposed framework was defined through a process model consisting of a series of activities. This process model was utilized as a procedural blueprint for implementing the framework. Each activity defined within the model was implemented as a system interface, with the corresponding algorithms developed under specific execution environments and system conditions. Furthermore, the communication mechanisms between heterogeneous activities were explicitly defined.

4.1. Implementation of the Building Design Information Extraction System for Energy Simulation (Activity A1)

Figure 2 shows the BIM model created using Autodesk Revit. The model represents a single architectural space (Room) composed of four exterior walls, one floor, and one roof. It contains material information (e.g., gypsum board, wood cladding, glass fiber, concrete) along with corresponding physical properties, including thermal conductivity, specific heat capacity, density, and thickness. Location and geographical information required to derive climate inputs for the simulation—for example, site latitude, longitude, and elevation—are not included in the BIM model. Therefore, they are assumed to be provided by the simulation environment.

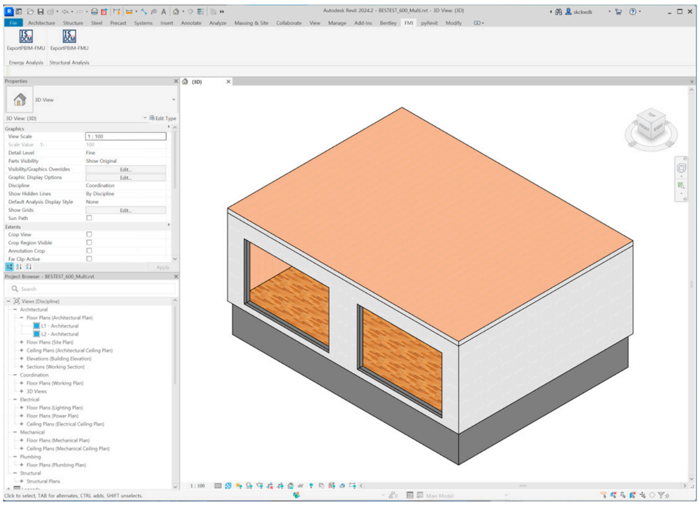


Figure 2. A Revit-based BIM model containing the necessary design information for thermal performance analysis.

Figure 3 presents the data structure of the PBIM model using a Unified Modeling Language (UML) class diagram, following the standards defined by the Object Management Group [39]. This class diagram represents the building design information required for BEM generation and the relationships between such data at the class level. In particular, the AddProperty class defines new physical attributes and includes the AddPhysicalProperty function, which allows the insertion of such properties into the BIM model. The defined UML class structure served as the foundation for designing and implementing the system interface, which facilitates access to the BIM model and enables the efficient extraction of building design information necessary for generating the BEM.

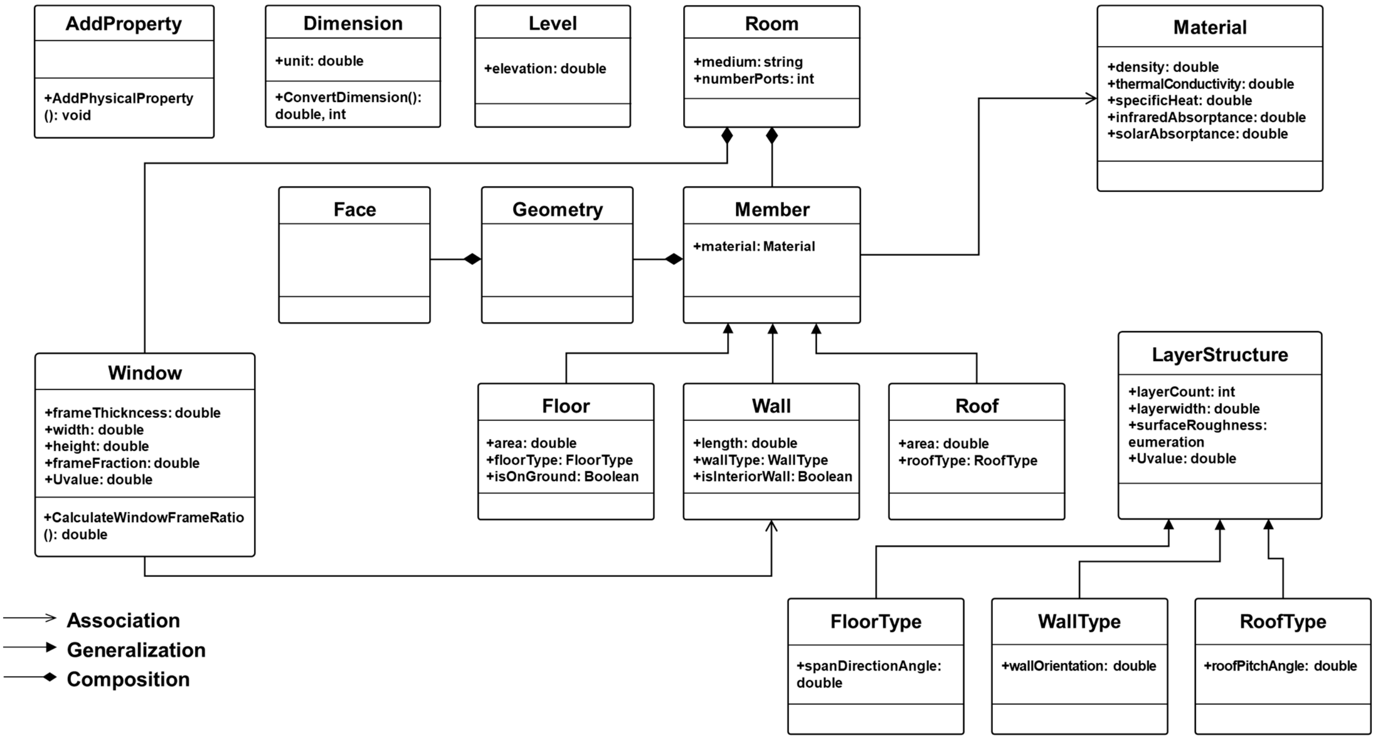


Figure 3. Class diagram illustrating data structure for Physical BIM (PBIM) representation.

Building upon this structure, additional physical attributes essential for generating a BEM—such as infrared absorptance, solar absorptance, indoor air modeling parameters, and frame thickness for window-to-frame area ratio—were incorporated. Since these attributes are not defined within Revit's data schema, new object types and attributes were implemented through the Revit API, thereby extending the BIM model into a PBIM model that integrates theses additional parameters.

4.2. Implementation of the PBIM-FMU Generation System (Activity A2)

4.2.1. Definition of PBIM-FMU Variable Semantics

The variables in the PBIM-FMU were defined according to the FMI standard using the same semantics as the corresponding variables in the Buildings library. This ensures that variable values in the PBIM-FMU can be transferred to the building envelope models provided by the Buildings library during simulation. To achieve semantic consistency between variables in the PBIM-FMU and those in the Buildings library, data mapping relationships were established between BIM objects and the Buildings library. These relationships enable the automated generation of the PBIM-FMU directly from the BIM model.

Table 2 presents the semantics of the variables comprising the PBIM-FMU, along with the BIM objects used to derive each variable. The "Model Variable" column specifies five semantic attributes: Variable Type, Causality, Variability, Initial (value type), and Declared Type. Table 3 presents the data mapping relationships between BIM objects and the Buildings library. The data mapping relationships are classified into three categories: Application, Calculation, and Translation.

Application refers to cases where the variable value extracted from a building component can be directly assigned to a corresponding variable in the simulation model provided by the Buildings library without any additional processing. For example, the thermal conductivity value of a material can be directly applied to the corresponding thermal conductivity variable in the simulation model, as no further computation or rule-based processing is required.

Calculation refers to cases where the variable value cannot be directly obtained from the building component but can be derived through simple arithmetic operations. For instance, the total number of material layers can be calculated by extracting the individual layer information and summing them arithmetically. The result can then be assigned to the corresponding variable (i.e., number of material layers) in the simulation model.

Translation refers to cases where the variable value cannot be assigned to the simulation model variables either directly or through simple calculations. In this case, rule-based translation grounded in domain knowledge (e.g., building physics, building energy simulation) is required. For example, building components such as walls, roofs, and floors must be translated into the corresponding thermal boundary conditions within the simulation model.

Since building design information represents physical properties in the BEM and affects the simulation outcome while remaining constant during execution, the Causality was set to parameter. Variability was set to fixed, indicating that the value remains unchanged after the simulation is initialized. To ensure simulation accuracy, the Initial attribute was set to exact, meaning that an exact value, not an estimated one, is provided to the simulation tool.

In particular, the roughness variable, which represents the surface roughness condition, was defined as an enumerated data type based on the FMI standard. It includes six constants—VeryRough, Rough, MediumRough, MediumSmooth, Smooth, and VerySmooth—to comply with the SurfaceRoughness data type used in the Buildings library.

Table 2. Semantic definition of physical property variables in the PBIM-FMU.

PBIM-FMU Property Name	Model Variable					Related BIM Class or Property Name
	Variable Type	Causality	Variability	Initial	Declared Type	
nConExtWin	Integer	parameter	fixed	exact	N/A	Member.Wall, Window
nConExt	Integer	parameter	fixed	exact	N/A	Member.Wall, Member.Roof
nConPar	Integer	parameter	fixed	exact	N/A	Member.Wall, Member.Wall.isInteriorWall
nSurBou	Integer	parameter	fixed	exact	N/A	Member.Wall, Room
nConBou	Integer	parameter	fixed	exact	N/A	Member.Floor, Level.elevation, Room
areaWall	Real	parameter	fixed	exact	N/A	Member.Wall.length, Level.elevation
areaFloor	Real	parameter	fixed	exact	N/A	Member.Floor.area
areaRoof	Real	parameter	fixed	exact	N/A	Member.Roof.area
azimuthWall	Real	parameter	fixed	exact	N/A	WallType.wallOrientation
tiltWall	Real	parameter	fixed	exact	N/A	Member.Wall, Face
tiltFloor	Real	parameter	fixed	exact	N/A	FloorType.spanDirectionAngle
azimuthRoof	Real	parameter	fixed	exact	N/A	Member.Roof, Face
tiltRoof	Real	parameter	fixed	exact	N/A	Member.Roof, Member.Roof.roofPitchAngle
infraredAbsorptance	Real	parameter	fixed	exact	N/A	Material.infraredAbsorptance
solarAbsorptance	Real	parameter	fixed	exact	N/A	Material.solarAbsorptance
roughness	Enumeration	parameter	fixed	exact	SurfaceRoughness	LayerStructure.surfaceRoughness
layerCount	Integer	parameter	fixed	exact	N/A	LayerStructure.layerCount
thickness	Real	parameter	fixed	exact	N/A	LayerStructure.layerWidth
thermalConductivity	Real	parameter	fixed	exact	N/A	Material.thermalConductivity
specificHeat	Real	parameter	fixed	exact	N/A	Material.specificHeat
density	Real	parameter	fixed	exact	N/A	Material.density
widthWindow	Real	parameter	fixed	exact	N/A	Window.width
heightWindow	Real	parameter	fixed	exact	N/A	Window.height
fractionFrame	Real	parameter	fixed	exact	N/A	Window.frameFraction
UvalueFrame	Real	parameter	fixed	exact	N/A	Window.Uvalue
heightRoom	Real	parameter	fixed	exact	N/A	Level.elevation
nPort	Integer	parameter	fixed	exact	N/A	Room.numberPorts
Medium	String	parameter	fixed	exact	N/A	Room.medium

Table 3. Definition of data mapping relationships between BIM objects and Buildings library.

PBIM-FMU Property Name	Related BIM Class or Property Name	Data Exchange Method	Exchange Mechanism
nConExtWin	Member.Wall, Window	Translation	Count the number of walls that serve as hosts for windows.
nConExt	Member.Wall, Member.Roof	Translation	Count the number of walls without windows and the number of roofs.
nConPar	Member.Wall, Member.Wall.isInteriorWall	Translation	Count the number of walls with the isInteriorWall property set to true.
nSurBou	Member.Wall, Room	Translation	Count the number of walls that bound a room but are adjacent either to the exterior environment or to spaces not defined as separate rooms.
nConBou	Member.Floor, Level.elevation, Room	Translation	Count the number of floors that separate two defined rooms (zones) at adjacent levels.
areaWall	Member.Wall.length, Level.elevation	Calculation	Calculate the area by multiplying the wall length by the room height.

Table 3. *Cont.*

PBIM-FMU Property Name	Related BIM Class or Property Name	Data Exchange Method	Exchange Mechanism
areaFloor	Member.Floor.area	Application	Extract the area directly from the floor's area attribute.
areaRoof	Member.Roof.area	Application	Extract the area directly from the roof's area attribute.
azimuthWall	WallType.wallOrientation	Application	Extract the angle (wallOrientation attribute) directly from the WallType object.
tiltWall	Member.Wall, Face	Calculation	Calculate the angle between the normal of the exterior side face of the wall and the vertical axis.
tiltFloor	FloorType.spanDirectionAngle	Application	Extract the angle (spanDirectionAngle attribute) directly from the FloorType object.
azimuthRoof	Member.Roof, Face	Calculation	Calculate the angle by obtaining the exterior side face from each roof, projecting its normal vector onto the horizontal plane, and calculating the angle between the projection and the north axis.
tiltRoof	Member.Roof, Member.Roof.roofPitchAngle	Application	Extract the angle (roofPitchAngle attribute) directly from the RoofType object.
infraredAbsorptance	Material.infraredAbsorptance	Application	Extract the infrared absorptance directly from the Material object associated with each building component.
solarAbsorptance	Material.solarAbsorptance	Application	Extract the solar absorptance directly from the Material object associated with each building component.
roughness	LayerStructure.surfaceRoughness	Application	Extract the roughness directly from the LayerStructure object associated with each building component.
layerCount	LayerStructure.layerCount	Application	Extract the number of layers directly from the LayerStructure object associated with each building component.
thickness	LayerStructure.layerWidth	Application	Extract the thickness of layers directly from the LayerStructure object associated with each building component.
thermalConductivity	Material.thermalConductivity	Application	Extract the thermal conductivity directly from the Material object associated with each building component.
specificHeat	Material.specificHeat	Application	Extract the specific heat directly from the Material object associated with each building component.
density	Material.density	Application	Extract the density directly from the Material object associated with each building component.
widthWindow	Window.width	Application	Extract the width directly from the Window object.
heightWindow	Window.height	Application	Extract the height directly from the Window object.
fractionFrame	Window.frameFraction	Calculation	Calculate the window area as the product of height and width, estimate the frame area from the frame thickness along the perimeter with corner corrections, and calculate the frame ratio as the frame area divided by the window area.
UvalueFrame	Window.Uvalue	Application	Extract the U-value directly from the Window object.
heightRoom	Level.elevation	Application	Extract the elevation directly from the Level object.
nPort	Room.numberPorts	Application	Extract the number of ports directly from the Room object.
Medium	Room.medium	Application	Extract the medium directly from the Room object.

4.2.2. Definition of PBIM-FMU Data Structure

To facilitate intuitive linkage between the BIM model and the Modelica library, the PBIM-FMU data structure was designed to follow the topological structure of architectural spaces as defined in the BIM model. Figure 4 illustrates the PBIM-FMU data structure, consisting of five object types: Wall (FMU_Wall), Roof (FMU_Roof), Floor (FMU_Floor), Window (FMU_Window), and Room (FMU_Room), each containing associated physical properties.

These objects represent object-based components derived from the physical properties defined in Table 2 and are organized from the perspective of spatial composition. Since the defined physical properties are semantically meaningful and directly usable for energy analysis, the PBIM-FMU data structure allows for a clear understanding of how changes to physical properties of building components during the design stage affect performance outcomes.

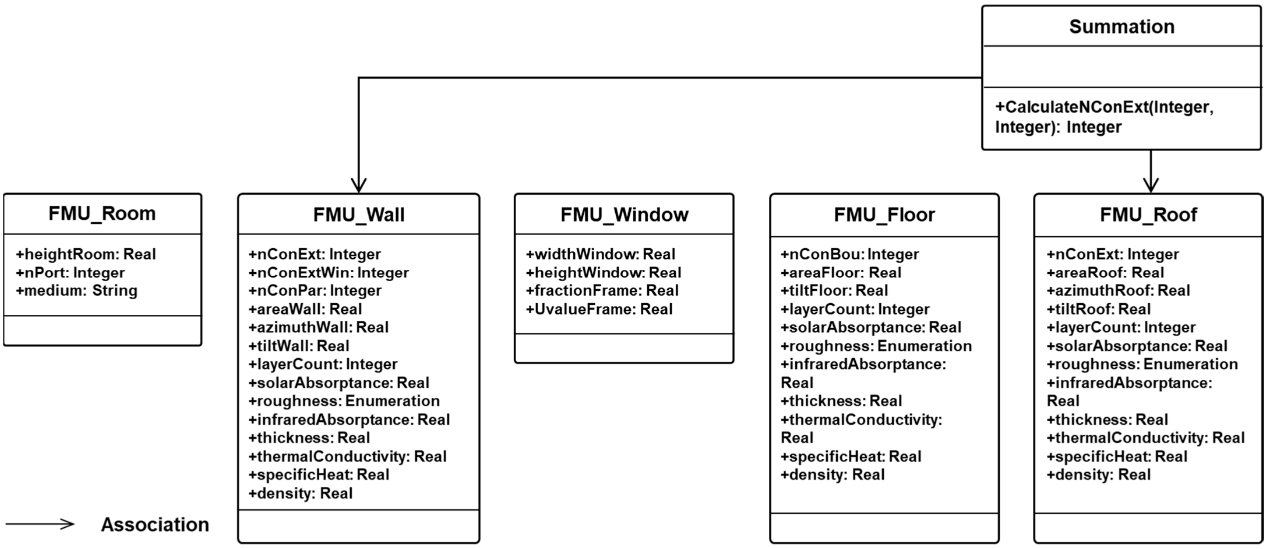


Figure 4. Schematic class diagram of the PBIM-FMU data model.

In the Buildings library, building envelope models are categorized into five boundary condition types: CE (Construction Exterior), CEW (CE with Window), CB (Construction Boundary), SB (Surface Boundary), and CP (Construction Partition). Accordingly, the five object types of the PBIM-FMU were designed to be mapped to their respective boundary condition types. For example, the FMU_Wall object can be mapped to CE if it represents an internal partition within the building. The FMU_Roof object can be mapped to CE since they are exposed to the outdoor environment, while the FMU_Floor object can be mapped to CB if they are located between two adjacent levels or if they are not in contact with the ground. The Summation object shown in Figure 4 aggregates all wall and roof objects exposed to the outdoor environment and assigns the results to the variable nConExt, which represents the total number of CE-type boundary conditions.

4.2.3. PBIM-FMU Implementation

The five object types—wall, roof, floor, window, and room—shown in Figure 4 were each implemented as a separate Model Exchange FMU (*.fmu). For PBIM-FMU development, the FMI Application Programming Interface (API) provided by Modelica Association [40], based on the C programming language and FMI version 2.0, was used in conjunction with the C# programming language and Revit API (version 2024). Since FMI version 2.0 does not natively support transferring variables as array structures, the building design information was decomposed into scalar variables and passed individually to the PBIM-FMU.

The PBIM-FMU implementation process consisted of three stages: (1) Generation of the model description file, (2) Generation of the model execution file, and (3) FMU compilation via an integrated development environment.

The model description file is an XML-formatted file that contains the FMU's metadata and configuration. A Revit add-in was developed using the C# programming language and the Revit API to extract building design information from the BIM model and generate model description files for each of the five object types. These files were saved to a pre-designated directory and included the FMU name, FMI version, FMU type, and the variable semantics and initial values defined in Table 2. Listing 1 illustrates the variable semantics and initial values for the parameters describing the number of boundary conditions of the Construction Exterior with Window type (nConExtWin) and the area of the Construction Exterior with Window (area_WallWindow_1).

Listing 1. Example of an XML-formatted PBIM-FMU model description file automatically generated by the Revit add-in.

```
<ModelVariables>
 <ScalarVariable name="nConExtWin" valueReference="0" causality="parameter"
 variability="fixed" initial="exact" description="Number of exterior construction
 with window">
   <Integer start="1"/>
 </ScalarVariable>
 <ScalarVariable name="area_WallWindow_1" valueReference="1" causality="parameter"
 variability="fixed" initial="exact" description="Area of 1st wall with window">
   <Real start="21.6"/>
 </ScalarVariable>
</ModelVariables>
```

The model execution file is a C source code (*.c) file that specifies the behavior of the FMU during simulation. The same Revit add-in also generated model execution files for each object type. These files included function definitions for reading and setting variable values during simulation and detailed the variable semantics and initial values as specified in Table 2. Listing 2 illustrates the internal operation of the PBIM-FMU implemented in the C programming language. The function setStartValues at line 1 initializes the model parameter nConExtWin. The function getFloat64 starting at line 2 is responsible for returning real-valued variables according to their value references. In a similar manner, the set functions can be implemented to assign new values to parameters. This mechanism ensures that the PBIM-FMU correctly handles initialization and variable access during simulation.

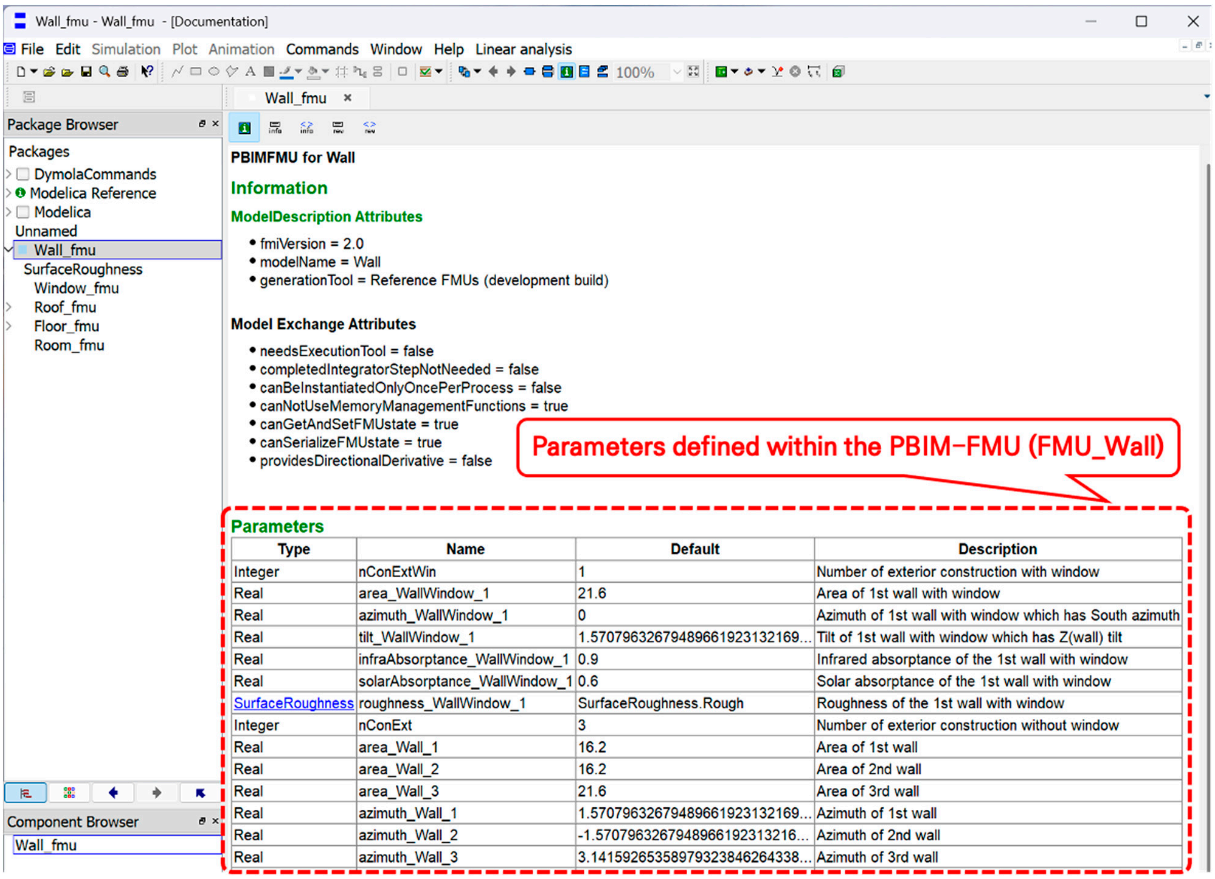
Listing 2. Example of an PBIM-FMU internal C source code automatically generated by the Revit add-in.

```
1    void setStartValues(ModelInstance* comp) { M(nConExtWin)=1; }
2    Status getFloat64(ModelInstance* comp, const ValueReference vr, double values[], size_t
     nValues, size_t* index) {
3      UNUSED(nValues);
4        switch (vr) {
5          case vr_areawallwindow1:
6            values[(*index)++] = M(area_WallWindow_1);
7            return OK;
8          default:
9            return Error; } }
```

Both the model description and execution files, along with a pre-defined build configuration file (specifying file paths, compiler settings, runtime environment, and library dependencies), were compiled, built, then packaged into a FMU file using Visual Studio 2019. The result was a set of five PBIM-FMU files corresponding to the wall, roof, floor, window, and room components.

4.3. Generation of FMI-Based OOPM Model (Activity A3)

The five PBIM-FMU files were imported into the Modelica-based simulation tool Dymola 2017 using its FMU import functionality. Each PBIM-FMU file was recognized as an independent component within the simulation environment, and—as illustrated in Figure 5—the FMU_Wall object could be loaded into the tool together with its model description. Each imported file was executable in isolation, and the parameter values defined within the PBIM-FMU could be exported as simulation results.



PBIMFMU for Wall

Information

ModelDescription Attributes

- fmiVersion = 2.0
- modelName = Wall
- generationTool = Reference FMUs (development build)

Model Exchange Attributes

- needsExecutionTool = false
- completedIntegratorStepNotNeeded = false
- canBeInstantiatedOnlyOncePerProcess = false
- canNotUseMemoryManagementFunctions = true
- canGetAndSetFMUstate = true
- canSerializeFMUstate = true
- providesDirectionalDerivative = false

Parameters

Type	Name	Default	Description
Integer	nConExtWin	1	Number of exterior construction with window
Real	area_WallWindow_1	21.6	Area of 1st wall with window
Real	azimuth_WallWindow_1	0	Azimuth of 1st wall with window which has South azimuth
Real	tilt_WallWindow_1	1.57079632679489661923132169...	Tilt of 1st wall with window which has Z(wall) tilt
Real	infraAbsorptance_WallWindow_1	0.9	Infrared absorptance of the 1st wall with window
Real	solarAbsorptance_WallWindow_1	0.6	Solar absorptance of the 1st wall with window
SurfaceRoughness	roughness_WallWindow_1	SurfaceRoughness.Rough	Roughness of the 1st wall with window
Integer	nConExt	3	Number of exterior construction without window
Real	area_Wall_1	16.2	Area of 1st wall
Real	area_Wall_2	16.2	Area of 2nd wall
Real	area_Wall_3	21.6	Area of 3rd wall
Real	azimuth_Wall_1	1.57079632679489661923132169...	Azimuth of 1st wall
Real	azimuth_Wall_2	-1.5707963267948966192313216...	Azimuth of 2nd wall
Real	azimuth_Wall_3	3.14159265358979323846264338...	Azimuth of 3rd wall

Figure 5. Example of wall object parameters of the PBIM-FMU imported into Dymola 2017.

To enable the transfer of parameter values from the PBIM-FMU to a Modelica model during simulation, the parameters of the Modelica model needed to be explicitly assigned to the corresponding parameters of the PBIM-FMU. Accordingly, Modelica code that incorporated these parameter assignments needed to be prepared in advance. In this study, the assignment code was generated manually. Ultimately, the PBIM-FMU were combined with the target Modelica models to establish an integrated building performance analysis model, i.e., an FIM-based OOPM model.

5. Validation

This section verifies the applicability of the proposed framework as a data exchange model between BIM and Modelica-based simulation environments. To this end, a representative case model was selected, and simulations were conducted to compare results with those obtained using a manually constructed model. If the simulation results between the two models are similar or identical, it can be inferred that the PBIM-FMU accurately transfers building design information from BIM to the Modelica library, thereby validating the effectiveness of the proposed framework.

5.1. Case Model Selection and Experimental Setup

For verification, Case600 from ASHRAE Standard 140-2017 [41] was adopted as the benchmark model. Case600 represents a single-zone thermal model with dimensions of 8 m in width, 6 m in depth, and 2.7 m in height, including two south-facing windows with a total glazing area of 6 m^2. The building is configured with low thermal mass, an infiltration rate of 0.5 ACH, and a constant internal sensible heat gain of 200 W. The heating and cooling setpoint temperatures are 20 °C and 27 °C, respectively. The location for the simulation is set to Denver, Colorado, USA.

The simulation was conducted for a one-year period, with a time step of 3600 s and a simulation tolerance of 1×10^{-6}. Weather data from DRYCOLD.TMY (Denver, CO, USA) were used, and Dymola 2017 served as the simulation tool. In particular, a pre-validated model from the Buildings library (version 8.1.3) was employed as a reference [20,42].

5.2. Application of the Framework

To apply the proposed framework, a BIM model of Case600 was created using Revit 2024. Additional physical properties not natively supported by Revit's data structure were manually incorporated to extend the model into a PBIM. Based on the PBIM, physical design parameters required for simulation were extracted to generate the model description and model implementation files. These files were then compiled using Visual Studio 2019 to create the PBIM-FMU.

The generated PBIM-FMU was imported into the Dymola 2017 simulation environment and coupled with the Case600 model from the Buildings library to construct the integrated building performance analysis model. Figure 6 presents an overview of the integrated model. Both the manually authored Case600 model and the PBIM-FMU-based integrated model were simulated for thermal analysis, and results such as annual heating/cooling loads and hourly peak loads were analyzed.

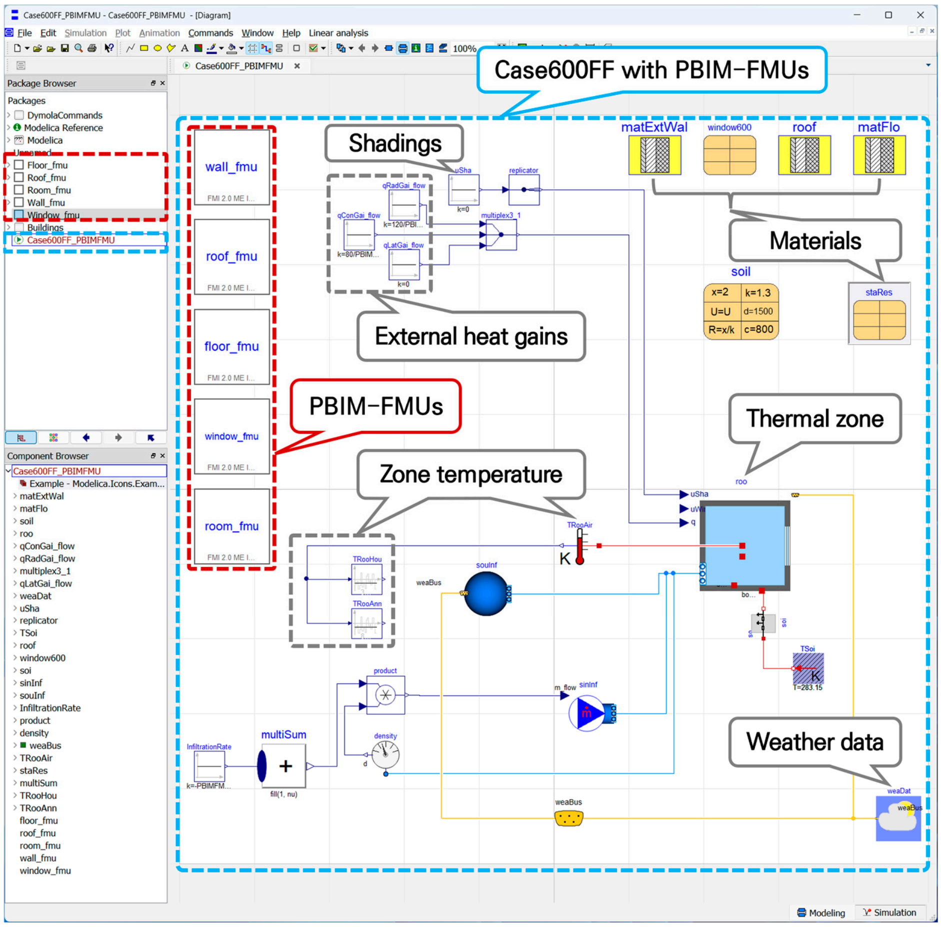


Figure 6. Schematic representation of the integrated building performance analysis model constructed by coupling PBIM-FMU instances with the Case600 model from the LBNL Buildings library.

5.3. Comparison of Simulation Results

The simulation results of the original Case600 model from the Buildings library and the integrated model incorporating PBIM-FMU showed strong agreement. Figure 7 compares the annual heating and cooling loads of these models with those of other simulation tools specified in ASHRAE Standard 140-2017. Both the original and integrated models yielded identical results: an annual heating load of 5.297 MWh and an annual cooling load of 6.669 MWh.

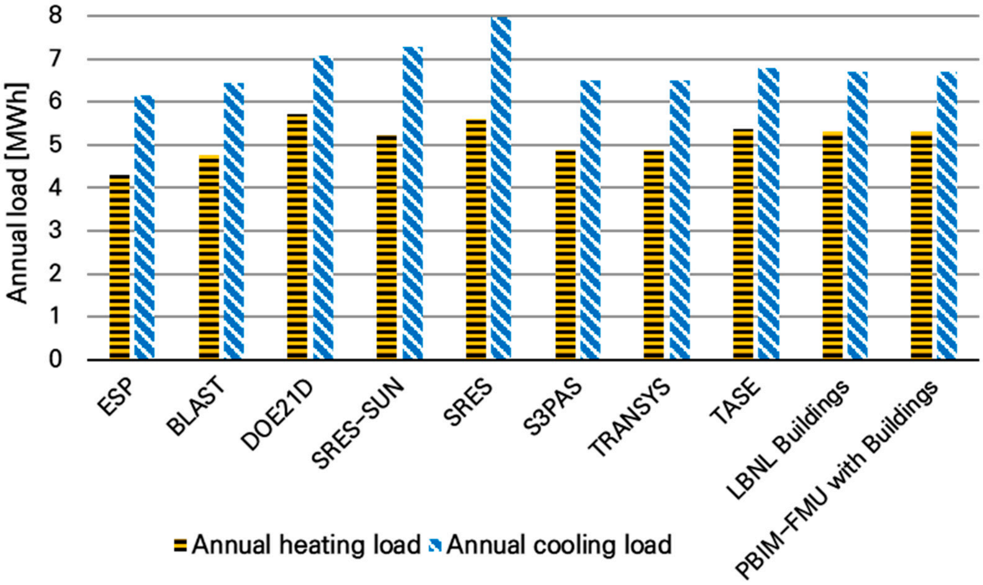


Figure 7. Annual heating and cooling loads of Case600 obtained from the proposed framework, the LBNL Buildings library, and other simulation tools referenced in ASHRAE Standard 140-2017.

Figure 8 presents a comparison of hourly heating and cooling loads on January 4, the date of maximum heating demand. The two curves are nearly indistinguishable, indicating a high level of consistency between the models in terms of hourly thermal load prediction. Table 4 compares the peak heating and cooling loads and their occurrence times with the results of various simulation tools from ASHRAE Standard 140-2017. Both the Buildings library model and the integrated model reported identical peak loads (heating: 4.172 kWh at 6:00 AM on 4 January; cooling: 6.669 kWh at 1:00 PM on 17 October).

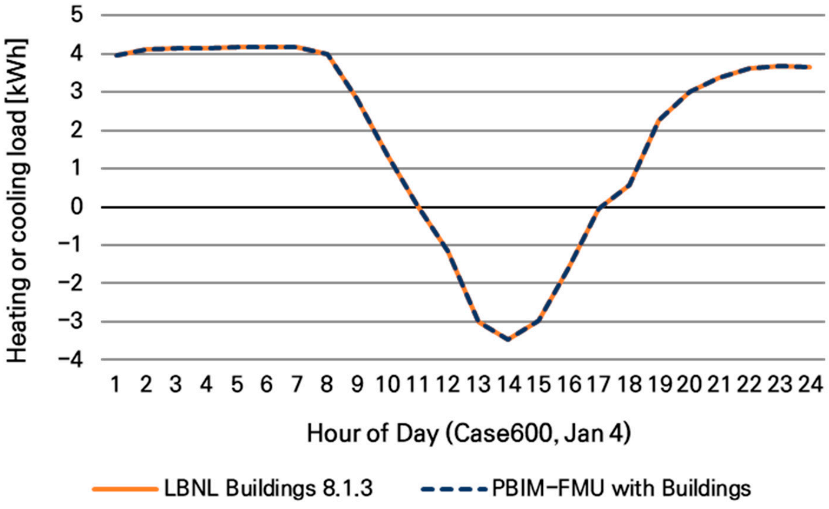


Figure 8. Comparison of hourly heating and cooling loads on 4 January between the proposed framework and the Case600 model from the LBNL Buildings library.

Table 4. Annual hourly integrated peak heating and cooling loads of Case600 from the proposed framework, LBNL Buildings library, and other simulators in ASHRAE Standard 140-2017.

Simulation Engine	Peak Heating Load (Day-Month, Time) [kWh]	Peak Cooling Load (Day-Month, Time) [kWh]
ESP	3.437 (4-January, 5)	6.194 (17-October, 13)
BLAST	3.940 (4-January, 5)	5.965 (16-October, 14)
DOE21D	4.045 (4-January, 5)	6.656 (16-October, 13)
SRES-SUN	4.258 (4-January, 2)	6.827 (16-October, 14)
S3PAS	4.037 (4-January, 2)	6.286 (25-November, 14)
TRANSYS	3.931 (4-January, 6)	6.486 (16-October, 14)
TASE	4.354 (4-January, 2)	6.812 (17-October, 14)
LBNL Buildings 8.1.3	4.172 (4-January, 6)	6.669 (17-October, 13)
PBIM-FMU with Buildings 8.1.3	4.172 (4-January, 6)	6.669 (17-October, 13)

These results confirm that the PBIM-FMU accurately transfers building design information to the Modelica-based simulation model. Therefore, the proposed framework can be effectively used as a data exchange model between BIM environments and Modelica-based multi-domain simulation tools.

6. Conclusions

This study proposed an information exchange framework to facilitate efficient multidisciplinary building performance analysis during the building design stage. The framework extracts building design information required for performance simulation from a BIM model and generates a Functional Mock-up Unit (PBIM-FMU) compliant with the FMI standard, enabling seamless interoperability with various Modelica models and Modelica-based simulation environments. Simulation results confirmed that the PBIM-FMU accurately transfers the building design information to the corresponding Modelica libraries.

Previous research on linking BIM models with simulation models has primarily focused on directly generating specific simulation models from BIM data. For example, the IEA/EBC Annex 60, BIM2Modelica, and Ifc2Modelica studies differ in terms of data models, target Modelica libraries, and translation processes, but all share the common approach of translating BIM data into simulation models by producing Modelica code. However, such outputs are inherently tied to the Modelica language and to the specific libraries (e.g., Buildings, IDEAS, BuildingSystems) on which they are based. This dependency arises because BIM objects are directly mapped to classes and components of a particular Modelica library, and the resulting models are constrained by the syntax of the Modelica language and the parameter structures of the selected library.

In addition, the FMI4BIM project demonstrated two workflows: (1) translating BIM data into building thermal model FMUs using specialized simulation engines such as NANDRAD, and (2) converting BIM data into Modelica libraries to generate HVAC system model FMUs. However, these outputs are FMUs that can only be generated by specific solvers or libraries, and thus they inherently contain engine- or library-level dependencies. Consequently, existing studies have focused on model translation for specific libraries or simulation tools rather than on generalized data exchange, and the data exchange into multi-domain simulation environments has remained limited.

In contrast, the PBIM-FMU is not a completed simulation model but an intermediate model that organizes and represents building design information extracted from the BIM model as simulation-relevant parameters. Moreover, it is not merely a data exchange format but an FMU object that can be independently loaded into simulation environments in accordance with the FMI standard. Unlike prior intermediate data models, which could only be used once translated into specific simulation models, the PBIM-FMU possesses modularity that enables it to exist independently within the simulation environment. Therefore, the PBIM-FMU is not bound to any specific programming language or library ecosystem and can provide building design information for use across multiple simulation domains.

This library- and tool-independent characteristic of the PBIM-FMU establishes a foundation for efficiently delivering information not only within traditional building performance domains such as building envelope, HVAC, and control systems but also to other heterogeneous domains such as structural analysis. In other words, the PBIM-FMU has the potential to serve as a component for constructing multi-domain simulation environments.

In the present study, verification was performed using a single-zone model to demonstrate the correctness of data exchange. However, further verification is needed to progressively extend applicability to a wider range of building types as well as to complex and large-scale buildings. This requires validation against standardized benchmark models, as well as application and verification using real-world BIM models to demonstrate practical validity and scalability. Furthermore, comparative testing between conventional data exchange approaches and the proposed method should be conducted. Such testing must quantitatively evaluate performance differences in terms of accuracy, speed, and usability, with results analyzed using appropriate statistical methods.

The proposed framework has not yet achieved full automation. A3 activity (the generation of the FMI-based OOPM model) in Figure 1 currently depends on manual work, highlighting the need for further research on automatic mapping between PBIM-FMUs and Modelica-based simulation models. This may involve employing code generators such as CoTeTo or developing interfaces that allow PBIM-FMU data to be visualized and inspected within the simulation environment, enabling users to more easily map PBIM-FMU parameters to OOPM model parameters.

Moreover, the framework defines a data structure for an extended Physical BIM model that includes additional physical properties required for simulation, and this structure was implemented using BIM authoring tool Autodesk Revit and the Revit API. On the basis of this defined data structure, implementation feasibility could be tested with other BIM authoring tools. Since the OOPM data structure has been designed with reference to the Buildings library, its reusability and interoperability may be limited in broader multi-domain simulation contexts. To better support linkage between BIM and OOPM, a data structure capable of representing building design information required for simulation in a library- and tool-independent manner is needed. Building upon this, extending the framework to domains beyond building environments—such as structural engineering—requires the development of OOPM models for structural analysis and research on their efficient integration with BIM. This will be essential to demonstrate that the proposed framework can serve as a foundation for multi-domain simulation platforms.

Author Contributions: Conceptualization, writing, visualization, funding acquisition, and methodology, W.J.; original draft writing, application development, and formal analysis, B.K. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by a funding for the academic research program of Chungbuk National University in 2025.

Data Availability Statement: The data presented in this study are available upon request from the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. International Energy Agency (IEA). World Energy Outlook 2023. Available online: https://www.iea.org/reports/world-energy-outlook-2023 (accessed on 5 August 2025).
2. U.S. Energy Information Administration (EIA). International Energy Outlook 2023. Available online: https://www.eia.gov/outlooks/ieo/index.php (accessed on 5 August 2025).
3. Fonseca Arenas, N.; Shafique, M. Recent Progress on BIM-Based Sustainable Buildings: State of the Art Review. *Dev. Built. Environ.* **2023**, *15*, 100176. [CrossRef]
4. Ebrahim, A.; Wayal, A.S. Green BIM for Sustainable Design of Buildings. In *Proceedings of the ICRRM 2019—System Reliability, Quality Control, Safety, Maintenance and Management*; Gunjan, V.K., Singh, S.N., Duc-Tan, T., Rincon Aponte, G.J., Kumar, A., Eds.; Springer: Singapore, 2020; pp. 185–189. [CrossRef]
5. Pan, X.; Mateen Khan, A.; Eldin, S.M.; Aslam, F.; Kashif Ur Rehman, S.; Jameel, M. BIM Adoption in Sustainability, Energy Modelling and Implementing Using ISO 19650: A Review. *Ain. Shams. Eng. J.* **2024**, *15*, 102252. [CrossRef]
6. Maglad, A.M.; Houda, M.; Alrowais, R.; Khan, A.M.; Jameel, M.; Rehman, S.K.U.; Khan, H.; Javed, M.F.; Rehman, M.F. Bim-Based Energy Analysis and Optimization Using Insight 360 (Case Study). *Case Stud. Constr. Mater.* **2023**, *18*, e01755. [CrossRef]
7. Visby Fjerbæk, E.; Seidenschnur, M.; Kücükavci, A.; Michael Smith, K.; Anker Hviid, C. Coupling BIM and Detailed Modelica Simulations of HVAC Systems in a Common Data Environment. *J. Build. Perform. Simul.* **2023**, *17*, 306–321. [CrossRef]
8. Richter, V.E.; Syndicus, M.; Frisch, J.; van Treeck, C. Extending the IFC-Based Bim2sim Framework to Improve the Accessibility of Thermal Comfort Analysis Considering Future Climate Scenarios. *Appl. Sci.* **2023**, *13*, 12478. [CrossRef]
9. Qiu, K.; Yang, J.; Gao, Z.; Xu, F. A Review of Modelica Language in Building and Energy: Development, Applications, and Future Prospect. *Energy Build.* **2024**, *308*, 113998. [CrossRef]
10. Afzal, M.; Liu, Y.; Cheng, J.C.; Gan, V.J. Reinforced Concrete Structural Design Optimization: A Critical Review. *J. Clean. Prod.* **2020**, *260*, 120623. [CrossRef]
11. Wetter, M.; Benne, K.; Tummescheit, H.; Winther, C. Spawn: Coupling Modelica Buildings Library and EnergyPlus to Enable New Energy System and Control Applications. *J. Build. Perform. Simul.* **2024**, *17*, 274–292. [CrossRef]
12. Wetter, M.; van Treeck, C. *IEA EBC Annex 60: New Generation Computational Tools for Building & Community Energy Systems*; International Energy Agency: Paris, France, 2017; ISBN 978-0-692-89748-5. Available online: https://www.iea-annex60.org/final-report.html (accessed on 5 August 2025).
13. Eckstädt, E.; Huang, C.; Liersch, C.; Paepcke, A.; Hoch, R.; Nicolai, A.; Grunewald, J.; Majetta, K.; Borchert, D.; Schumann, F.; et al. *FMI4BIM—Standardisierte Schnittstelle für Analysemodelle von Anlagen- und Gebäudekomponenten für BIM-Basierte Planung und Betrieb*; Fraunhofer IIS EAS: Dresden, Germany, 2023; ISBN 978-3-95908-323-2. Available online: https://publica.fraunhofer.de/entities/publication/b31b4fc7-f284-4d03-bd19-06aa69e16ac2 (accessed on 5 August 2025).
14. Wetter, M. Modelica-Based Modelling and Simulation to Support Research and Development in Building Energy and Control Systems. *J. Build. Perform. Simul.* **2009**, *2*, 143–161. [CrossRef]
15. Blochwitz, T.; Otter, M.; Åkesson, J.; Arnold, M.; Clauss, C.; Elmqvist, H.; Friedrich, M.; Junghanns, A.; Mauss, J.; Neumerkel, D. Functional Mockup Interface 2.0: The Standard for Tool Independent Exchange of Simulation Models. In Proceedings of the 9th International Modelica Conference, München, Germany, 3–5 September 2012; pp. 173–184. [CrossRef]
16. Fishwick, P.A. Integrating Continuous and Discrete Models with Object Oriented Physical Modeling. In Proceedings of the 1997 Western Simulation Multiconference, Phoenix, AZ, USA, 12–15 January 1997.
17. Elmqvist, H.; Mattsson, S.E.; Otter, M. Modelica—A language for physical system modeling, visualization and interaction. In Proceedings of the 1999 IEEE International Symposium on Computer Aided Control System Design (CACSD), Big Island, HI, USA, 22–27 August 1999; pp. 630–639. [CrossRef]
18. Mattsson, S.E.; Elmqvist, H.; Otter, M. Physical System Modeling with Modelica. *Control Eng. Pract.* **1998**, *6*, 501–510. [CrossRef]
19. Brück, D.; Elmqvist, H.; Mattsson, S.E.; Olsson, H. Dymola for Multi-Engineering Modeling and Simulation. In Proceedings of the Modelica 2002 Conference, Munich, Germany, 18–19 March 2002; pp. 55–64.
20. Wetter, M.; Zuo, W.; Nouidui, T.S.; Pang, X. Modelica Buildings Library. *J. Build. Perform. Simul.* **2014**, *7*, 253–270. [CrossRef]
21. Jorissen, F.; Reynders, G.; Baetens, R.; Picard, D.; Saelens, D.; Helsen, L. Implementation and Verification of the IDEAS Building Energy Simulation Library. *J. Build. Perform. Simul.* **2018**, *11*, 669–688. [CrossRef]
22. Nicolai, A.; Söhnchen, A. FMI Co-Simulation between 2D/3D Component Models and HVAC/Control Models. In Proceedings of the Building Simulation 2021, Brugge, Belgium, 1–3 September 2021; pp. 1983–1990. [CrossRef]
23. Nicolai, A.; Paepcke, A. Co-Simulation between Detailed Building Energy Performance Simulation and Modelica HVAC Component Models. In Proceedings of the 12th International Modelica Conference, Prague, Czech Republic, 15–17 May 2017; pp. 63–72. [CrossRef]
24. Jeong, W.; Kim, J.B.; Clayton, M.J.; Haberl, J.S.; Yan, W. A Framework to Integrate Object-Oriented Physical Modelling with Building Information Modelling for Building Thermal Simulation. *J. Build. Perform. Simul.* **2016**, *9*, 50–69. [CrossRef]

25. Kim, J.B.; Jeong, W.; Clayton, M.J.; Haberl, J.S.; Yan, W. Developing a Physical BIM Library for Building Thermal Energy Simulation. *Autom. Constr.* **2015**, *50*, 16–28. [CrossRef]
26. Remmen, P.; Cao, J.; Ebertshäuser, S.; Frisch, J.; Lauster, M.; Maile, T.; O Donnell, J.; Pinheiro, S.; Rädler, J.; Thorade, M.; et al. An open framework for integrated BIM-based building performance simulation using Modelica. In Proceedings of the Building Simulation 2015: 14th Conference of IBPSA, Hyderabad, India, 7–9 December 2015. [CrossRef]
27. Thorade, M.; Rädler, J.; Remmen, P.; Maile, T.; Wimmer, R.; Cao, J.; Lauster, M.; Nytsch-Geusen, C.; Müller, D.; Van Treeck, C. An open toolchain for generating Modelica code from Building Information Models. In Proceedings of the 11th International Modelica Conference, Versailles, France, 21–23 September 2015; pp. 383–391. [CrossRef]
28. Andriamamonjy, A.; Saelens, D.; Klein, R. An Automated IFC-Based Workflow for Building Energy Performance Simulation with Modelica. *Autom. Constr.* **2018**, *91*, 166–181. [CrossRef]
29. Nytsch-Geusen, C.; Rädler, J.; Thorade, M.; Tugores, C.R. BIM2Modelica—An open source toolchain for generating and simulating thermal multi-zone building models by using structured data from BIM models. In Proceedings of the 13th International Modelica Conference, Regensburg, Germany, 4–6 March 2019; pp. 33–38. [CrossRef]
30. Seidenschnur, M.; Kücükavci, A.; Fjerbæk, E.V.; Smith, K.M.; Pauwels, P.; Hviid, C.A. A Common Data Environment for HVAC Design and Engineering. *Autom. Constr.* **2022**, *142*, 104500. [CrossRef]
31. Jansen, D.; Hering, D.; Müller, D. BIM2SIM for Hydraulic-Focussed Energy Simulations—Automatic Generation of Pre Parametrized Simulation Models. In Proceedings of the 36th International Conference on Efficiency, Cost, Optimization, Simulation and Environmental Impact of Energy Systems (ECOS 2023), Las Palmas de Gran Canaria, Spain, 25–30 June 2023; pp. 3229–3240. [CrossRef]
32. Autodesk. Revit 2024: Release Notes. Available online: https://help.autodesk.com/view/RVT/2024/ENU/?guid=RevitReleaseNotes_2024release_html (accessed on 5 August 2025).
33. Gao, H.; Koch, C.; Wu, Y. Building Information Modelling Based Building Energy Modelling: A Review. *Appl. Energy* **2019**, *238*, 320–343. [CrossRef]
34. Lawrence Berkeley National Laboratory. Buildings Library 11.0.0—Release Notes. Available online: https://simulationresearch.lbl.gov/modelica/releases/v11.0.0/help/Buildings_UsersGuide_ReleaseNotes.html#Buildings.UsersGuide.ReleaseNotes.Version_1_3_build1 (accessed on 5 August 2025).
35. Nytsch-Geusen, C.; Inderfurth, A.; Kaul, W.; Mucha, K.; Rädler, J.; Thorade, M.; Tugores, C.R. Template Based Code Generation of Modelica Building Energy Simulation Models. In Proceedings of the 12th International Modelica Conference, Prague, Czech Republic, 15–17 May 2017; pp. 199–207. [CrossRef]
36. Hejlsberg, A.; Torgersen, M.; Wiltamuth, S.; Golde, P. *The C# Programming Language (Covering C# 4.0)*; Addison-Wesley Professional: Boston, MA, USA, 2010; ISBN 978-0-13-248172-4.
37. Yang, Z.; Du, H.; Chen, J.; Shen, J. Design of High-Precision Solver Based on FMI Specification. In Proceedings of the 2023 IEEE 18th Conference on Industrial Electronics and Applications (ICIEA), Ningbo, China, 18–22 August 2023; pp. 527–532.
38. National Institute of Standards and Technology (NIST). *FIPS Publication 183: Integration Definition for Function Modeling (IDEF0)*; NIST: Gaithersburg, MD, USA, 1993. Available online: https://nvlpubs.nist.gov/nistpubs/Legacy/FIPS/fipspub183.pdf (accessed on 5 August 2025).
39. Object Management Group (OMG). Unified Modeling Language (UML). Available online: http://www.uml.org/#Links-Tools (accessed on 5 August 2025).
40. Modelica Association. Reference FMUs: Functional Mock-Up Units for Development, Testing, and Debugging (Repository). Available online: https://github.com/modelica/reference-fmus (accessed on 5 August 2025).
41. American Society of Heating, Refrigerating, and Air-Conditioning Engineers, Inc. ANSI/ASHRAE 140-2017: Standard Method of Test for the Evaluation of Building Energy Analysis Computer Programs. Atlanta, GA, USA, 2017. Available online: https://codehub.building.govt.nz/resources/ansiashrae-140-2017 (accessed on 5 August 2025).
42. Nouidui, T.S.; Phalak, K.; Zuo, W.; Wetter, M. Validation and Application of the Room Model of the Modelica Buildings Library. In Proceedings of the 9th International Modelica Conference, Munich, Germany, 3–5 September 2012; pp. 727–736. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.