CLIMA 2022
Eye on 2030 | Towards digitalized, healthy, circular and energy efficient HVAC
REHVA 14th HVAC World Congress
22nd – 25th May, Rotterdam, The Netherlands

From BIM databases to Modelica - Automated simulations of heating systems

Esben Visby Fjerbæk[a], *Mikki* Seidenschnur[a,b], *Ali* Kücükavci[a,c], *Kevin Michael* Smith[a], *Christian Anker* Hviid[a]

[a] Department of Civil Engineering, Technical University of Denmark, Kgs. Lyngby, Denmark, evifj@byg.dtu.dk, kevs@byg.dtu.dk, cah@byg.dtu.dk.

[b] Rambøll, Copenhagen, Denmark, msei@ramboll.dk

[c] Cowi, Kgs. Lyngby, Denmark, alkc@cowi.dk

Abstract. Detailed simulations of HVAC systems play a crucial role in creating 1:1 digital twins of buildings and their systems. In particular, detailed models of hydronic systems are essential for fault detection of building services and control optimization. However, modeling HVAC systems is labour intensive due to the components and connections that one must create based on drawings or models. Creating the HVAC simulation models from BIM data eases the modeling burden, simplifying the creation of digital twins. Straight-forward HVAC simulations can aid the design process. Instead of prescriptive design based on the worst-case conditions, simulations enable performance-based design with partial-loads and dynamic behaviour. This paper presents a preliminary tool using BIM data to create and simulate models of heating systems. The tool uses a central BIM data platform with a dedicated data format – defining components and their relations in a database. Python scripts apply model templates to create heating system models in the Modelica language. The tool simulates the models in Dymola, while Python scripts read and parse the results to the database for visualization and analysis. The tool efficiently simulated a small heating system and obtained results for the return temperatures of several loops.

Keywords. BIM, Modelica, HVAC, simulations
DOI: https://doi.org/10.34641/clima.2022.365

1. Introduction

There is a large discrepancy between the estimated energy consumptions of buildings and the one measured, often leading to underestimated energy consumption [1, 2]. The issue, known as the performance gap, has many causes. One significant cause is the precision of building energy performance simulations (BEPS) used to estimate energy consumption. BEPS tools rarely consider HVAC systems in detail [3] or simply assume ideal performance of components and controls [4]. This means that commonly occurring phenomena such as oscillation or system imbalance, that create disturbances, are not identified by the BEPS models.

Modeling HVAC systems in detail ensures that all non-ideal performance is considered in the BEPS simulations, which increases simulation precision. In the design phase, this can help to evolve the design process from a steady-state practice, where the design of HVAC systems is based on a worst-case full-capacity situation, to a dynamic design paradigm, where requirements for part-load conditions and dynamic behavior define the design. Additionally, it will be possible to check that the detail of the design is sufficient for actual operation under all conditions. Today, this is ensured through guidelines provided by component manufacturers and empirical knowledge of practitioners.

The combination of traditional BEPS models and detailed models of all HVAC systems can act as a digital twin if connected to live weather data. During operation, the digital twin estimates the expected/optimal operation of the building systems alongside the actual building. The expected operation can be used for fault detection and to test different control strategies continuously throughout the building's lifetime.

Modeling HVAC systems can, however, be a laborious task since the systems include many different elements that must be created manually. Often, the simulation models derive from diagrams of the systems, that the simulation engineer manually interprets and translates to the simulation model format. This error-prone process results in two separate models where changes to one does not affect the other.

Generating the HVAC simulation models from BIM

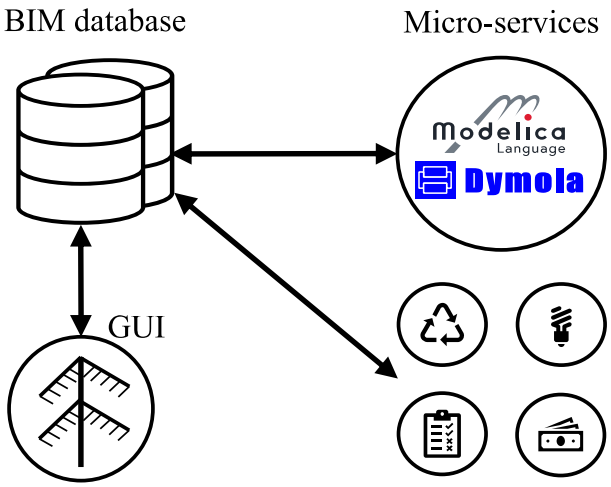

Fig. 1 - Several micro-services in addition to the Modelica service interface the BIM database platform.

data eases the burden of modeling. Integrating BIM and BEPS ensures that the BIM and simulation models share similar information. Both the BIM and BEPS industry work towards linking BIM and BEPS models. Tools such as IESVE [5] and IDA ICE [6] have functionalities to import geometry and construction parameters, whereas the BIM tool Revit contains some BEPS functionalities. Several tools for using BIM as a basis for models in the open-source Modelica language [7] exist [8–10], but all of these, including the traditional BEPS tools, have a primary focus on the envelope and thermal zone model. *IFC2Modelica* [10] includes an example for ventilation systems.

Common for all BIM to BEPS methodologies is the fact that they depend on file-based BIM information. Several critics argue that the use of file-based BIM models limits interoperability [11, 12]. A solution is to transfer from file-based to web-based collaboration, where information is exchanged through open data formats and stored in centralized databases [11, 12]. This corresponds to the BIM level 3 in the Bew-Richards BIM maturity model described in [13]. In the BIM maturity model, the information exchange on levels 0, 1, and 2 has different degrees of standardized data structures. In level 3, the information exchange is handled through standardized, open data formats for integration with various tools.

2. Cloud BIM platform

The toolchain, presented in the following sections, is implemented in a cloud platform that stores BIM models in a database to allow cross-platform access to the models. The platform is built with a micro-service structure, which means that several *micro-services* for design and evaluation of HVAC systems can utilize the data. Amongst these is the Modelica micro-service, which creates models in Modelica language and simulates them with Dymola. As seen in Fig. 1, the data flows back and forth between the micro-service and the database so that results are read and analyzed in the platform for analysis and visualization in the graphical user interface (GUI).

In the database, components and their relations are defined with the Flow Systems Ontology (FSO) [14, 15]. This ontology uses class hierarchies to define the type of component its relation to other components. E.g., a pipe supplying water to a radiator would have the class *Segment* and have the property *ConnectedWith* equal to the radiator's unique tag. Selected classes relevant to this project are listed in table Tab. 1, whereas the full list of classes and connections can be found online in [15].

Tab. 1 - Selected component classes

FSO class	Examples
Radiator	Radiators for heating
Segment	Pipe or duct segments
FlowController	Valves and dampers
FlowMovingDevice	Pumps and fans
HeatExchanger	Heating coils and heat recovery units

3. Toolchain

The toolchain automatically generates and simulates Modelica models of heating systems from BIM data. In fig. Fig. 2 the main processes of the toolchain are shown along with the flow of data. On the left side, the tool is connected through an API (see Fig. 3), that establishes an integration between the micro-service and the database. As seen in Fig. 3 the JSON format is used to parse data between the database and tool. As seen in Fig. 2 this is maintained throughout the tool, except for the last two steps, where the simulation environment needs a Modelica file and writes the results to a .mat file.

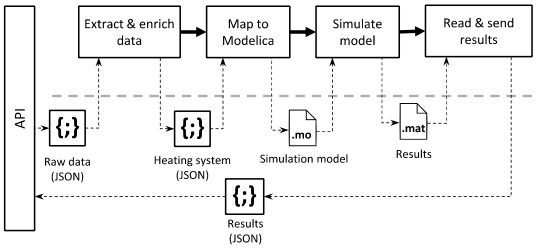

Fig. 2 - System architecture.

All functions are written with Python, since it is a straight-forward tool which is well suited for translation between data formats and since it is used in the BIM platform. Python does not carry out the simulations itself, but simply interfaces the simulation environment Dymola.

When the toolchain is activated, in the BIM platform, the platform sends a Post request, including BIM data for the desired system(s) to the service's API as seen in Fig. 3 where all data exchange between the database, the API and the toolchain is seen.

In the following sections, each step in the tool is described.

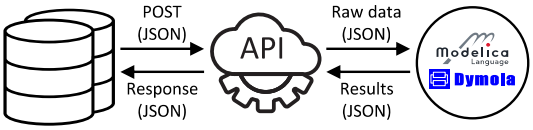

Fig. 3 - Interaction between the database and the toolchain through an API.

3.1. System extraction and data enrichment

When activating the tool, BIM data for the heating system is sent from the database, through the API to the tool. As a precaution and for future scenarios with several systems the tool extracts all components in the heating system from the data. To support the following mapping process, minor changes are made to the data by adding certain parameters based on the component classes. E.g., the length of pipe segments is calculated from the component's start and end coordinates. The enriched/manipulated data is then sent to the mapping process for model generation.

3.2. Mapping

In the mapping step, the Modelica models are generated. This is where the original data format and classes are translated to Modelica language and classes. This step is be divided into two separate processes; in the first, the program loops through all components and maps them to a corresponding Modelica class and instantiates it in the model code. In the second, all connections between the components are translated to Modelica connectors. To handle the lack of information on control, this is also where default control connections are established.

In the mapping process, seven FSO classes are mapped to 10 different Modelica classes. Some FSO classes have been mapped to multiple Modelica classes, depending on the value of certain attributes. The full mapping and the translated attributes are seen in Tab. 2. The parameters are all required in the BIM data; if not, the program will fail.

All Modelica classes, except the bend model, originate from the Buildings library [16] which includes models for most components in HVAC systems in addition to detailed models of thermal zones. To simplify the mapping process, a purpose-built library with models that combine component models from *Buildings* was created. The combined models simplify the mapping process, since several Modelica models would otherwise have to be instantiated for each database component. Examples are the radiator model, which combines a radiator model and a thermostat, acting as a proportional controller and the MotorValve class, which combines a motorized valve with a PI controller and a setpoint.

Tab. 2 - Mapping between classes in the database and their corresponding Modelica classes.

Component	Modelica
Segment	**model:** Pipe[a] **parameters:** nominal flow, insulation thickness, insulation lambda, diameter, length
FlowMoving-Device[d]	**model:** PumpConstantSpeed[b] **parameters:** speed, performance curve
FlowMoving-Device[d]	**model:** PumpConstantPressure[b] **parameters:** head, performance curve
Radiator	**model:** Radiator[b] **parameters:** nominal heat flux, nominal supply temp, nominal return temp, nominal room temp, nominal pressure loss
HeatExchanger	**model:** DryCoilCounterFlow[a] **parameters:** nominal air flow, nominal water flow, dp nominal air, dp nominal water, UA nominal
Bend	**model:** CurvedBend[c] **parameters:** angle, diameter, bend radius
Tee	**model:** Junction[a]
FlowControl-ler[d]	**model:** Valves.TwoWayLinear[a] **parameters:** nominal flow, Kv-value
FlowControl-ler[d]	**model:** CheckValve[a] **parameters:** nominal flow, Kvs-value
FlowControl-ler[d]	**model:** MotorValve[b] **parameters:** nominal flow, Kvs-value

[a] In Buildings.Fluid library

[b] In purpose-built library
[c] In Mocelica.Fluid library
[d] Mapping depends on component attributes

In the connection process, the connections between components are translated from the database format to Modelica language. In the database, the connection between components are described in *connectors*, in the *ConnectedWith* attribute, as seen in Fig. 4 that shows an example of two connected pipes. All components have at least 2 connectors. Each connector defines the expected direction of flow (in or out) and the connected component's tag, among other properties not relevant to this project.

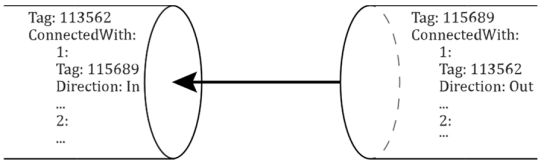

Fig. 4 - Example of connector definition.

The toolchain loops through all components and for every ingoing connector, a corresponding Modelica connector will be established. Only ingoing connections are considered to avoid duplicate connectors. In Modelica, components are connected through *ports*. The name of the ports vary, depending on the component class, and hence, they are stored in the components during data enrichment.

Since the BIM data does not support definition of control logics, default controls are assumed for the components that need control. E.g., all components mapped to the MotorValve class (see Tab. 2) are assumed to be controlling flow in a heating coil. Thus, these all take the measured ventilation supply temperature as an input for the processed variable for the PI controller.

For each component, a component model template is instantiated and added to a text string, containing the model information. After looping through all components, both for class mapping and connection establishment, the text string contains the entire model. The model is saved in a temporary model file for simulation.

3.3. Simulation and results reading

When simulating Modelica models, the models must first be compiled to a machine-readable executable and after that simulated. In the toolchain, this is done through *BuildingsPy* [17], which interfaces the commercial Modelica simulation environment Dymola. BuildingsPy takes the file path to the simulation file and simulation parameters, such as duration and solver, as parameters and parses these to Dymola. After simulation, the results are read through BuildingsPy, and the results are parsed to a JSON format and returned to the database. Since the simulations return many results for each component, the wanted results for each component class are defined in a specific file, and only these results are sent back to the database.

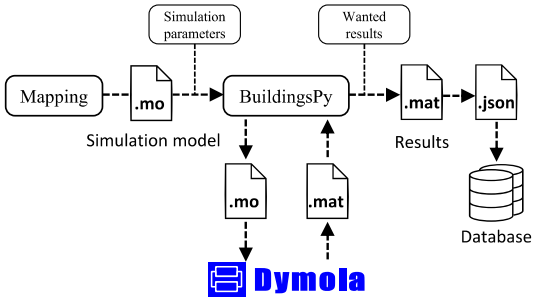

Fig. 5 - Simulation procedure.

4. Testing

To ensure that the tool is usable, it was tested on a small heating system model. The test did not focus on assessing the system's performance but merely to check whether the tool works, and the obtained results make sense and are of interest.

4.1. Test case description

The test case system, depicted in Fig. 6, consists of a heating coil and a radiator, each in separate loops. The main pump supplies flow to both loops, and a secondary pump is connected to the heating coil. To simulate the dynamic behavior, both the heating coil and the radiator are connected to a generic room with the parameters given in Tab. 3. For simplicity, only heat loss through the walls and window was considered.

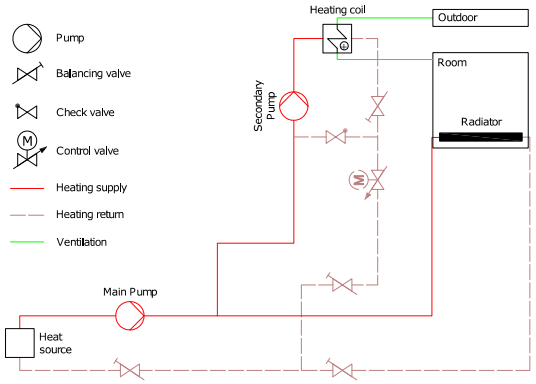

Fig. 6 - Heating system with one radiator and one heating coil used for testing the tool.

The radiator is controlled by a thermostatic radiator valve (TRV), connected to the room temperature. The TRV is not depicted in figure Fig. 6, since it is considered a part of the radiator. To control the heating output of the heating coil, a PI controller adjusts the control valve position to change the heating supply temperature. This is based on the ventilation supply temperature to the room, which has a constant setpoint. For simplicity, both pumps are operated with constant speed, although under normal circumstances, such pumps would either be controlled for constant or proportional pressure. The heat source is not considered, and it is assumed that

it supplies water at a fixed temperature of 70 °C.

Tab. 3 - Room parameters

Parameter	Value	
Floor area	30	m^2
Wall area	60	m^2
Window area (south)	4	m^2
Total envelope area	66	m^2
Wall U-value	0.27	$W/(m^2 K)$
Window U-value	1.31	$W/(m^2 K)$
Window g-value	0.73	[-]

4.2. Simulation setup

The simulation was done for the first five days of weather data for Chicago, USA. In this period, the temperature ranges between -15 °C and 0 °C. Hence, it is possible to see the dynamic effects in varying external temperatures, including maximum capacity conditions.

To initialize the solution, in addition to the Modelica initialization, one preceding day was simulated before the first day of the year.

4.3. Simulation results

By simulating the system through the toolchain, the overall temperature curves in Fig. 7 were achieved. Fig. 7 shows the temperature of the room and the ventilation supply air, compared to the external temperature. It is seen that the room temperature is stable, but that there is an offset from the setpoint. This offset is caused by the TRV, which in Modelica is modeled as a proportional controller, which will normally introduce an offset. Hence, this behavior is expected. The supply temperature is stable with no offset since it is controlled by a PI-controller.

In Fig. 8, the return temperature for both loops and the full system is seen. The return temperatures are stable around 30 °C, with a slight tendency to increase with lower external temperatures, as expected.

5. Discussion and gained experiences

The development process highlighted several points of attention during the mapping process. Most importantly, all needed data for the considered components must be available. If the parameters in Tab. 2 are not available for all components, the simulations will fail. This puts high demands on the level of information in BIM, but with tools for system dimensioning and component databases, the amount of information is not unrealistic. The possibility to perform detailed simulations of, e.g., return temperatures may even motivate designers to populate BIM models with more information on hydronic components.

In the presented work, controls were handled by applying a set of assumptions suitable for the specific test case. To work expand the work to larger systems, unambiguous control relations must be established in the database. This can be handled in two ways. Either the existing data format is extended with

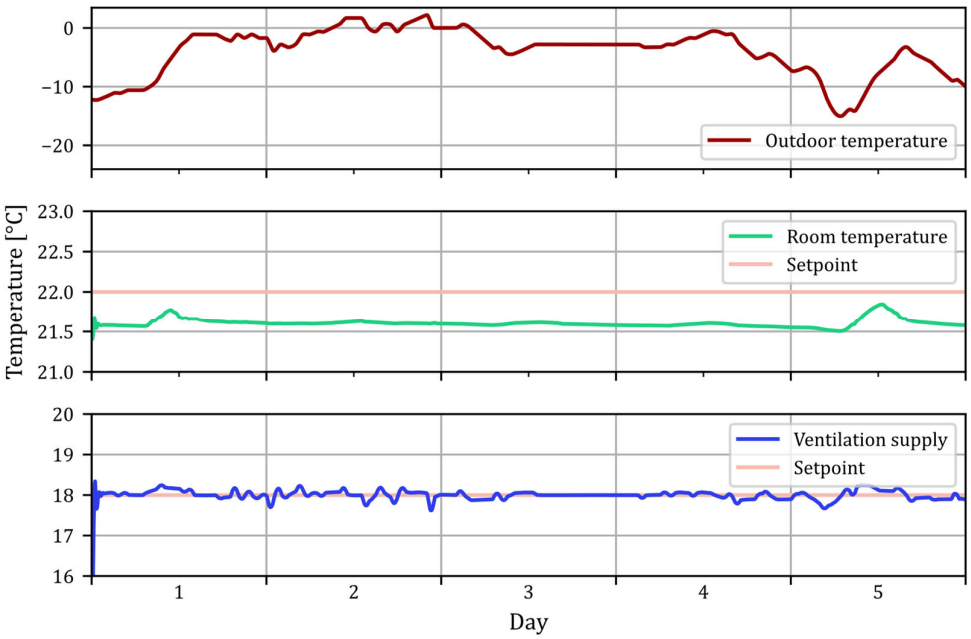

Fig. 7 - Results for the outdoor (t_{ext}), room (t_{room}) and ventilation supply (t_{vent}) temperature, including setpoints (SP).

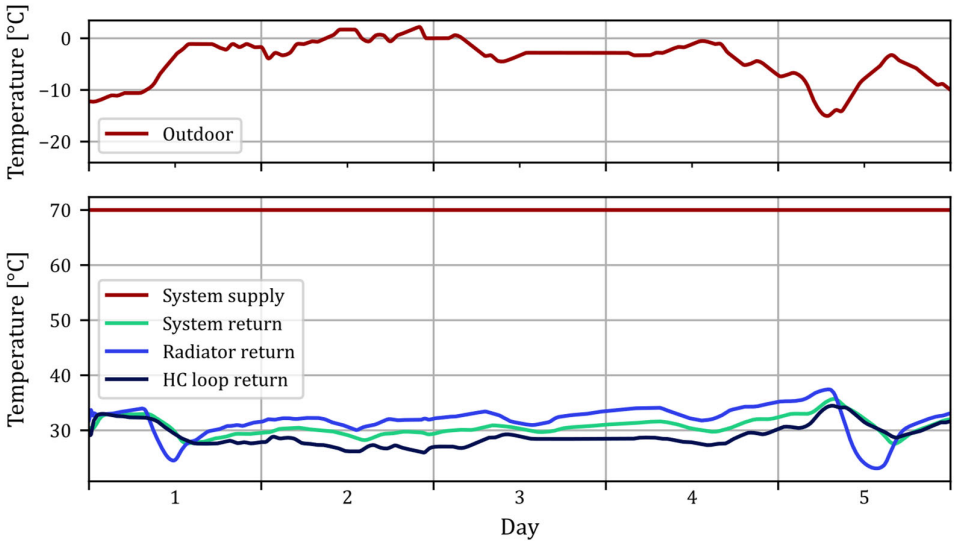

Fig. 8 – Return temperatures for heating coil loop (HC), radiator loop and for the entire system.

component parameters for controls, such as process variables, setpoints, etc., or a new data format for controls must be used. Work towards digitizing control information is in growth with several projects under development [11, 18]. Utilizing these existing frameworks to represent control logic in a database seems like a viable solution, but for simpler systems, the simple approach of added attributes may prove sufficient.

The connection between end units and rooms is a vital piece of information to correctly simulate the systems. Creating a link between end units and rooms may be a simple process in the BIM domain, but it requires additional mapping modules to include the rooms in the simulations.

6. Conclusion and future work

In this paper, it was proved that it is possible to create a tool for the simulation of heating systems based on BIM models. The simulations provided detailed and vital information on the performance of the individual components in the testing case. This showed the value of such simulations that are usually too time-consuming to be made. While the presented results may be trivial for a system as small as the test case, the same analysis for larger system will uncover results that are difficult for normal practitioners to quantify.

Several important attention points for a larger implementation were identified, the biggest being the lack of representation of controls in BIM models. These points resulted in several assumptions built into the tool, especially regarding control strategies. These assumptions mean that the tool is less flexible to different system configurations. By extending the data format to include the needed information on controls, the tool can easily be modified to simulate larger systems with both heating, ventilation, and cooling systems. When this work is done, the full models can be used in fault detection, detailed analysis of the dynamic effects of coupling the systems, etc.

7. Acknowledgement

This work was funded by the national IFD grant J.nr. 8090-00046B for the project “HEAT 4.0 - Digitally supported Smart District Heating”, Elforsk grant 352-042, IFD grant J.nr. 9065-00266A for the project “Virtual Commissioning in Building Services” and “Databaseret energistyring i offentlige bygninger”, EU Interreg-ÖKS 2020-2022.

8. References

[1] de Wilde P. The gap between predicted and measured energy performance of buildings: A framework for investigation. Automation in Construction. 2014;41:40–9.

[2] Menezes AC, Cripps A, Bouchlaghem D, Buswell R. Predicted vs. actual energy performance of non-domestic buildings: Using post-occupancy evaluation data to reduce the performance gap. Applied Energy [Internet]. 2012;97:355–64. Available from: http://dx.doi.org/10.1016/j.apenergy.2011.11.075

[3] Virta M. Initial Commissioning. In: HVAC Commissioning Guidebook [Internet]. REHVA; 2021. p. 13–42. Available from: https://app.knovel.com/hotlink/khtml/id:kt011ZBWH4/hvac-commissioning-process/initial-commissioning

[4] Wetter M. Modelica-based modelling and simulation to support research and development in building energy and control systems. Journal of Building Performance Simulation [Internet]. 2009 Jun 19;2(2):143–

61. Available from: https://www.tandfonline.com/doi/full/10.1080/19401490902818259

[5] Integrated Environmental Solutions | IES. Iesve 2021 [Internet]. Available from: https://www.iesve.com/ve2021

[6] EQUA Simulation Technology Group. IDA Indoor Climate and Energy [Internet]. 2014. Available from: http://www.equa-solutions.co.uk/de/software/idaice

[7] Mattsson SE, Elmqvist H. Modelica - An International Effort to Design the Next Generation Modeling Language. IFAC Proceedings Volumes [Internet]. 1997 Apr;30(4):151–5. Available from: https://linkinghub.elsevier.com/retrieve/pii/S1474667017436287

[8] Kim JB, Jeong W, Clayton MJ, Haberl JS, Yan W. Developing a physical BIM library for building thermal energy simulation. Automation in Construction. 2015;50(C):16–28.

[9] Nytsch-Geusen C, Inderfurth A, Kaul W, Mucha K, Rädler J, Thorade M, et al. Template based code generation of Modelica building energy simulation models. In: Proceedings of the 12th International Modelica Conference, Prague, Czech Republic, May 15-17, 2017. Linköping University Electronic Press; 2017. p. 199–207.

[10] Andriamamonjy A, Saelens D, Klein R. An automated IFC-based workflow for building energy performance simulation with Modelica. Automation in Construction. 2018;91(March):166–81.

[11] Terkaj W, Schneider GF, Pauwels P. Reusing domain ontologies in linked building data: The case of building automation and control. CEUR Workshop Proceedings. 2017;2050.

[12] Janowicz K, Rasmussen MH, Lefrançois M, Schneider GF, Pauwels P. BOT: The building topology ontology of the W3C linked building data group. Janowicz K, editor. Semantic Web [Internet]. 2020 Nov 19;12(1):143–61. Available from: https://www.medra.org/servlet/aliasResolver?alias=iospress&doi=10.3233/SW-200385

[13] Bew M, Richards M. BIM Maturity Model. Paper presented at the Construct IT Autumn 2008 Members' Meeting. 2008;

[14] Kukkonen V, Kücükavci A, Seidenschnur M, Rasmussen MH, Smith KM, Hviid CA. Proposing a Semantic Web Ontology to Support Flow System Descriptions from Design to Operation of Buildings. Automation in Construction. 2021;134.

[15] Flow Systems Ontology Web Page [Internet]. 2021. Available from: https://alikucukavci.github.io/FSO/

[16] Wetter M, Zuo W, Nouidui TS, Pang X. Modelica Buildings library. Journal of Building Performance Simulation [Internet]. 2014 Jul 4;7(4):253–70. Available from: http://www.tandfonline.com/doi/abs/10.1080/19401493.2013.765506

[17] Lawrence Berkeley National Laboratory - Simulation Research Group. BuildingsPy [Internet]. 2021 [cited 2021 May 4]. Available from: https://simulationresearch.lbl.gov/modelica/buildingspy/

[18] Wetter M, Ehrlich P, Gautier A, Grahovac M, Haves P, Hu J, et al. OpenBuildingControl: Digitizing the control delivery from building energy modeling to specification, implementation and formal verification. Energy [Internet]. 2022 Jan 1 [cited 2021 Sep 14];238:121501. Available from: https://doi.org/10.1016/j.energy.2021.121501

Data Statement

The datasets generated during and/or analysed during the current study are not publicly available because of ongoing development but are/will be available upon request.