2020 Building Performance Modeling Conference and
SimBuild co-organized by ASHRAE and IBPSA-USA

LIFTING THE GARAGE DOOR ON SPAWN, AN OPEN-SOURCE BEM-CONTROLS ENGINE

Michael Wetter[1], Kyle Benne[2], Antoine Gautier[1], Thierry S. Nouidui[3], Agnes Ramle[4], Amir Roth[5], Hubertus Tummescheit[6], Stuart Mentzer[7] and Christian Winther[4]

[1]Lawrence Berkeley National Laboratory, Berkeley, CA, USA
[2]National Renewable Energy Laboratory, Golden, CO, USA
[3]The United African University of Tanzania, Dar Es Salaam, Tanzania
[4]Modelon AB, Lund, Sweden
[5]US Department of Energy, Washington, DC, USA
[6]Modelon AB, Hartford, CT, USA
[7]Objexx Engineering, Boston, MA, USA

ABSTRACT

Spawn is the latest whole-building energy simulation engine developed by the US Department of Energy, National Labs and industry. Whereas EnergyPlus was designed as a successor to DOE-2, Spawn is not a direct successor of–nor is it intended as an imminent replacement for–EnergyPlus. Instead, Spawn reuses parts of EnergyPlus while supporting new use cases in HVAC and controls.

Spawn is intended to provide several capabilities that significantly advance beyond EnergyPlus. It is intended to support the evaluation of novel HVAC and district energy systems in a more physically realistic way. Critically, it can model control in a physically realistic way, using portable specifications that can be compiled for execution on control platforms. Spawn is also intended to support co-simulation in an intrinsic way to enable integration with third-party models.

This paper describes the software architecture of Spawn from model authoring to compilation and simulation. It explains how Spawn reuses the envelope and daylighting modules of EnergyPlus and couples them to HVAC and control models from the Modelica Buildings Library using the Functional Mockup Interface (FMI) standard. It presents a number of examples that: i) validate Spawn's coupled simulation approach by comparing its results to those of EnergyPlus, ii) illustrate the Spawn methodology for modeling and simulating HVAC systems, and iii) evaluate the performance of Spawn's Quantized State System (QSS) time integration algorithms.

INTRODUCTION

EnergyPlus is DOE's open-source whole-building energy modeling (BEM) program (Crawley et al. 2001). It has a broad range of capabilities and serves as the basis for both energy-efficiency codes and a growing ecosystem of commercial software.

EnergyPlus was designed to calculate annual energy use and to support the traditional energy-efficiency oriented BEM use cases that build on such calculations: new construction and retrofit design, energy-efficiency measure (EEM) evaluation, performance-path code compliance, performance documentation for LEED and utility incentives, and building stock modeling for code, program and product development. It fulfills this mission effectively. However, EnergyPlus' traditional structure makes it difficult to evaluate HVAC control sequences and to integrate BEM with control workflows. Controls are an increasingly important part of building energy-efficiency and building-to-grid integration – simply by changing the control sequence, building HVAC energy use can be reduced by 30% (Wetter et al. 2018). Aligning control sequences used in simulation with those actually implemented in buildings – ideally supporting an automated, digitized control delivery process in which sequences can be ported directly from simulation to execution – can help capture these potential savings. This is difficult to do in EnergyPlus. EnergyPlus is organized around primary and secondary HVAC loop simulation and its load-based models have inputs and outputs that are semantically different from actuator commands and sensor signals. Moreover, its numerical methods cannot handle fast dynamics, events, certain sampled systems and finite state machines. This poses fundamental difficulties for simulation of actual control other than rule-based supervisory control sequences.

The Spawn project attempts to address these challenges and enable integrated BEM-control workflows. Spawn leverages two open standards, the Modelica language (Mattsson, Otter, and Elmqvist 1999) and the Functional Mockup Interface (FMI) for co-simulation and model-exchange (Blochwitz et al. 2011). Spawn reuses EnergyPlus' envelope heat transfer, daylighting and internal load calculation modules. It couples these to HVAC

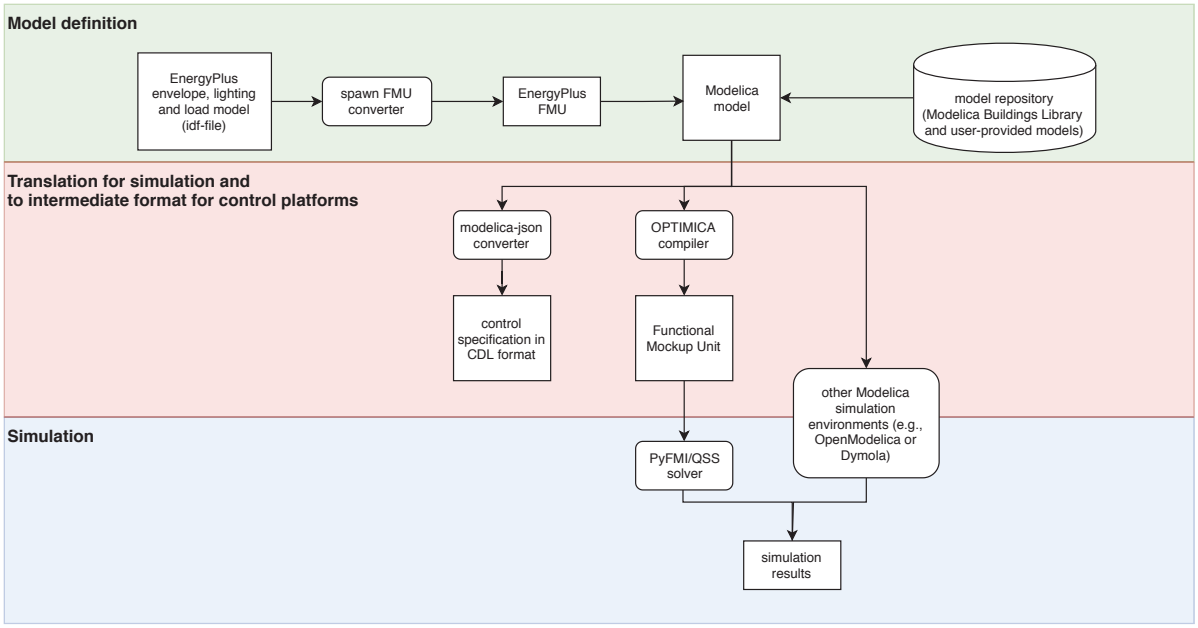

Figure 1: Architectural diagram of Spawn.

and control models from the Modelica Buildings Library (Wetter et al. 2014). Spawn control sequences are expressed as they are implemented in the real world using a subset of Modelica called the Control Description Language (CDL) (Wetter, Grahovac, and Hu 2018). A library of CDL control sequences as well as translators from CDL to physical control platform languages like Automated Logic's EIKON are currently under development as part of OpenBuildingControl (`https://obc.lbl.gov`), a companion project to Spawn.

Spawn compiles Modelica HVAC and control models for simulation using Modelon's OPTIMICA Compiler Toolkit, and couples them to EnergyPlus' envelope and internal loads models using an FMI interface. The internal use of standard co-simulation protocols creates a modular architecture that allows Spawn to integrate externally developed component and system models, including data-driven machine learning models, and to export models for integration with other simulation tools or for use as digital twins during operation.

ARCHITECTURE

Figure 1 shows the Spawn software architecture and its key underlying technologies and components. This section describes these components – EnergyPlus, the Modelica Building Library, the OPTIMICA Modelica compiler and toolkit, and the PyFMI solver and time integrator – and their integration.

EnergyPlus

Spawn reuses EnergyPlus to model both internal loads and envelope heat and light transfer, including electric lighting, shading, daylight transport, solar heat gains, ground heat transfer, surface conduction, surface-to-surface radiation, and convective heat flow from interior surfaces to room air. Room-air heat and mass balance including inter-zone air-exchange, HVAC operation and control are simulated in Modelica as shown in Figure 2.

There are three reasons for this choice of partitioning, i.e., simulating room air balance and inter-zone air-exchange in Modelica in addition to HVAC airflow. First, air temperature has fast dynamics compared to heat conduction in the building fabric, and so simulating it in Modelica enables the use of variable time step solvers and reduces the number of synchronization steps between EnergyPlus and Modelica. Second, because HVAC system airflow, room air, multi-zone air exchange and infiltration are tightly coupled–HVAC system airflow rate is typically computed based on pressure distributions in the HVAC system–simulating the three together reduces cross-coupling and yields better performance. Third, this separation simplifies drop-in substitution of the current 1D room air model with the 3D computational fluid dynamics (CFD) model available in the Modelica Buildings Library (Zuo et al. 2016).

This partitioning reuses EnergyPlus' 3D envelope model, including shading, daylighting, conduction, and detailed

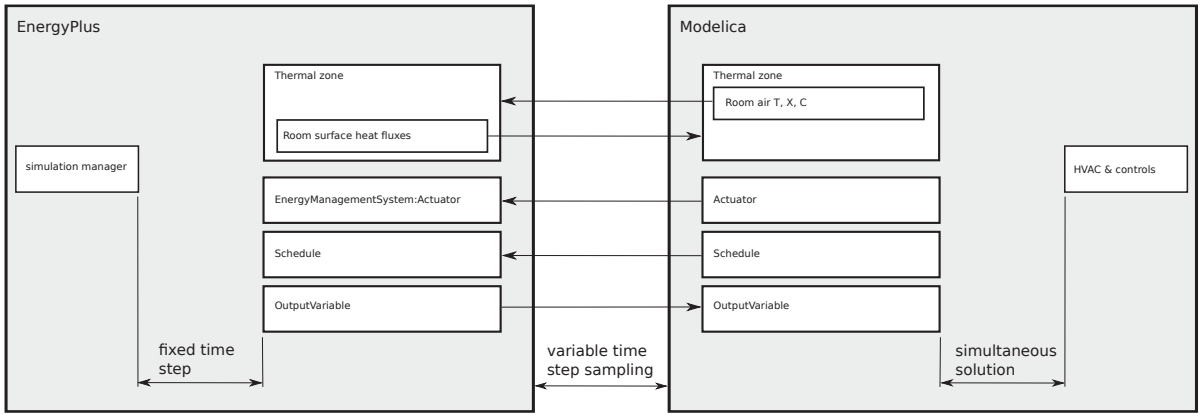

Figure 2: Partitioning of the envelope, room and HVAC model.

radiative heat exchange. HVAC and control models lead to sparse equation systems with coupled fast and slow dynamics that benefit from a Modelica implementation with its symbolic optimization. In contrast, envelope and lighting models benefit from special structured code such as polygon projections and time series solutions that are less amenable to symbolic optimization and therefore less likely to benefit from a Modelica reimplementation. Reusing EnergyPlus for these leverages existing infrastructure and allows us to focus our efforts in HVAC and control. And although conceptually significant, the modifications to the underlying EnergyPlus engine are surgical and support improved functionality and integration in mainstream EnergyPlus applications in addition to enabling Spawn.

Enabling coupling between EnergyPlus and Modelica required some modifications to EnergyPlus itself. The default workflow for EnergyPlus is to specify a run period in the input (IDF) file and to output data in any one of several formats (e.g., CSV, SQL) at the end of the simulation. EnergyPlus does have an External Interface feature that enables data-exchange on a time-step basis, but this feature does not provide the granular access to EnergyPlus needed by Spawn. It also uses TCP sockets for communications which cause additional overhead. To accommodate Spawn, EnergyPlus was refactored to allow more granular simulation control and data exchange with lower overhead. In lieu of sockets, a C++ API that relies on direct memory access was added. The new API provides functions to advance the simulation in time, to set inputs and read outputs within each timestep, and to wholesale bypass the HVAC and control simulation algorithms, allowing EnergyPlus to be used strictly as a heat balance and daylight calculation engine. Work is ongoing to expose output variables and schedules via the API. The API executes on a separate thread from EnergyPlus, with blocking at key points to manage simulation advance, allowing the latter to execute much as it always has.

This modified version of EnergyPlus is compiled into a library with the API exposed for external calls. At start-up, Modelica writes an interface variable specification and invokes a script. This script reads this specification and the IDF file referenced in this specification, and generates a Functional Mockup Unit (FMU) that contains everything needed by the Modelica model to dynamically link to and execute EnergyPlus.

Modelica Buildings Library

A critical building block of Spawn is the Modelica Buildings Library, an open-source model library that is based on the Modelica IBPSA Library (Wetter et al. 2015) that arose from IEA EBC Annex 60 (Wetter and van Treeck 2017) and is continuing to be developed in IBPSA Project 1 (Wetter et al. 2019b).

The Modelica Buildings Library contains about 1,500 models and functions for building and district HVAC systems and electrical AC and DC systems (Wetter et al. 2014; Bonvini, Wetter, and Nouidui 2014). It contains control components and sequences, including sequences from ASHRAE Guideline 36 (Wetter et al. 2018), implemented in the Control Description Language (CDL) (Wetter, Grahovac, and Hu 2018). It also contains components that link to EnergyPlus, and that support co-simulation with Python applications, e.g., for embedding data-driven models into whole-building simulation.

Spawn also allows users to use models from other Modelica libraries, e.g., for detailed dynamic vapor compression modeling (Tummescheit, Eborn, and Prölss 2005), as well as user-provided Modelica models.

For Spawn, we developed a Modelica package with models that specify the EnergyPlus IDF file name and that

synchronize, during the time stepping, variables between Modelica and the EnergyPlus room heat balance, EnergyPlus output variables and EnergyPlus schedules and actuators. These models automatically configure and execute the co-simulation, avoiding any need for manual co-simulation setup by the user. The co-simulation can be done with one EnergyPlus model when simulating an individual building, or with any number of EnergyPlus models when simulating a district energy system.

Thus, a Spawn model consists of a Modelica model and the EnergyPlus files referenced by the Modelica model.

OPTIMICA and PyFMI

Spawn Modelica models can be translated and simulated with any tool that supports the Modelica standard. In particular, Spawn uses the OPTIMICA Compiler Toolkit, a state-of-the-art Modelica/FMI based simulation platform. The OPTIMICA Modelica compiler is based on the latest technology and standard practices for translating models defined as differential algebraic equation systems into an ordinary differential equation system suitable for FMI. In order to generate efficient code and to achieve high-performance simulations, it performs numerous computer algebra operations on the Modelica model such as automatic differentiation, block lower triangularization, tearing and index reduction (see e.g., Cellier and Kofman 2006). As part of Spawn, the OPTIMICA translation and simulation back-end will be distributed at no cost to application developers or end users.

The Spawn FMU created by OPTIMICA is simulated and time-integrated using the open-source solver PyFMI.

Quantized-State System (QSS)

Traditional continuous time integration algorithms advance the entire state vector, i.e., every variable, in lock-step. At a high level the process is as follows. A candidate time step is first chosen. A polynomial approximation is used to determine the values of all state variables at the end of this time step. If the error in any of the variables exceeds a threshold, the process is repeated with a shorter time step. If a state event – such as when a thermostat switches – occurs, iteration is used to determine the timing of the event. Post event, the higher-order terms of the polynomial approximation become invalid and time integration is restarted with a low-order polynomial approximation until higher-order approximations can be created in successive time-steps. This process is expensive, especially for building HVAC simulation which typically contains both fast and slow transients as well as controls with many timers and switches.

For Spawn, we are developing a solver based on Quantized State System (QSS) integration methods (Zeigler and Lee 1998; Cellier and Kofman 2006; Kofman 2003; Bergero et al. 2018) that promise to be efficient for such systems. The main difference between QSS and traditional methods is that QSS advances each state variable independently. Rather than discretizing time, the state vector is discretized instead, and all communication occurs at discrete events. QSS tracks dependencies between variables – using the right hand sides of equations – and invokes re-evaluation of dependent variables if changes in independent variables exceed specified thresholds. Generally speaking, in real configurations, dependencies are localized – by physics – and events are handled without affecting the entire state vector.

For Spawn we have built a QSS solver, extended the FMI specification, and integrated both with OPTIMICA. The solver supports QSS methods of up to 3rd order, including linearly implicit methods (LIQSS) for stiff systems. We extended OPTIMICA to provide additional structural information about the equations required for the efficient use of QSS, and to provide more granular access to the FMU. We have verified the solver's performance against a canonical implementation in a non-FMU context, and it is behaving as expected. We will verify its performance with large FMU models when certain extensions to the FMI API are implemented. Upcoming development of the Spawn QSS solver will focus on refinements of zero-crossing algorithms, performance assessment and improvement, and parallelization.

EXAMPLES

We present a few simulation examples that validate free floating room air temperatures computed by Spawn, show how to link Modelica to EnergyPlus using Spawn and demonstrate simulation of a model with the QSS solver.

Validating EnergyPlus-Modelica Coupling

This example validates the coupling of Modelica via the room-air heat and mass balance to the EnergyPlus envelope model. It compares free-floating room temperatures computed by Modelica using the model shown in Figure 3 against those computed by a conventional EnergyPlus simulation. The model here is the DOE small office reference building for Chicago, IL, which is a single story building with five zones plus attic. In Figure 3, the block labeled `building` contains the IDF and weather file specification, and each blue icon is a Modelica thermal zone model that connects to the corresponding EnergyPlus zone model. The model is available from the Modelica Buildings Library, model `EnergyPlus.Validation.RefBldgSmallOffice`.

For this experiment, we used for Spawn the OPTIMICA compiler and the CVode solver with a tolerance of 10^{-6}, and for EnergyPlus we used a 15 minute time step. Figure 4 shows temperatures for January 6 for the South, West, and Core zones. As shown, the results agree within the solver tolerance.

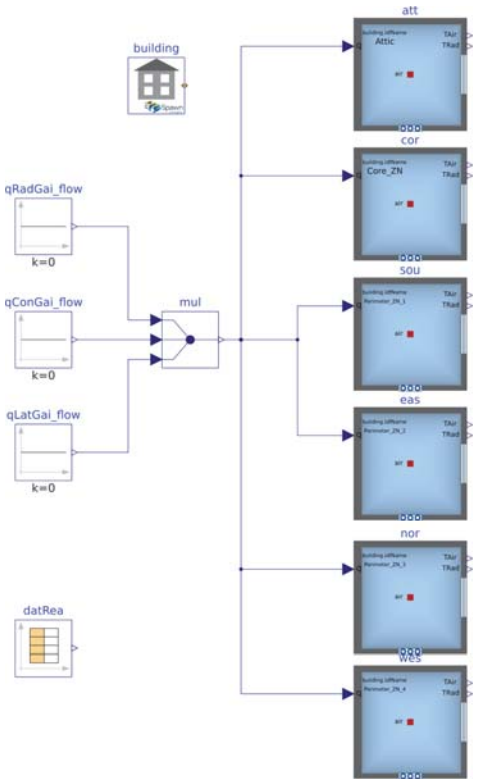

Figure 3: Modelica model of the office building used to validate the EnergyPlus-Modelica coupling.

Adding a Simple HVAC System

In this section, we add a simple HVAC system. The example is also available from the Modelica Buildings Library, model `EnergyPlus.Validation.OneZoneWithControl`.

Figure 5 shows the Modelica schematic model view. The `building` block contains the IDF and weather file specification. The `zon` block is the Modelica model for the thermal zone which connects to the corresponding thermal zone of the EnergyPlus FMU. It outputs the room air temperature `TAir`. It also has fluid connections for the HVAC system. These are connected to a simple fluid loop with a fan, a heater, and a PI controller configured to meet the zone setpoint temperature.

Figure 6 shows the room air temperature T_{air}, the temperature set point T_{set} and the resulting control signal y for one day of simulation.

QSS

In this example, we tested the development version of the 2nd order QSS2 solver with an FMU generated by OPTIMICA for the model shown in Figure 7. The model is identical to `Buildings.Fluid.Examples.SimpleHouse` except that we added the time sampler `Tzon` with a sample period of 2 minutes. We added this sampler because many building control sequences contain time samplers and because frequent events impact the performance of ordinary differential equation solvers. The model consists of a room air volume (`zone`) with fresh air supply and heat recovery (`hexRec`), heat conduction and storage in a one-node wall model, and a water loop with a heater (`heaWat`), radiator (`rad`) and circulation pump. The heater and pump are controlled using an on/off controller (`hysRad`).

Figure 8 shows a few hours of simulation of the state variables for the zone air T_{zon} as computed by the 2 minute time sampler, the thermal mass of the wall T_{wal} and the temperatures of the five water control volumes of the radiator T_{rad}. The solid lines are computed with the CVode solver. For easier readability, we indicated every 5th time instant when the QSS solver updates the respective state variable with a black dot. The figure illustrates the frequent updates during fast transients, the time scale separation between fast and slow transients such as T_{rad} and T_{wal}, and the fact that despite the 2 minute sampling of T_{zon}, the trajectories of the other states are updated less often. QSS interpolates the sample based on a polynomial expression of the trajectory T_{zon}, and only updates elements of the downstream state $\{T^i_{rad}\}^5_{i=1}$ when they change by more than a tolerance. In contrast, but not visualized here, CVode makes a time step at each sample instant and restarts after this time event with a small step. QSS has comparable accuracy to CVode. CVode takes, on average for an annual simulation, a step every 28 seconds, because the 2 minute sampling prevents it from enlarging its step. In contrast, QSS takes, an average step length of 55 seconds, with shorter steps during fast transients.

Although QSS performs fewer evaluations, it currently has a 5% higher computing time than CVode. We are currently implementing automatic differentiation – which reduces the number of calls to the FMU and increases the accuracy of the trajectory prediction – as well as sparse evaluation within OPTIMICA. We therefore expect these to reduce QSS computing time significantly.

A key benefit of QSS is that its computing time scales linearly in the number of states, whereas that of traditional continuous time integration methods scales super-linearly. We expect the advantage of QSS over traditional methods to grow for models larger than this small example. QSS performance evaluation and refinement on a larger class of building models is subject of further research.

WORKFLOW INTEGRATION

EnergyPlus has a by-now familiar interface that consists of text-based input and outputs in various formats such as CSV, SQL and HTML. A number of tools have been written directly to this interface. The OpenStudio Software Development Kit (SDK) simplifies application inte-

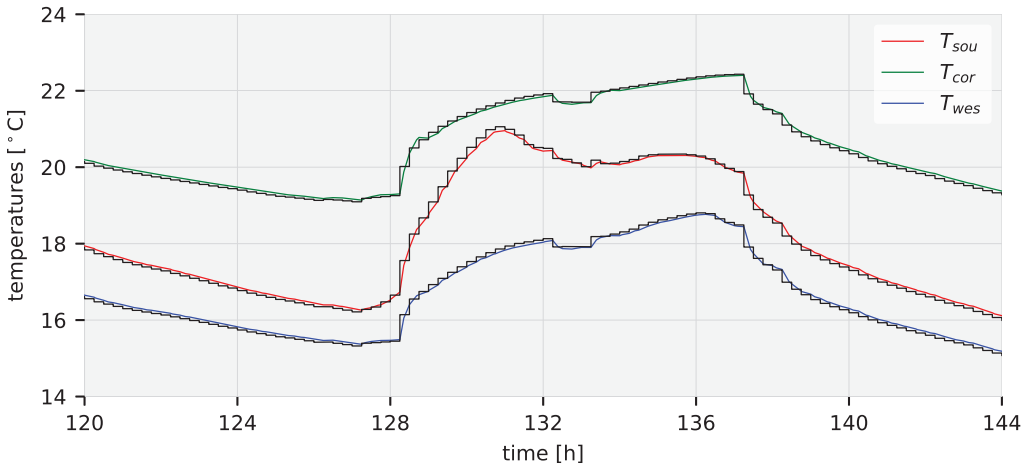

Figure 4: Free floating room air temperatures for south, core and west zones for the DOE small office reference building for Chicago, IL, shown in Figure 3. Black lines are from legacy EnergyPlus, and colored lines are from Spawn.

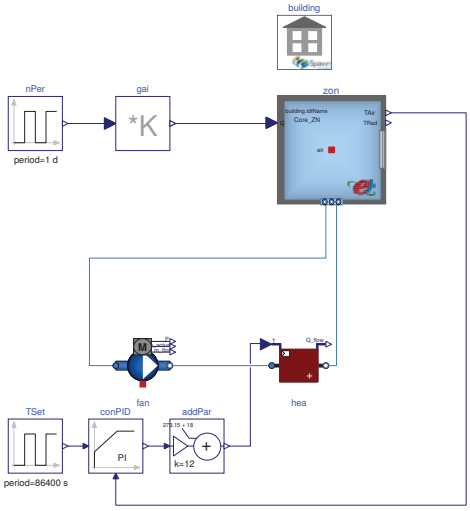

Figure 5: Coupling a simple Modelica HVAC system model to an EnergyPlus envelope model.

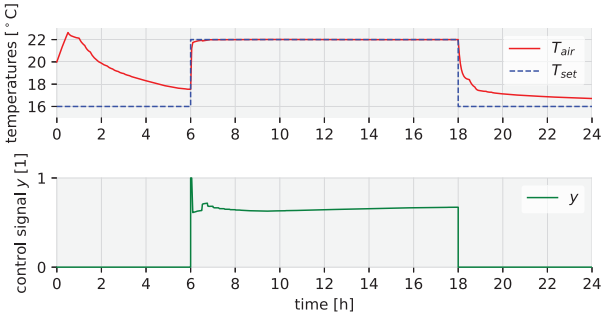

Figure 6: Room air temperature T_{air}, room air temperature set point T_{set} and resulting control signal y for the model in Figure 5.

gration by providing programmatic access to EnergyPlus input and output. OpenStudio also supports automation in the form of scripts, which in turn enables systematic large-scale analysis. Spawn can reuse the portions of OpenStudio that correspond to weather, schedules, internal loads, geometry, constructions, and space and zone assignment.

Similar infrastructure is needed for HVAC and controls. Here, it is more difficult to directly retarget OpenStudio to Spawn because of the desire to support an open-ended range of component models, system configurations, control strategies, and free-form Modelica code. We have developed a JSON schema that describes Modelica components, their parameters, and their connectivity. It also supports templates for standard system configurations. The schema is designed to support the development of HVAC and control model authoring and editing tools as well as scripting and automation. We have developed a `modelica-json` utility that parses Modelica libraries and generates Modelica-JSON. This utility is under development at `https://github.com/lbl-srg/modelica-json`.

We are also currently developing specifications for a Modelica-JSON based HVAC and controls authoring and editing tool. Of course, general purpose Modelica authoring environments, such as Dymola, OpenModelica's OMEdit or Modelon's IMPACT, can also be used to create and edit Spawn HVAC and control models.

In addition to supporting model authoring, the Modelica-JSON format is also used to translate control sequences to commercial control platforms. A prototype translator for

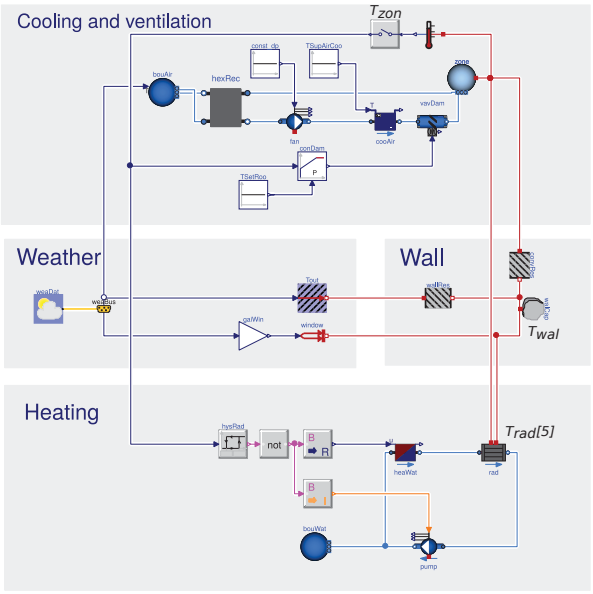

Figure 7: Modelica model used for the QSS example.

Automated Logic's EIKON language is currently in development. Modelica-JSON also supports the generation of controls documentation that can be used in control submittals for bidding and specification of control sequences and documentation for operators. Also, software has been developed that verifies that the actual implemented control sequence computes a control response that is identical, within a user-specified tolerance, to a digital specification of the control sequence that is used in Spawn (Wetter et al. 2019a). Also in development is a Modelica-JSON export to a BRICK semantic model (Balaji et al. 2018), which will facilitate the setup of system-level fault detection and diagnostics algorithms and energy analytics software.

CONCLUSIONS AND FUTURE WORK

Spawn was conceived a number of years ago and has been under development since. At this time, the basic end-to-end architecture is in place as are all of the major components.

Work is proceeding along three lines. On content, we are continuing to build functionalities of component and system model libraries, including the Modelica Buildings Library. On compilation and simulation, the focus is on enhancements for faster translation of large models and further development and testing of QSS. Building Spawn around the open standards Modelica and FMI allows us to leverage the international investments in IBPSA Project 1 and to collaborate with industry on technology development and integration. Finally, we are continuing to develop infrastructure for integrating the developing capabilities of Spawn into the existing BEM ecosystem and into control workflows, bridging these two domains.

To access the development version of Spawn, visit `https://lbl-srg.github.io/soep/`.

ACKNOWLEDGMENTS

This research was supported by the Assistant Secretary for Energy Efficiency and Renewable Energy, Office of Building Technologies of the U.S. Department of Energy, under Contract No. DE-AC02-05CH11231.

This work emerged from the IBPSA Project 1, an international project conducted under the umbrella of the International Building Performance Simulation Association (IBPSA). Project 1 will develop and demonstrate a BIM/GIS and Modelica Framework for building and community energy system design and operation.

REFERENCES

Balaji, B., A. Bhattacharya, G. Fierro, J. Gao, J. Gluck, D. Hong, A. Johansen, J. Koh, J. Ploennigs, Y. Agarwal, M. Bergés, D. Culler, R. K. Gupta, M. Baun Kjærgaard, M. Srivastava, and K. Whitehouse. 2018. "Brick : Metadata schema for portable smart building applications." *Applied Energy* 226:1273 – 1292.

Bergero, Federico Martín, Francesco Casella, Ernesto Kofman, and Joaquín Fernández. 2018. "On the efficiency of quantization-based integration methods for building simulation." *Building Simulation* 11 (2): 405–418 (April).

Blochwitz, T., M. Otter, M. Arnold, C. Bausch, C. Clauß, H. Elmqvist, A. Junghanns, J. Mauss, M. Monteiro, T. Neidhold, D. Neumerkel, H. Olsson, J.-V. Peetz, and S. Wolf. 2011, March. "The Functional Mockup Interface for Tool independent Exchange of Simulation Models." *Proc. of the 8-th Int. Modelica Conf.* Modelica Association, Dresden, Germany.

Bonvini, Marco, Michael Wetter, and Thierry Stephane Nouidui. 2014, September. "A Modelica package for building-to-electical grid integration." *5th BauSim Conf.* IBPSA-Germany, Aachen, Germany, 6–13.

Cellier, François E., and Ernesto Kofman. 2006. *Continuous System Simulation*. Springer.

Crawley, Drury B., Linda K. Lawrie, Frederick C. Winkelmann, Walter F. Buhl, Y. Joe Huang, Curtis O. Pedersen, Richard K. Strand, Richard J. Liesen, Daniel E. Fisher, Michael J. Witte, and Jason Glazer. 2001. "EnergyPlus: creating a new-generation building energy simulation program." *Energy and Buildings* 33 (4): 443–457.

Kofman, Ernesto. 2003. "Quantization-Based Simulation of Differential Algebraic Equation Systems." *SIMULATION* 79 (7): 363–376.

Mattsson, S. E., M. Otter, and H. Elmqvist. 1999, De-

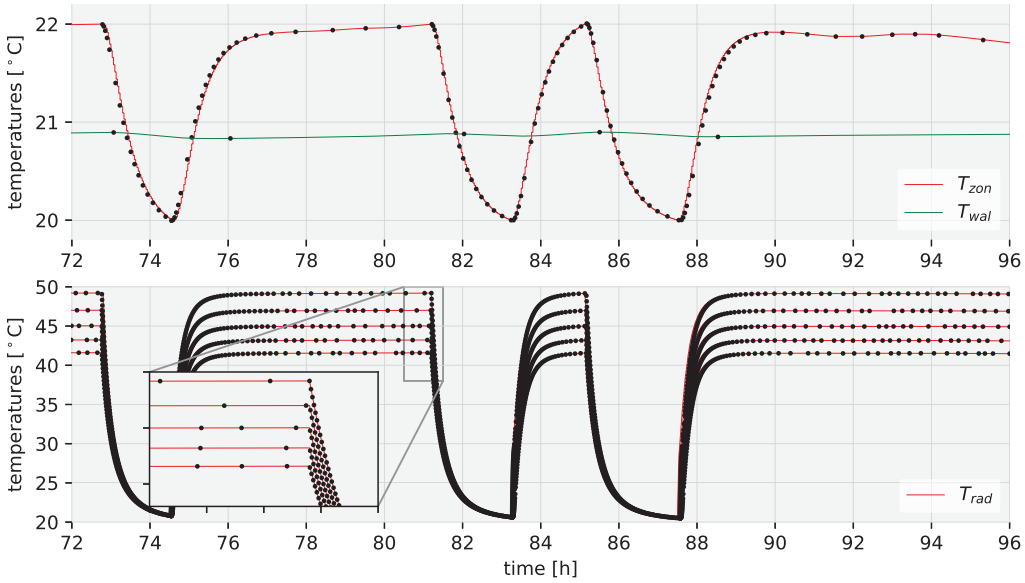

Figure 8: Trajectories computed by CVode (red) and points (black) at which QSS updates the respective state for the model in Figure 7. For easier readability, for QSS only every 5th state update is drawn.

cember. "Modelica Hybrid Modeling and Efficient Simulation." *38th IEEE Conf. on Decision and Control*. IEEE, Phoenix, AZ, 3502–3507.

Tummescheit, Hubertus, Jonas Eborn, and Katrin Prölss. 2005, March. "AirConditioning - a Modelica Library for Dynamic Simulation of AC Systems." Edited by Gerhard Schmitz, *Proc. of the 4th Modelica conference*. Modelica Association and Hamburg University of Technology, Hamburg, Germany, 185–192.

Wetter, Michael, Marcus Fuchs, Pavel Grozman, Lieve Helsen, Filip Jorissen, Moritz Lauster, Dirk Müller, Christoph Nytsch-Geusen, Damien Picard, Per Sahlin, and Matthis Thorade. 2015, December. "IEA EBC Annex 60 Modelica Library – An international collaboration to develop a free open-source model library for buildings and community energy systems." *14-th IBPSA Conf.* 395–402.

Wetter, Michael, Antoine Gautier, Milica Grahovac, and Jianjun Hu. 2019a, September. "Verification of Control Sequences within OpenBuildingControl." *16-th IBPSA Conference*.

Wetter, Michael, Milica Grahovac, and Jianjun Hu. 2018, August. "Control Description Language." *1st American Modelica Conf.* Cambridge, MA, USA.

Wetter, Michael, Jianjun Hu, Milica Grahovac, Brent Eubanks, and Philip Haves. 2018, September. "OpenBuildingControl: Modeling feedback control as a step towards formal design, specification, deployment and verification of building control sequences." *Proc. of Building Performance Modeling Conference and SimBuild*. Chicago, IL, USA, 775–782.

Wetter, Michael, and Christoph van Treeck. 2017, September. *IEA EBC Annex 60: New Generation Computing Tools for Building and Community Energy Systems*.

Wetter, Michael, Christoph van Treeck, Lieve Helsen, Alessandro Maccarini, Dirk Saelens, Darren Robinson, and Gerald Schweiger. 2019b. "IBPSA Project 1: BIM/GIS and Modelica framework for building and community energy system design and operation – ongoing developments, lessons learned and challenges." *IOP Conference Series: Earth and Environmental Science* 323 (sep): 012114.

Wetter, Michael, Wangda Zuo, Thierry S. Nouidui, and Xiufeng Pang. 2014. "Modelica Buildings library." *Journal of Building Performance Simulation* 7 (4): 253–270.

Zeigler, Bernard P., and J. S. Lee. 1998. "Theory of Quantized Systems: Formal Basis for DEVS/HLA Distributed Simulation Environment." *SPIE Conf. on Enabling Technology for Simulation Science*, Volume SPIE Vol. 3369. Orlando, 49–58.

Zuo, Wangda, Michael Wetter, Wei Tian, Dan Li, Mingang Jin, and Qingyan Chen. 2016. "Coupling Indoor Airflow, HVAC, Control and Building Envelope Heat Transfer in the Modelica Buildings Library." *Journal of Building Performance Simulation* 9:366–381.