Co-Simulation and Validation of Advanced Building Controls with VOLTTRON™ and EnergyPlus™

Charles D. Corbin, Draguna Vrabie, Srinivas Katipamula
Pacific Northwest National Laboratory, Richland, WA, USA

Abstract

This paper describes a streamlined building controls development process using the VOLTTRON™ platform. The paper introduces a new software agent that enables co-simulation with EnergyPlus™ and accelerates the development process from concept to commissioning. With this new co-simulation capability in VOLTTRON, building control developers have a method to easily design, test, validate, and deploy building applications on a single platform. The paper describes the implementation of the software agent and concludes with a case-study.

Introduction

New building control concepts typically undergo a staged development process. The stages of the process — based on the US DOE technology readiness level development scale (U.S. Department of Energy, 2011) — can be described as follows:

1. ***Basic technology research***: The control concept is selected based on principled reasoning. The control algorithm is specified using high-level mathematical descriptions and decision flow diagrams, and realized using a scripting language that enables rapid prototyping.
2. ***Feasibility proving and performance validation***: The main functional components and architecture of the control technology are defined. The energy performance associated with a given control strategy is evaluated under diverse operating conditions. This is accomplished by either integrating the control technology prototype with building simulation software or utilizing co-simulation between the building energy modeling software and the control prototyping software.
3. ***Technology development***: Robust development of control code to be integrated and validated with a real building system. This typically includes re-implementation of the prototype code in a development language supported by the physical platform where this code will be executed.
4. ***Technology demonstration***: Validation of technology performance in a pilot demonstration at a building test bed site for a set of relevant operating conditions. This stage includes preliminary specification of the technology installation process.
5. ***System commissioning***: Pre-commercial demonstration of technology in a full range of operating conditions at pilot building sites to validate technology performance and robustness. This stage includes full specification of the technology installation and deployment processes.
6. ***System operation***: Permanent control system operation at building sites in a full range of operating conditions.

The *feasibility proving and performance validation* stage requires utilization of modern whole-building modeling software. Existing building energy simulation tools have enabled the rapid design and validation of control algorithms prior to testing in physical demonstrations. However, these tools do not simulate the communication and computation architecture characteristic to continuous physical deployments, where scalability and security are critical to building operation. Thus, application testing using these tools alone is insufficient to validate the expected performance achievable at a building test site. For this reason, the control algorithms must be re-implemented and the control software must be re-validated in a suitable building control platform during the *technology development* stage, prior to deployment at a real building test site. This additional step is lengthy and prone to implementation error.

In this paper we show how the control development process can be integrated with the VOLTTRON platform to accelerate the transition from application design, testing and validation to physical deployment. This integration eliminates the re-implementation and re-validation steps of the *technology development stage*, thus speeding-up the overall control technology development process from concept to commissioning. We show how this speed-up is enabled by a new software agent built upon the VOLTTRON™ platform that allows co-simulation of VOLTTRON-based control agents with EnergyPlus building models. This capability provides a means to design, test and validate control algorithms in co-simulation, then

directly deploy the control code to building sites in demonstration, commissioning and stand alone operation contexts.

We describe the VOLTTRON agent and the process for configuring and running a co-simulation using VOLTTRON and EnergyPlus. We then demonstrate the agent's use in the development of a transactive control application. The paper concludes with a section describing a physical experiment performed to test the application in a small commercial building.

Prior Work

EnergyPlus (Crawley et al., 2000) is regarded as the standard research tool for advanced building energy modeling and analysis. Early versions provided little ability to modify built-in control algorithms. With the introduction of the Energy Management System (EMS), practitioners gained the ability to write custom control sequences using a native scripting language called the EnergyPlus Runtime Language (ERL) (Ellis et al., 2008).

Soon after the introduction the EMS and ERL, Wetter and Haves (2008) introduced the Building Controls Virtual Test Bed (BCVTB) and the ExternalInterface object, which enabled more advanced control algorithms developed in external software such as Simulink (Mathworks, 2017) or Dymola (Modelon, 2017) to be co-simulated with EnergyPlus. This has allowed researchers to develop and test algorithms not practicable using the EMS. Wetter (2010) also developed the Modelica Buildings Library — a library of building components including equipment and controls — that allows researchers to co-simulate EnergyPlus with dynamic equipment models and advanced controls simulated in Dymola, using the BCVTB to federate communication between the two.

Nouidui et al. (2013) introduced a utility that can export an EnergyPlus model as a functional mock-up unit (FMU) for import into software supporting the functional mock-up interface (FMI) standard (FMI Standard, 2017). Subsequently, Nouidui et al. (2014) implemented a functional mock-up unit (FMU) interface in EnergyPlus that allows it to import and execute component and control models complying with the FMI standard. These two methods allow a variety of options for researchers looking to extend the control capabilities of EnergyPlus in co-simulation.

More recently, Nouidui and Wetter (2014) introduced a new component to the Niagra AX (Tridium, 2017) platform that enables the import of simulation and control models exported as FMUs. Their work provided the missing link in a tool chain that enables integrated control development and validation, with detailed building simulation, and deployment to a commercial building control platform.

VOLTTRON (Haack et al., 2013) is being developed to enable the deployment of secure, scalable applications that allow greater integration of buildings, vehicles and other end-use loads with the electric power system. VOLTTRON is an open-source platform for distributed sensing and control developed by the Pacific Northwest National Laboratory (PNNL) for the U.S. Department of Energy (DOE). The platform provides services for collecting and storing data from buildings and devices and provides an environment for developing applications that interact with that data. VOLTTRON enables the creation of advanced control applications which can monitor and control building operations using industry-standard protocols in a manner similar to existing supervisory building control systems. Herein we show how the VOLTTRON platform enables the control application development process.

Method

Applications built upon the VOLTTRON platform typically consist of two or more software agents that communicate over a common bus via publish-subscribe (PubSub) and remote procedure call (RPC) interfaces. The simplest of applications is comprised of a single agent that executes a control algorithm, and a *Master Driver Agent* that communicates control signals to the building automation system (BAS) over either BACnet (ASHRAE, 2017) or MODBUS (Modbus Organization, Inc., 2017). More sophisticated control applications may consist of many more agents responsible for various aspects of building control. In this way, a distributed, agent-based control system can be implemented in VOLTTRON. The control system presented in this paper is one such example.

Control developers previously had limited ability to validate or test control algorithms in simulation prior to deployment on the VOLTTRON platform. Algorithms developed in a separate environment had to be re-implemented in VOLTTRON before they could be deployed to a physical building system, and validation of the algorithm could only be performed in situ. Often, the algorithm would be modified to allow it to communicate with the BAS interface, i.e. the *Master Driver Agent*, introducing the potential for errors, and exposing the physical system to untested code.

To accelerate the control development process, we developed an *EnergyPlus Agent* that mimics the behavior of a *Master Driver Agent*, thus allowing development and testing of building control applications on a single platform. The use of this agent provides a mechanism for control performance validation using co-simulation with EnergyPlus, accelerating the *Feasibility proving and performance validation*, and eliminates steps in the development process that are prone to error in the *technology development* stage.

Implementation

The *EnergyPlus Agent* manages communication between the VOLTTRON message bus and the EnergyPlus simulation engine. It builds upon previous work by Wetter and Haves (2008), who modified En-

ergyPlus for co-simulation using the BCVTB. The agent initializes the EnergyPlus simulation and performs time step synchronization. Like the BCVTB, the agent instantiates a simple socket server to which EnergyPlus connects. In contrast to the BCVTB, the agent automates the creation of the socket and variable configuration files required by EnergyPlus.

Figure 1a shows the communication architecture of a hypothetical control application running in co-simulation. Multiple EnergyPlus models may be executed in a given co-simulation, each managed by a separate *EnergyPlus Agent* instance. Figure 1b shows the same application running against a physical control system. The communication sequence is not explicitly shown, as it will vary by application.

From the perspective of other agents running on the platform, the *EnergyPlus Agent* appears to be a physical building. The *EnergyPlus Agent* exposes all *Master Driver Agent* RPC methods, and publishes data in a format identical to that published by the *Master Driver Agent*. This allows VOLTTRON applications to be developed and tested first in co-simulation, and then deployed *without modification* to a physical test bed.

Dependencies

The VOLTTRON platform runs on the Linux OS. Development and testing of applications on other operating systems typically occur in a virtual machine (VM). Additionally, the *EnergyPlus Agent* requires the following:

EnergyPlus version 3.0 or higher. In version 3.0, the `ExternalInterface` object was introduced, allowing EnergyPlus to exchange information over a local BSD socket connection.

Java version 1.X or higher. EnergyPlus calls an external BCVTB library to perform validation of socket and variable configuration files written by the agent.

BCVTB Library. This library consists of a Java archive (JAR) and an XML Document Type Definition (DTD). This library is included in the *EnergyPlus Agent* source repository.

Integrated Development and Deployment

Software development has become increasingly integrated with the availability of full-featured software development environments. The Eclipse Integrated Development Environment (IDE) provides many useful tools for application development, including debugger support and launch configuration, among others. The development process for VOLTTRON control applications in Eclipse — from design to co-simulation and physical deployment — is summarized below. The following steps assume that an EnergyPlus model of a test building is available.

1. ***Identify input/output data required by control application.*** This step has far-reaching implications on design and implementation of the control application. Not all points available from a BAS are available in EnergyPlus and vice versa. For example, a zone's cooling set point may be represented in the control application as a single value, but represented in the BAS as a master set point plus offset during occupied hours, and a separate set point during unoccupied hours. A strategy must be devised for either setting all three points in the application, or modifying the BAS logic to accept a single set point. Identifying the requirements early will ease deployment and modifications to the BAS to occur in parallel to application development.
2. ***Prototype control application in VOLTTRON.*** Early stages of control application development are performed in the preferred VOLTTRON IDE, or transitioned from the original development environment. This stage may force the developer to consider the operation of the control application as a distributed, agent-based application, which may impact design and operation due to the asynchronous nature of the platform.
3. ***Perform unit testing of control application.*** Unit testing provides assurance that the control application performs as expected under a given set of conditions, and is a fundamental component of modern software development. Designing and writing tests may preceded agent development in some development practices. The importance of well-designed unit tests can not be overstated.
4. ***Configure EnergyPlus Agent and control application agent(s).*** This step maps EnergyPlus outputs to control application inputs and vice versa via PubSub topics. A design decision must be made at this point regarding the specific topics used by the application. It is recommended that the topics to which the application is subscribed be identical to those that will be provided by the *Master Driver Agent* running on the VOLTTRON platform during deployment. This is a key step, and ensures that messaging in co-simulation match the messaging in the deployed application exactly.
5. ***Modify IDF to communicate with control application.*** The IDF must be modified to allow communication between the *EnergyPlus Agent* and EnergyPlus. This is accomplished by inserting `Output` and `ExternalInterface` objects into the simulation model's IDF file. This step exposes EnergyPlus output values to the control application, and enables EnergyPlus to accept controller output.
6. ***Perform co-simulation studies.*** Through co-simulation, a control application may be tested and validated using a building energy model as 'hardware in the loop'. This step often reveals flaws in the control algorithm, and introduces

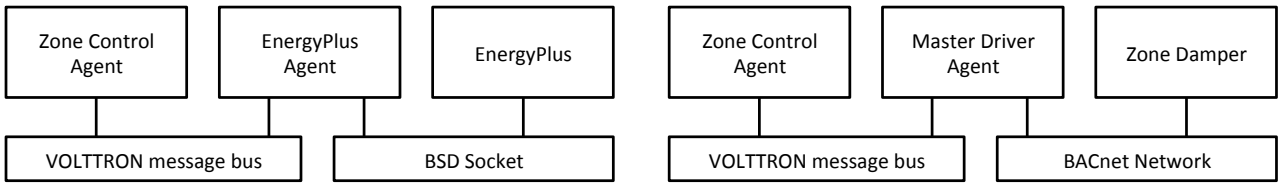

Figure 1: Communication network architectures for control application testing in co-simulation (a), and for the same application deployed to a physical building (b).

edge cases and conditions not previously considered. Co-simulation is a critical step in the application development process which is often overlooked or minimized. By performing co-simulation within the IDE, built-in debugging allows resolution of issues as they are encountered. A screen capture of a co-simulation performed in the Eclipse IDE is shown in Figure 2.

7. ***Analyze and revise application.*** This is typically an iterative process, returning to *Step 2* based on the results produced in co-simulation. Analysis may be performed in real-time with the *Plotter Agent* included in the source repository. Application revisions may be quickly made and tested because development and co-simulation environments are integrated.
8. ***Configure platform agent(s) to publish controller input(s), accept controller output(s).*** In a physical deployment, the *Master Driver Agent* will be responsible for publishing data from the BAS to the the VOLTTRON message bus. This agent should be configured to publish the topics referred to in *Step 4*. Similarly, the *Master Driver Agent* must be configured to control those points output by the control application. These too must match those chosen in *Step 4*.
9. ***Deploy application to physical test bed.*** The control application is deployed to a VOLTTRON instance connected to the physical test bed. This instance typically runs the *Master Driver Agent*. It is extremely important at this stage to ensure that appropriate limits on application output are enforced, either by the application or the BAS. Additionally, one should consider providing a 'kill-switch' to halt the application, or otherwise provide the ability for facility staff to override application commands.
10. ***Perform site testing.*** The deployed control application is tested in the physical test bed, and data recorded for later analysis. Experiments should be continuously monitored to confirm expected behavior. Observations made during tests will likely lead to application refinement, as simulation model behavior and physical test bed behavior can differ due to a number of factors, including: 1) insufficient model complexity, 2) inherent modeling limitations, 3) unexpected data latency, and 4) unanticipated behavior of the BAS in response to application control decisions. Many of these may be resolved by thoroughly researching the building and its control system early in the development process, and by modeling the system at the appropriate level of complexity.

Example Configuration

Agent configuration in VOLTTRON is specified by an external file in JavaScript Object Notation (JSON) format. Figure 3 is an example configuration of the *EnergyPlus Agent* for a simple application. In this example, the configuration tells the *EnergyPlus Agent* which EnergyPlus model to execute, `model`, the weather file to use, `weather`, the location of the BCVTB library, `bcvtb_home`, and its identity on the VOLTTRON platform, `identity`.

The `inputs` and `outputs` blocks define the topics to which the agent will subscribe and publish, respectively. In this example, the *EnergyPlus Agent* will receive a "ZoneTemperatureSetPoint" on the "devices/CAMPUS/BLDG1/ZONE1" topic, and will publish a "ZoneTemperature" to the "devices/CAMPUS/BLDG1/ZONE1" topic.

Because messaging in a multi-agent application is typically asynchronous, the *EnergyPlus Agent* must block the execution of EnergyPlus until all data it requires are received from agents in the application. The `blocking` keyword tells the *EnergyPlus Agent* to wait until the "ZoneTemperatureSetPoint" has been updated before proceeding to the next simulation timestep. Full documentation of configuration options will be made available on the VOLTTRON website (U.S. Department of Energy, 2017).

The EnergyPlus IDF used in co-simulation must also be modified to export variables needed by agents in the VOLTTRON application. Figure 4 shows the corresponding IDF entries required for EnergyPlus to export the "ZoneTemperature" expected by the *EnergyPlus Agent*, and the "ZoneTemperaureSetPoint" provided by the *EnergyPlus Agent* to EnergyPlus.

Application Development Example

The control application is an implementation of transactive control within a small commercial building (Hao et al., 2016). In this application, components

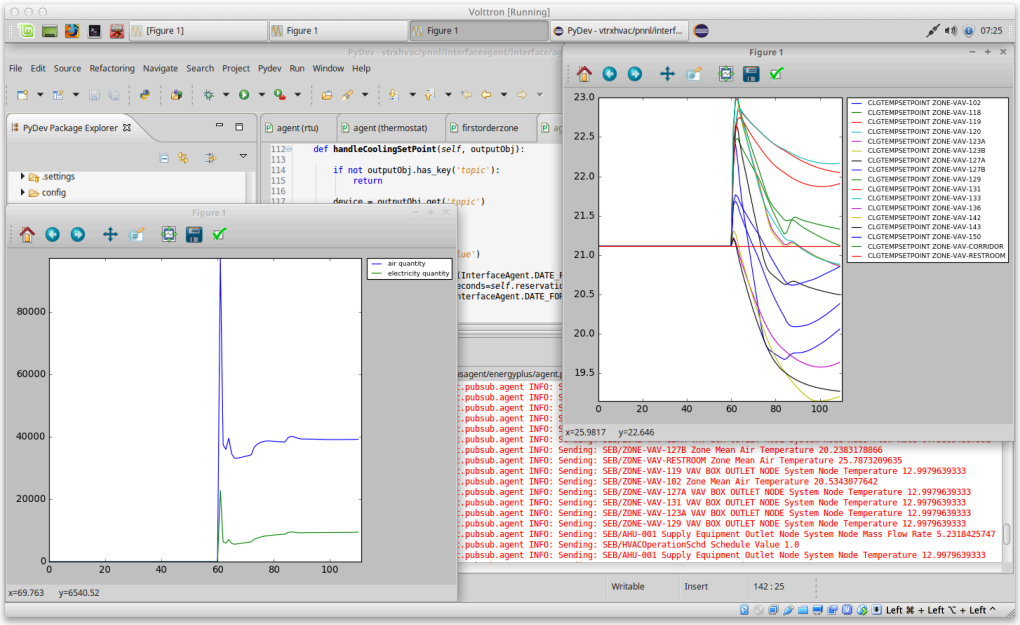

Figure 2: Screen capture of co-simulation within the Eclipse IDE.

```
{
  "properties" : {
    "model" : "~/EPlus/SimpleExample.idf",
    "weather" : "~/Weather/Hanford.epw",
    "bcvtb_home" : "~/bcvtb/",
    "identity" : "platform.actuator"
  },
  "inputs" : {
    "VAV1 Cooling Set Point" : {
      "name" : "ZONE1 CLGSETP",
      "type" : "Schedule",
      "topic" : "devices/CAMPUS/BLDG1/ZONE1",
      "field" : "ZoneTemperatureSetPoint",
      "blocking" : True
    }
  },
  "outputs" : {
    "VAV1 Zone Temperature" : {
      "name" : "Z1",
      "type" : "Zone Mean Air Temperature",
      "topic" : "devices/CAMPUS/BLDG1/ZONE1",
      "field" : "ZoneTemperature",
      "meta" : {"units": "C", "tz": "UTC",
        "type": "float"}
    }
  }
}
```

Figure 3: EnergyPlus Agent configuration for a hypothetical co-simulation.

of a variable air volume (VAV) heating and cooling system are represented by software agents. Agents interact through a set of virtual markets, submitting bids for resources they desire. We consider only cooling operation of the VAV system.

```
ExternalInterface,
  PtolemyServer;

ExternalInterface:Schedule,
  ZONE1 CLGSETP,
  Temperature,
  21.11;

Output:Variable,
  Z1,
  Zone Mean Air Temperature,
  Timestep;
```

Figure 4: EnergyPlus configuration for a hypothetical co-simulation.

VAV Agents estimate the cooling demand of the zone they serve and purchase cool air from the *Air Market Agent*. *VAV Agents* submit their cooling demand into the air market at each market interval, and respond to the cleared air price by resetting their cooling set point within a predefined range. In this example, there are 17 *VAV Agents*.

The *Air Market Agent* has a single supplier, the *AHU-Chiller Agent*. This agent represents the air handler and chiller plant. It estimates the electricity required to power the fan and chiller and submits the resulting electricity demand to the *Electricity Market Agent*. The supplier in the electricity market is the *Electricity Meter*.

The *Air Market Agent* and *Electricity Market Agent* collect bids from buyers and sellers, determining the clearing price and quantity of their commodities, and

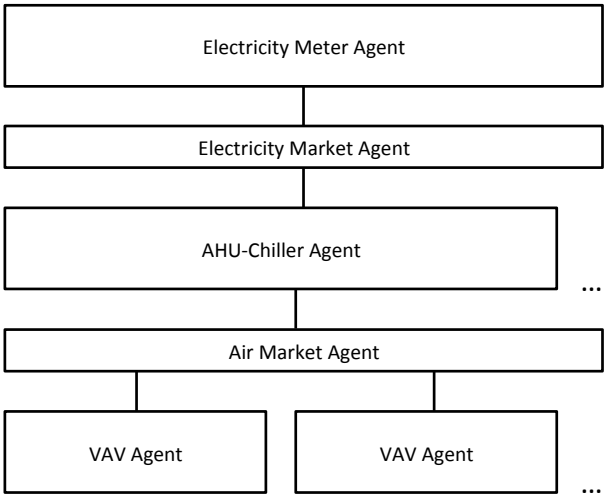

Figure 5: VOLTTRON agents and market structure in the transactive control application.

broadcasting the cleared values to their buyers and sellers. The proposed bi-level market structure is shown in Figure 5.

Building electricity demand is modulated by adjusting the price offered by the *Electricity Meter Agent*. In this example, the *Electricity Meter Agent* supplies a fixed price. This has the effect of constraining the amount of electricity and therefore cool air available, and encourages the individual VAV agents to compete for the resource. The electricity price need not be fixed. In practice, it may be determined by a campus or regional market. In this way, the building can become responsive to system-level constraints, using its inherent demand flexibility to assist with electric power system operations.

Simulation Model

The simulated building is modeled after a modern 24,000ft^2 office building located on a campus. In this study, only a portion of the building that accounts for approximately 50% of the conditioned area, and 90% of occupants is considered. Figure 6 shows the portion of the building modeled.

Heating and cooling are delivered by a variable air volume system (VAV) served by a single built-up air handler having supply and return fans, hot water and chilled water coils, and an economizer. The AHU serves 17 VAV boxes having hot water reheat. A boiler supplies hot water to the VAV boxes and AHU coil. Chilled water is supplied by a central plant modeled by a *DistrictCooling* object.

The simulation model was developed from as-built engineering drawings and equipment specifications. Set points and schedules were obtained from the building manager. Performance data collected by the BAS were used to roughly calibrate the model. Lighting and internal loads were estimated from a site visit.

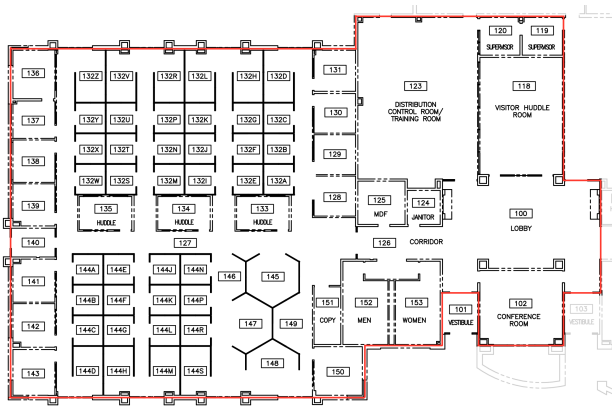

Figure 6: Floorplan of building physical test bed. An EnergyPlus model created from this building is used in co-simulation.

Table 1: Points from building simulation supplied to AHU-Chiller Agent

Point
AHU Discharge Air Temperature
AHU Return Air Temperature
AHU Mixed Air Temperature
AHU Discharge Air Flow
Duct Static Pressure

Agent Configuration

Each agent was configured to receive a set of data points from the building simulation. These points were published by the *EnergyPlus Agent* and used by the agents to estimate the quantities of cool air and electricity needed to condition the building. Points required by the *AHU-Chiller Agent* and *VAV Agents* are summarized in Tables 1 and 2.

Co-simulation Results

The transactive control application and building model were co-simulated for five consecutive weekdays in August. This period was selected to ensure only cooling operation and no instances of reheat were

Table 2: Points from building simulation supplied to VAV Agents

Point
Outdoor Air Temperature
AHU Discharge Air Temperature
VAV Discharge Air Temperature
VAV Discharge Air Flow
Zone Air Temperature
Zone Occupancy
Zone Standby Status
System Status

observed in the baseline simulation. The baseline simulation assumed normal operation, with fixed zone cooling set points of 22.2°C. In the transactive control case, *VAV Agents* were allowed to adjust their cooling set points within a predefined range of 21.1 to 23.9°C, in response to the market-settled price.

Figure 7 shows the resulting HVAC electric demand observed in both simulations. Demand shown is the sum of fan and chiller electric demand, where fan demand is output directly from the building simulation, and chiller demand is estimated from chilled water heat transfer rate using a nominal coefficient of performance (COP)[1].

The co-simulation results show that the transactive control application leads to increased demand in the morning and decrease demand in the afternoon. This has the effect of flattening the demand curve and generally decreasing peak afternoon demand by 4.5% on average. Additionally, a small reduction in total electricity consumption of 3% was measured. The change in building demand results from a depression of zone cooling set points in the morning hours, and a raising of set points in the afternoon. A detailed discussion of the results is presented in Hao et al. (2016).

Physical Experiment

The transactive control application was then tested at the small commercial building test site to demonstrate its deployment and operation. Only the portion of the building corresponding to the areas investigated in co-simulation were controlled.

Agents comprising the transactive control application were deployed to a VOLTTRON instance running on an Intel NUC mini PC. This device was physically located in the building and connected to the subnet of the building's IP network reserved for the BAS. The BAS consists of Alerton field controllers networked through an Alerton Ascent Control Module (Alerton, 2017). The system is managed via a Johnson Controls Metasys Network Automation Engine using the Metasys graphical user interface (Johnson Controls, Inc., 2017).

The *EnergyPlus Agent* was replaced by an instance of a *Master Driver Agent* configured to communicate directly with the VAV controllers over BACnet. The *Master Driver Agent* was configured to publish the data points required by the transactive application agents. Configurations for the *VAV Agents* and *AHU-Chiller Agent* were updated with model coefficients calculated for the physical building[2]. No changes were made to the market structure or agent code.

The experiment was performed on a relatively hot weekday, with outdoor dry bulb temperatures exceeding 40°C by late afternoon. As in the co-simulation example, the *Electricity Meter Agent* provided a fixed price to the building. The experiment ran continuously during normal operating hours from 07:00 to 17:00 on June 29.

Set points and zone temperatures were monitored during the experiment to ensure that the 21.1 to 23.9°C temperature range was not exceeded. Data collection was performed by VOLTTRON using a one-minute sampling interval.

Results and Discussion

Figure 8 shows the HVAC electric demand measured during the experiment. HVAC electric demand is defined in the same manner as above. Chilled water flow rate, cooling coil supply temperature, and cooling coil return temperature were among the data collected during the experiment, allowing estimation of the chilled water heat transfer. Fan power was measured by a power meter installed on the AHU supply fan.

Under the application's control, HVAC power was maintained at a higher level in the morning compared to the previous day, despite nearly identical outdoor temperature profiles. In the afternoon, the relationship reversed, resulting in a flattening of HVAC demand. In general, the transactive control application resulted behavior similar to simulation, despite having somewhat different baseline demand profiles.

Similar to the simulation example, zone temperature set points were depressed in the morning, and raised in the afternoon. This can be seen for a few typical zones in Figure 9.

The peak HVAC demand is defined as the highest value in a 30-minute rolling average of HVAC demand, consistent with the building's peak demand tariff. During the experiment, peak HVAC demand was reduced by 1.8%. Total energy consumption over the day increased by less than 0.1% compared to the simulation.

HVAC demand oscillations were observed on both days, and regularly throughout the summer. These oscillations were a result of static pressure reset logic which was outside of the application's direct control.

Conclusions

This paper outlines the steps of an integrated development and deployment process for control applications using the VOLTTRON platform. The paper introduces a VOLTTRON agent that enables co-simulation of VOLTTRON-based control applications with detailed building energy models in EnergyPlus. The use of the agent is demonstrated in the development of a transactive control application. The *EnergyPlus Agent* and transactive control application presented here are open-source and available for download from the VOLTTRON project repository linked from the VOLTTRON website (U.S. Department of

[1]The building is supplied by a central plant and represents approximately 10% of the plant load. Data measured from the physical plant during summer operation showed a relatively constant COP, and little influence by the building on performance.

[2]The models used by *VAV* and *AHU-Chiller* agents are described fully in Hao et al. (2016).

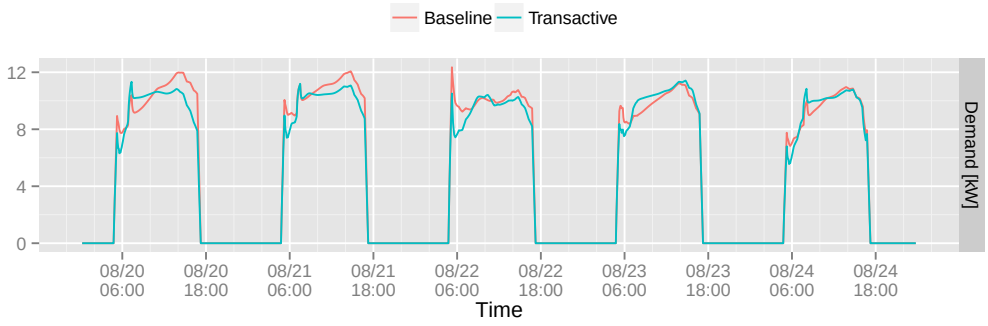

Figure 7: Co-simulation results comparing baseline HVAC electric demand to demand observed during transactive control.

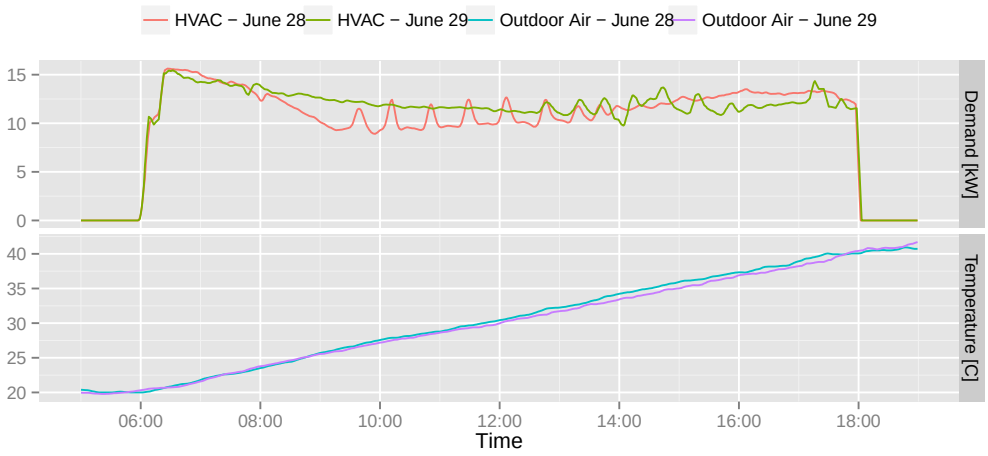

Figure 8: Comparison of HVAC electric demand measured during physical experiment to the previous day's electric demand.

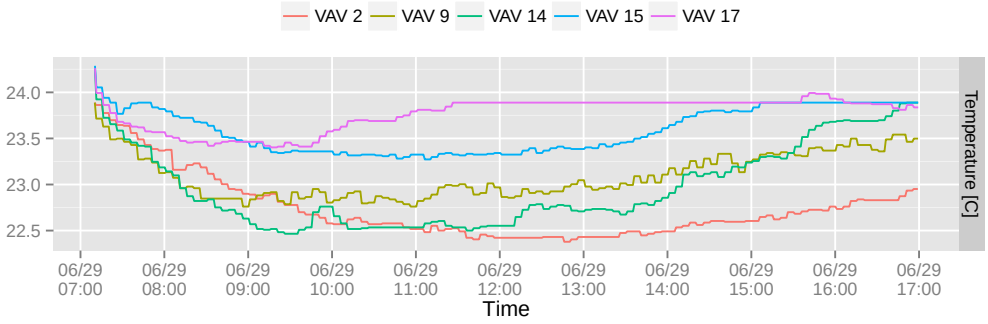

Figure 9: Cooling set points determined by transactive control application during physical experiment.

Energy, 2017).

The control application was demonstrated at a small commercial building test site. In co-simulation, the control application reduces peak demand by a 4.5% on average, and reduces total HVAC electricity consumption by 3% over a five day period. In the single-day physical experiment, the application reduced peak HVAC demand by 1.8%, but resulted in a increase in total HVAC electricity consumption of less than 0.1%. Building behavior and trends observed during the physical experiment were consistent with the co-simulation results.

The limitations in the approach are the result of the inherent limitations in building simulation software, which constrain the type of control application that may be co-simulated, and contribute to uncertainty in the behavior of a physical test bed subjected to said application. The lack of a duct static pressure model in EnergyPlus is such an example. To address limitations imposed by the simulation model presented here, it is recommended that additional integration efforts be undertaken to link VOLTTRON with transient building simulation software such as TRNSYS (TRNSYS, 2017) and Dymola. Simulink has been integrated with VOLTTRON through PNNL's Framework for Network Co-Simulation (Ciraci et al., 2014), however a direct integration of Simulink with VOLTTRON is desirable for some use cases. Additionally, the ability to import FMUs would allow VOLTTRON to leverage a significant body of work across a number of domains..

Acknowledgments

This work was supported by the U.S. Department of Energy's Building Technologies Office through the Buildings-to-Grid Integration Program.

References

Alerton (2017). Alerton Ascent Control Module. `https://alerton.com/en-US/products/BACnet/ACM/Pages/index.asp`.

ASHRAE (2017). BACnet. `http://www.bacnet.org/`.

Ciraci, S., J. Daily, J. Fuller, A. Fisher, L. Marinovici, and K. Agarwal (2014). Fncs: A framework for power system and communication networks co-simulation. In *Proceedings of the Symposium on Theory of Modeling & Simulation-DEVS Integrative*, pp. 36. Society for Computer Simulation International.

Crawley, D. B., L. K. Lawrie, C. O. Pedersen, and F. C. Winkelmann (2000). Energyplus: Energy simulation program. *ASHRAE journal 42*(4), 49–56.

Ellis, P., P. Torcellini, and D. Crawley (2008). Simulation of energy management systems in energyplus. Technical report, National Renewable Energy Laboratory (NREL), Golden, CO.

FMI Standard (2017). FMI Standard. `https://www.fmi-standard.org/`.

Haack, J. N., B. Akyol, S. Katipamula, and R. G. Lutes (2013). Volttron lite: Integration platform for the transactional network. *PNNL-22935, Pacific Northwest National Laboratory, Richland, WA*.

Hao, H., C. D. Corbin, K. Kalsi, and R. G. Pratt (2016). Transactive control of commercial buildings for demand response. *IEEE Transactions on Power Systems PP*(99), 1–1.

Johnson Controls, Inc. (2017). METASYS Building Automation System. `http://www.johnsoncontrols.com/buildings/building-management/building-automation-systems-bas`.

Mathworks (2017). Simulink - Simulation and Model-Based Design. `https://www.mathworks.com/products/simulink/`.

Modbus Organization, Inc. (2017). The Modbus Organization. `http://www.modbus.org/`.

Modelon (2017). Modelon Dymola. `http://www.modelon.com/products/dymola/`.

Nouidui, T., D. Lorenzetti, and M. Wetter (2013). EnergyPlusToFMU. `http://simulationresearch.lbl.gov/fmu/EnergyPlus/export/`.

Nouidui, T., M. Wetter, and W. Zuo (2014). Functional mock-up unit for co-simulation import in energyplus. *Journal of Building Performance Simulation 7*(3), 192–202.

Nouidui, T. S. and M. Wetter (2014). Linking simulation programs, advanced control and fdd algorithms with a building management system based on the functional mock-up interface and the building automation java architecture standards. In *In ASHRAE/IBPSA-USA Building Simulation Conference, Atlanta, GA*.

Tridium (2017). Niagra AX. `https://www.tridium.com/en/products-services/niagaraax`.

TRNSYS (2017). TRNSYS: Transient System Simulation Tool. `http://www.trnsys.com/`.

U.S. Department of Energy (2011). Technology readiness assessment guide. *U.S. Department of Energy, Washington D.C.*.

U.S. Department of Energy (2017). VOLTTRON. `http://energy.gov/eere/buildings/volttron`.

Wetter, M. (2010). Modelica library for building heating, ventilation and air-conditioning systems. *Lawrence Berkeley National Laboratory*.

Wetter, M. and P. Haves (2008). A modular building controls virtual test bed for the integration of heterogeneous systems. In *3rd NAtional Conference of IBPSA-USA, International Building Performance Simulation Association, USA chapter. Bekeley, CA, USA*.