10-2012

MLE+: A Tool for Integrated Design and Deployment of Energy Efficient Building Controls

Willy Bernal
University of Pennsylvania, willyg@seas.upenn.edu

Madhur Behl
University of Pennsylvania, mbehl@seas.upenn.edu

Truong Nghiem
University of Pennsylvania, nghiem@seas.upenn.edu

Rahul Mangharam
University of Pennsylvania, rahulm@seas.upenn.edu

Willy Bernal, Madhur Behl, Truong X. Nghiem, and Rahul Mangharam. (2012). MLE+: A Tool for Integrated Design and Deployment of Energy Efficient Building Controls. *4th ACM Workshop On Embedded Sensing Systems For Energy-Efficiency In Buildings*, (BuildSys '12), Toronto, Canada. Conference site: http://www.buildsys.org/2012/

MLE+: A Tool for Integrated Design and Deployment of Energy Efficient Building Controls

Abstract
We present MLE+, a tool for energy-efficient building automation design, co-simulation and analysis. The tool leverages the high-fidelity building simulation capabilities of EnergyPlus and the scientific computation and design capabilities of Matlab for controller design. MLE+ facilitates integrated building simulation and controller formulation with integrated support for system identification, control design, optimization, simulation analysis and communication between software applications and building equipment. It provides streamlined workflows, a graphical front-end, and debugging support to help control engineers eliminate design and programming errors and take informed decisions early in the design stage, leading to fewer iterations in the building automation development cycle. We show through an example and two case studies how MLE+ can be used for designing energy-efficient control algorithms for both simulated buildings in EnergyPlus and real building equipment via BACnet.

Keywords
Simulation and Modeling, Software engineering, computer applications, tool, energy, buildings, matlab, energy plus

Disciplines
Controls and Control Theory | Power and Energy | Systems Engineering

Comments
Willy Bernal, Madhur Behl, Truong X. Nghiem, and Rahul Mangharam. (2012). MLE+: A Tool for Integrated Design and Deployment of Energy Efficient Building Controls. *4th ACM Workshop On Embedded Sensing Systems For Energy-Efficiency In Buildings*, (BuildSys '12), Toronto, Canada.
Conference site: http://www.buildsys.org/2012/

MLE+: A Tool for Integrated Design and Deployment of Energy Efficient Building Controls*

Willy Bernal, Madhur Behl, Truong X. Nghiem, Rahul Mangharam
Department of Electrical and Systems Engineering
University of Pennsylvania
{willyg,mbehl,nghiem,rahulm}@seas.upenn.edu

Abstract

We present MLE+, a tool for energy-efficient building automation design, co-simulation and analysis. The tool leverages the high-fidelity building simulation capabilities of EnergyPlus and the scientific computation and design capabilities of Matlab for controller design. MLE+ facilitates integrated building simulation and controller formulation with integrated support for system identification, control design, optimization, simulation analysis and communication between software applications and building equipment. It provides streamlined workflows, a graphical front-end, and debugging support to help control engineers eliminate design and programming errors and take informed decisions early in the design stage, leading to fewer iterations in the building automation development cycle. We show through an example and two case studies how MLE+ can be used for designing energy-efficient control algorithms for both simulated buildings in EnergyPlus and real building equipment via BACnet.

Categories and Subject Descriptors

I.6.3 [**Simulation and Modeling**]: Applications; D.2.2 [**Software Engineering**]: Design Tools and Techniques; J.7 [**Computer Applications**]: Computers in Other Systems

General Terms

Design

Keywords

Building simulation, building control, control design, energyplus, energy-efficient building, integrated design, matlab

1 Introduction

In the design of energy-efficient buildings, systems engineers require both, tools capable of simulating high-fidelity plant models, and tools to design, evaluate and deploy modern control methods on those plant models. There is currently a significant gap between the functionality of the tools available for building energy research. While simulation tools can accurately simulate the behavior of a building and its energy consumption, their capabilities for advanced control design and optimization are inadequate. This is, in part, is due to the limited interaction between building modeling and simulation experts and the control systems engineers community. Consequently, as these research communities form tighter collaborations, there is a need for software tools for end-to-end design of energy-efficient building control systems. This will not only influence how architectural design choices directly affect the operational efficiency of building automation systems, but also facilitate the co-design of high-performance control systems for the specific building model. The focus of this effort is on the development of MLE+, a co-simulation tool, that utilizes the simulation capabilities of building energy software tools such as EnergyPlus while taking full advantage of the Matlab environment for control design.

Building simulation tools like EnergyPlus [1], TRNSYS [2], ESP-r [3], eQuest [4], DOE-2 [5] and DesignBuilder [6] offer powerful methods for simulating realistic behavior of buildings and for evaluating their energy efficiency and sustainability. These tools use high fidelity physical models for heat conduction through surfaces, heat and mass transfer, coupling of air and water loops, thermal comfort, fenestrations, daylighting control, weather conditions, atmospheric pollution, occupancy and Heating Ventilation and Air Conditioning (HVAC) equipment. Using increasingly detailed descriptions of a building, one can obtain an estimate of the building's energy requirements in terms of heating and cooling loads, interior environmental conditions and building automation operation cost. Although building simulation tools can run high-fidelity simulations, they have limited capability for algorithm development, optimization, control synthesis and model-based system design.

EnergyPlus [1] is one of the most robust building energy analysis and thermal load simulation tools available today and it has become the de facto whole building simulation tool sup-

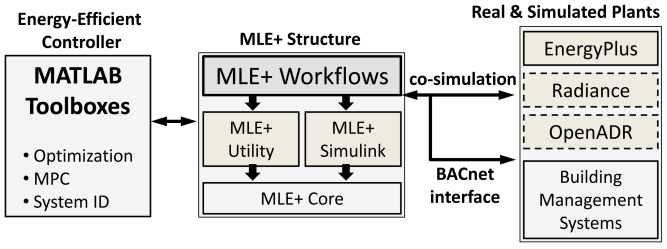


Figure 1. MLE+ interfaces control systems toolboxes with building models and systems

* This material is based upon work supported by the DoE Energy Efficient Buildings HUB sponsored by the Department of Energy under Award Number DE-EE0004261.

Buildsys'12, November 6, 2012, Toronto, ON, Canada.

ported by the U.S. Department of Energy. It is a stand-alone simulation engine which processes text-based input files to run realistic building simulations. It allows simultaneous simulation of loads, systems, and plant and therefore permits quick assessment of building performance. The most recent version also supports coupling with Functional Mock-up Units (FMUs) for co-simulation [7], which allows extending EnergyPlus with custom simulation code in the C language. It provides a built-in energy management system that allows integration of simple rule-based control.

However, EnergyPlus lacks the capability to directly interface with scientific computation and simulation software such as Matlab and Simulink. Therefore, it is difficult to implement and simulate advanced control feedback strategies such as Model Predictive Control (MPC)[8], where an optimal control signal is repeatedly computed based on the current state of the controlled plant. It requires considerable effort and time for a control expert to obtain working knowledge of EnergyPlus, become familiar with its elaborateness and use it for implementing advanced control algorithms. Thus, we identify the following desired capabilities of such a co-design and co-simulation tool:

- Support high fidelity building simulation software and standard scientific computation software.
- Have the capability to compare and rapidly simulate scenarios of different control algorithm implementations across a range of building model parameters. This helps in taking informed decisions during the early phases of design.
- Facilitate common tasks in energy-efficient building design, such as identifying and validating simplified models from high order physical models, optimizing parameters of a building model, designing advanced controllers and their quantative analysis for a building.

In this paper, we present MLE+, an open-source Matlab/Simulink toolbox for building energy research and development (Fig. 1). MLE+ provides the capability to perform co-simulation with EnergyPlus from Matlab. Co-simulation (or co-operative simulation) is a simulation methodology that allows individual components to be simulated by different tools running simultaneously and exchanging information in a collaborative manner. The following are the main features of MLE+:

1. **Simulation configuration**: The MLE+ front-end streamlines the configuration process of linking the building model and the controllers by abstracting the necessary parameters from the co-simulation. This reduces setup time and configuration problems.
2. **Controller design**: MLE+ provides a control development workflow as well as graphical front-ends for designing advanced control strategies, in which the building simulation is carried out by EnergyPlus while the controllers are implemented in Matlab or Simulink.
3. **Simulation-based optimization**: MLE+ can be used to find optimal parameters or control sequences for building system simulations in EnergyPlus.
4. **Data analysis**: After a co-simulation run, using MLE+, the output data from EnergyPlus can be aggregated, analyzed and visualized in Matlab.

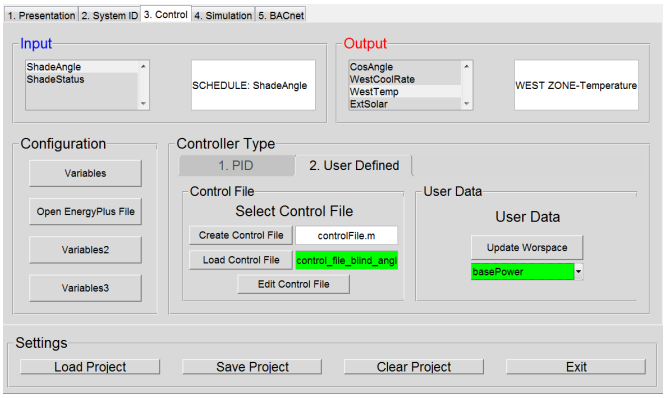


Figure 2. MLE+ tool interface

5. **Building Management System Interface**: MLE+ provides a BACnet interface to develop and implement control methods for real building equipment.
6. **Matlab environment**: MLE+ allows complete access to the Matlab environment and toolboxes such as Global Optimization Toolbox, System Identification Toolbox and Model Predictive Control Toolbox. The user can step through the code for debugging and pause the co-simulation at any time.

The rest of the paper is structured as follows. We first present a running example of using MLE+ for designing a controller for co-simulation with EnergyPlus. Section 3 describes the architecture of MLE+. Two case studies using MLE+ are presented in sections 4 and 5. We conclude the paper with related work in section 6, use cases of MLE+ in section 7 and a discussion in section 8.

2 Using MLE+: A Simple Example

To explain the usefulness of MLE+, we use an example of designing a feedback controller in MLE+ for actuating the window blinds in a building simulated in EnergyPlus.

A single-storied building shown in Fig 3(a) consists of three zones with a total floor area of 130m^2. The West zone of the building consists of a large window equipped with blinds/shades and is subject to strong solar radiation during the day. The goal is to control the window shade deployment of the West zone such that the transmitted solar radiation through the window never exceeds a certain threshold. The window blinds can be controlled using two EnergyPlus variables: (i) `Shading_Deployment_Status` controls whether the blinds are deployed or not; (ii) `ShadeAngle_Schedule` controls the glare inside the zone and should be perpendicular to the incident solar radiation whenever the blinds are deployed. We will design a controller in MLE+ which monitors the angle and intensity of the solar radiation incident on the West zone window. If the incident solar radiation exceeds a certain threshold, the blinds will be deployed and the shade angle will be set to reduce the possibility of glare.

2.1 Environment Configuration

To start a new control development project, the simulation environment needs to be configured. An EnergyPlus Input Data File (IDF) and a weather file are first selected and parsed using the MLE+ front-end. For our example we use the `EMSWindowShadeControl.idf` file, which is available as an example in the EnergyPlus distribution and contains the description for our building.

2.2 I/O Variables Configuration

MLE+ provides a graphical front-end for specifying the input-output variables to be exchanged between EnergyPlus and Matlab for co-simulation. An input variable serves as an input to EnergyPlus at each step of the co-simulation, while output variables are those which can be repeatedly read from EnergyPlus to monitor its internal state. For our example, we specify `Shading_Deployment_Status` and `ShadeAngle_Schedule` as the inputs to EnergyPlus as these are the variables that we will control using MLE+. The controller will need to monitor the incident solar radiation and angle at the West zone, therefore these are specified as output variables. After a description file for a building has been loaded, the MLE+ front-end displays all available input and output variables pertaining to that building.

In MLE+, an alias can be specified for each of the variables (Fig. 3(b)). The alias allows a user to reference a variable with a more intuitive name and avoid the intricate names specified by EnergyPlus. For instance, the EnergyPlus variable `Zn001_Wall001_Win001_Shading_Deployment_Status` can be assigned a more intuitive name as `ShadeStatus`. The alias of a variable can be used later during the controller design and to map to addresses of physical devices in a BACnet network.

2.3 Control Design

In this step, a controller for the window blinds is designed and implemented. The MLE+ front-end for control design generates a Matlab code template for specifying the control algorithm based on the input-output variables defined in the previous step. The I/O variables specified by the user are referred to by their aliases throughout the control file as shown in Fig 3(c). In the code snippet shown in Fig 3(c) the value of the incident solar radiation is compared against the threshold (100 W/m^2) to determine if the shades will be deployed.

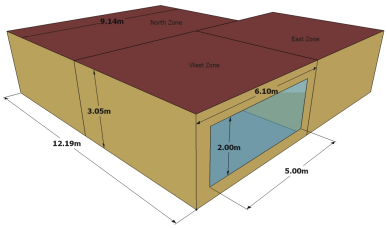
2.4 Simulation and Assessment

Once a preliminary control design has been agreed upon, we can run the simulation or step through it using Matlab debugging environment. MLE+ successfully decouples the simulation engine and the control strategy. This way we can work on tuning the control scheme from Matlab, running multiple simulations without the need of modifying the EnergyPlus file. After the co-simulation finishes, MLE+ extracts and parses all output variables generated by EnergyPlus making them available at the front-end (Fig 3(d)). The output variables can be plotted for a quick view and can be easily saved to the Matlab workspace for further analysis.

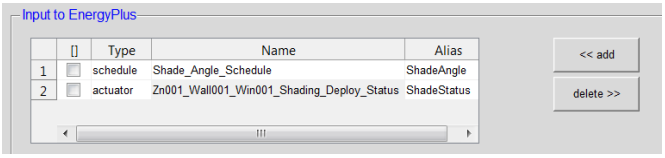
Fig 3(e) shows the results of the window shade controller for our example. Notice how the shades are deployed whenever the incident solar radiation exceeds a certain threshold thereby limiting the transmitted solar radiation. Although the example presented in this section is simple, it demonstrates how MLE+ can expedite the control design by allowing the user to quickly make changes to the controller, then simulate and assess the results.

3 MLE+ Architecture

The structure of MLE+, shown in Fig 1, consists of MLE+ Core components and the MLE+ Workflow. The MLE+ Core components provide interfaces to building simulation tools and building devices. The MLE+ workflow is a sequence of operations which utilize the core components to efficiently design, simulate and evaluate a controller for a given plant model or building automation platform.



(a) EnergyPlus window shading control model

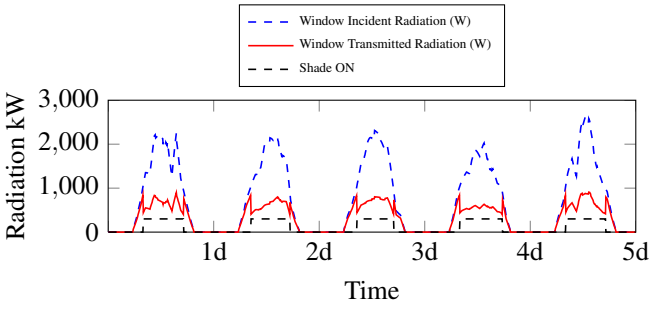


(b) Assigning aliases to variable names

```
1     if Zone_West_Solar > 100
2         % DEPLOYED WHEN SOLAR RADIATION EXCEEDS THRESHOLD
3         ShadeStatus = userdata.Shade_Status_Exterior_Blind_On;
4         ShadeAngle = IncidentAngle;
5     else
6         % SHADES NOT DEPLOYED
7         ShadeStatus = userdata.Shade_Status_Off;
8         ShadeAngle = IncidentAngle;
9     end
10    % FEEDBACK
11    eplus_in_curr.ShadeStatus = ShadeStatus;
12    eplus_in_curr.ShadeAngle = ShadeAngle;
13 end
```

(c) Matlab code snippet of the shading controller (notice alias variables)

(d) viewing plots of EnergyPlus outputs



(e) Window shade controller results

Figure 3. MLE+ workflow example

3.1 MLE+ Core

The MLE+ core handles the interfaces for data-exchange with building simulation tools and communication with building management systems. It uses a Java socket library for co-simulation with EnergyPlus and a BACnet stack library for communicating with BACnet devices. Currently, the interface supports communication with EnergyPlus and BACnet but in

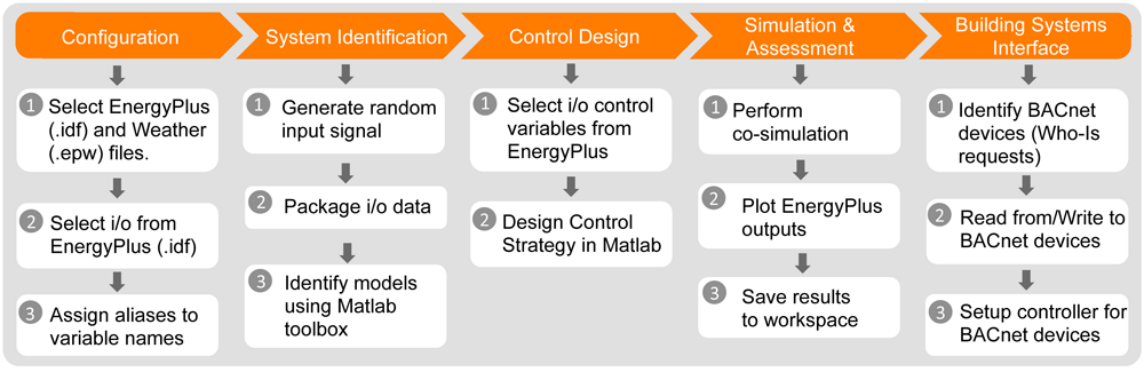


Figure 4. MLE+ control development workflow.

future it will be extended to support other systems as well, for example the Radiance lighting simulation software [9] and OpenADR [10]. The MLE+ Core also provides an Application Programming Interface (API) which contains a set of low-level Matlab functions and classes responsible for all other components the MLE+.

3.2 MLE+ Utilities

MLE+ utilities are built on top of the API to facilitate the development of building energy simulation, control, analysis and optimization. Some examples of MLE+ utilities are:

- A function for parsing and extracting parameters from EnergyPlus building description files.
- An editor that automates the configuration and mapping of external variables for EnergyPlus.
- A simulation result viewer that can load, plot, and export simulation results from EnergyPlus to Matlab.
- A drawing viewer that exports the building geometry from EnergyPlus and display it in Matlab.

3.3 MLE+ Simulink blocks

MLE+ provides a Simulink blockset library for co-simulation between Simulink and other building simulation software. The Simulink blocks are essentially a wrapper of the MLE+ Core for Simulink. This facilitates model based design for building energy control.

3.4 MLE+ Workflow

The core components of MLE+ enable efficient workflow for common tasks, such as developing advanced building controls and optimizing building automation parameters. The *development workflow*, as shown in Fig 4, facilitates a complete design cycle from model identification of an EnergyPlus building model, to designing an energy-efficient controller, to finally deploying the control algorithm in real buildings through BACnet.

4 Case Study 1: Energy-Efficient Controller Design

Radiant heating systems serve as an alternative to the conventional forced-air heating, ventilation and air conditioning (HVAC) systems for thermal conditioning of buildings. A radiant floor heating system works by warming up the floor surface which then slowly radiates heat upward into the living space. In [11] an algorithm for peak power reduction of radiant heating systems is presented and is implemented using EnergyPlus. For this case study, we will use the same example as in [11]. The objective here is not to develop new energy-efficient control algorithms for radiant heating systems, but to show the potential and ease of use of MLE+ in developing and evaluating such algorithms.

Table 1. List of parameters

$R_{z_i,j}$	thermal resistance of the j^{th} layer in zone z_i
$C_{z_i,j}$	thermal capacitance of the j^{th} layer in zone z_i
C_{z_i}	thermal capacity of zone z_i
K_{z_i}	thermal conductance between the zone and outside air
K_{ij}	thermal conductance between the zone i and zone j, $i \neq j$
T_a	outside ambient air temperature
Q_{hg,z_i}	internal heat gain due to occupants *etc*.
Q_{sol,z_i}	heat gain due to solar radiation *etc*.
Q_{rad,z_i}	heat gain from radiant floor system *etc*.

4.1 Simulation set-up

A single floor, L-shaped building divided into three interior zones is used as the plant model. The `RadLoTempElecTermReheat.idf` description file provided with EnergyPlus examples was used for this case study. An electric low temperature radiant system is used for heating the floor of each zone, with power ratings of 12kW, 8kW and 8kW for the North, West and East zones respectively. Temperatures in each zone were required to be kept between $l = 22\,^{\circ}\text{C}$ and $h = 24\,^{\circ}\text{C}$. The ambient air temperature profile was of Chicago, IL, USA. We will compare two different scenarios. In the first one, we used MLE+ to implement a control strategy to switch the electric radiant system in each zone ON or OFF (binary) in an energy efficient manner while ensuring that thermal comfort inside each zone is maintained. The second scenario considers a continuous control where the controlled variable takes any number between 0 and the maximum output power of the radiant system. For this case, we will use MPC to minimize the total energy usage throughout the day.

4.2 Model Identification

The internal thermal model of the EnergyPlus building is not accessible from outside EnergyPlus, therefore the first step was to identify a linear state-space model for the building using the MLE+ system identification workflow.

4.2.1 System Model

The dynamics of each zone is modeled using a RC network "lumped-parameter" model as shown in Fig 5. The list of all parameters in the model is given in Table 1. The law of conservation of energy gives us the following heat balance equation

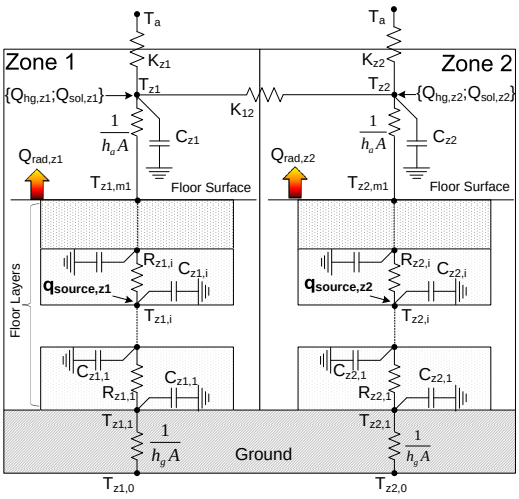


Figure 5. Thermal RC network model for an electric radiant floor heating system

for zone z_i:

$$C_{z_i}\frac{\mathrm{d}T_{z_i}(t)}{\mathrm{d}t} = K_{zi}\left(T_a(t) - T_{z_i}(t)\right) + \sum_{j\neq i} K_{ij}\left(T_{z_j}(t) - T_{z_i}(t)\right) + Q_{hg,z_i}(t) + Q_{sol,z_i}(t) + Q_{rad,z_i}(t) \quad (1)$$

where T_{z_i} is the temperature of zone z_i. The control input to each zone z_i is the state of the electric radiant system. This control input is denoted by $u_i \in [0,1]$ where $u_i = 0$ corresponds to the OFF state with no power consumption and $u_i = 1$ the ON state with maximum power consumption and any intermediate value of u denotes the fraction of maximum power being consumed. Differential equations for all the zones can be combined to give the following state space model for the building:

$$\dot{x}(t) = A_i x(t) + Bu(t) + Dw(t) \quad (2)$$

Where the state *x(t)* consists of node temperatures (3 nodes per zone), control input *u(t)* to the radiant heater, and *w(t)* are disturbances to the system (solar heat gain, occupants etc.).

4.2.2 *System Identification with MLE+*

System identification involves using time-domain and frequency-domain input-output data to identify continuous-time and discrete-time transfer functions, process models, and state-space models. The MLE+ system identification workflow helps in (i) Generating input-output data from EnergyPlus and (ii) Structuring input-output data in a format ready to be used by Matlab's System Identification toolbox.

The MLE+ system identification front-end can generate signals with distinct characteristics for identification purposes. Fig 6(a) shows how the control inputs to EnergyPlus are initialized as random binary signals (rbs) to carry out controlled simulations in EnergyPlus. The MLE+ system identification module streamlines the process of packaging the simulated I/O variables into a Matlab object. This object can be imported into the Matlab's System Identification Toolbox to identify the parameters of the radiant system (Fig 6(b)). This particular example demonstrates how MLE+ allows a seamless integration with Matlab Toolboxes and built-in functions.

4.3 Energy-Efficient Controller Design

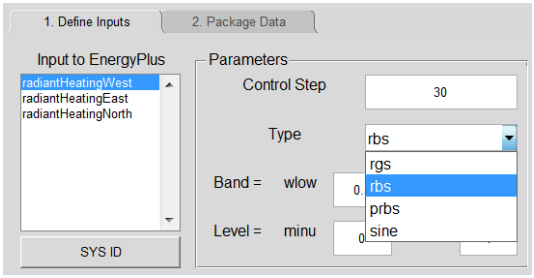
In the ON/OFF case, uncoordinated operation among electric radiant heating systems across multiple zones can cause peaks in the electricity consumption of the building. This occurs when all electric radiant heating systems are simultaneously consuming electricity i.e., the electric radiant system in each zone is ON at the same time. Our goal is to evaluate different control methods to reduce the peak power consumption while ensuring that the temperature inside each zone stays within the comfort range. Through MLE+, one can specify any valid control method for a plant, since it provides a template control file as starting point. We implemented two strategies for this scenario in MLE+:

1. **On-Off uncoordinated control:** To observe uncoordinated control among the zones, we implemented a baseline controller in MLE+ to regulate the temperature of each zone like a two-position thermostat.
2. **Green Scheduling for peak power minimization:** Using the identified state-space model, we implemented the Green Scheduling algorithm for peak power reduction of radiant heating systems in MLE+. Green Scheduling [11], is a simple and lightweight approach for coordinating energy consuming control systems to reduce their peak power consumption.

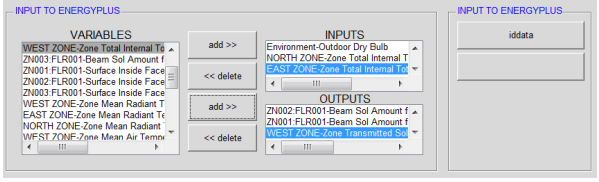
In the second case, we implemented two continuous control schemes to maintain the zone temperatures within the same comfort range.

1. **Proportional control:** This simple control feedback is purely reactive as it only considers the current zone's temperature values. The controlled variable, the electric power of the radiant system, is proportional to the difference of the setpoint and the zone's temperature.
2. **Model Predictive Control:** Using MLE+, we implemented a (relatively sophisticated) model based predictive controller for total power minimization. We used the identified linear state-space model with Matlab's MPC toolbox to compute the optimal power level of the electric radiant heater in each zone. We then fed these power levels as control inputs to EnergyPlus via MLE+ at each simulation step.

By designing multiple controllers in MLE+ for the same plant in EnergyPlus, we can make a fair comparison between the performance of the control algorithms for each scenario.

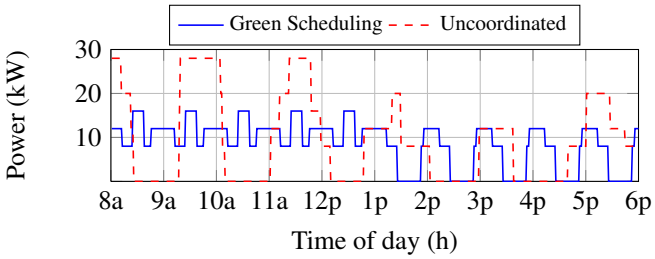


(a) Generating input-output data for system identification

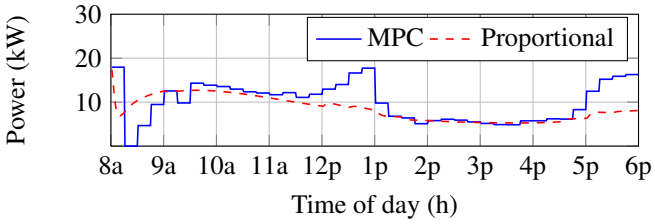


(b) Packaging input-output data System Identification

Figure 6. System Identification using MLE+



(a) Green Scheduling vs. Uncoordinate Power Consumption



(b) MPC vs. Proportional Power Consumption

Figure 7. Case Study 1 Power Profile

The power profile of the four control methods are shown in Fig 7(a) and 7(b) while total power consumption and peak power are presented in Table 2. In both cases, zone temperatures were kept in the desired range between 22 °C and 24 °C. We observed that the curve of electricity demand for the uncoordinated control strategy had several high peaks while it was smoother and flatter for the green scheduling strategy. In total, green scheduling helped save 8% in electricity consumption and reduce peak demand by 42.9%. Also, there is a decrease in the total energy consumption since the periodic schedule tends to operate at a lower mean temperature than the uncoordinated control.

For the continuous control case, the MPC simulation did not achieve a smaller total power during the day. This outcome can be attributed to the fidelity of the model. The MPC implementation requires a more rigorous model and higher accuracy in predicted parameters. These improvements are definitely achievable with MLE+. We did not pursue this task, as our intention was not focused on the development of a novel feedback control, but on the potential of MLE+ to achieve this. In the control design iteration process, MLE+ proves extremely useful as it allows access to the complete debugging environment within Matlab. This translates into considerable time and effort savings.

5 Case Study 2: BACnet interface with test-bed

Controllers implemented in MLE+ can be used for controlling real building devices through BACnet. BACnet [12] was designed to allow communication of building automation and control systems for applications such HVAC, lighting control, access control, fire detection systems and their associated devices. The BACnet protocol defines the format and delivery of messages these devices exchange.

	ON-OFF Control		Continuous Control	
	Uncoordinated	GS (%saved)	Proportional	MPC (%saved)
Consumption (kW h)	93.2	85.7 (8.0%)	84.5	99.7(−2.8%)
Peak demand (kW)	28.0	16.0 (42.9%)	17.4	17.9 (−15.3%)

Table 2. Consumption and peak demand in case study 1

5.1 The BACnet standard

BACnet provides a way of representing any device as long as the device has certain functions specified in the BACnet standard. The standardized BACnet model of a device represents these functions as collections of related information called *objects* each of which has a set of *properties* that further describe it. Each analog input of a device, for instance, is represented by a BACnet `Analog Input` object which has a set of standard properties like `Present Value`, `Sensor Type`, `Location`, `Alarm Limits`, and so on. Some of these properties are required while others are optional. One of the object's most important properties is its identifier, a numerical name that allows BACnet to unambiguously access it.

An example of a BACnet device is a temperature sensor. It is represented as a BACnet `Analog Input` object with a `Present Value` property which is read as the sensor signal.

5.2 MLE+ BACnet interface

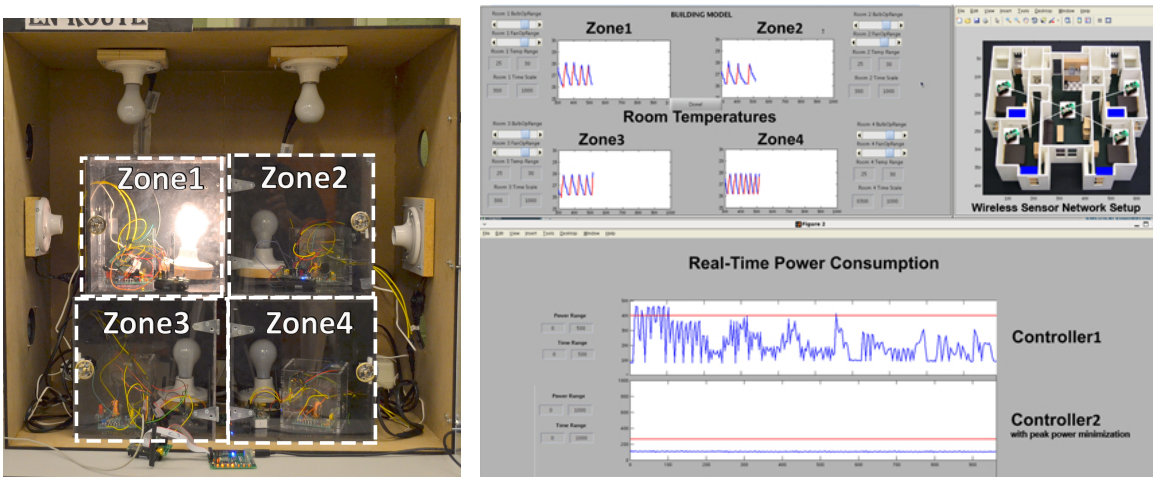
The BACnet interface in MLE+ (Fig5.2) is built upon the BACnet Stack [13] , which is an open source BACnet protocol stack for embedded systems. For the case study, MLE+ is interfaced with a test-bed that simulates both the dynamics of a building and the behavior of BACnet devices.

The test-bed consists of a building with four zones, as shown in Fig 5. Each zone has independent heating and cooling elements which can be controlled to regulate its temperature. Sensor nodes monitor the temperature and energy levels in different zones of the building. An energy dashboard, shown in Fig.8(b), displays the current temperature of each zone and the power consumption of the building. The test-bed is isolated from the surroundings by an outer box, which acts as the outside environment for the building. Fig. 8(b) shows that the temperature dynamics of each zone in the test-bed is similar to the dynamics of a real building.

Each zone of the building test-bed acts as a single BACnet device with two objects: (i) An `Analog Input` object to read the zone temperature, and (ii) An `Analog Output` object to control the status of the heating element in the zone.

BACnet is based on a "client-server" model, therefore BACnet messages are often called service requests. MLE+ behaves as a client and sends service requests to a server, which is usually a BACnet Device. The device/server then performs the service and reports the result to MLE+. A common service request is the `Who-Is` broadcast message, which help the client to identify devices on the BACnet network. MLE+ can generate `Who-Is` service requests to identify the BACnet devices in the building (Fig9(a)). Each device replies back with an `I-am` message containing its device ID. MLE+ can identify the list of all available BACnet devices or devices with IDs that fall in a specific range. MLE+ can also query a specific device if its ID is known. This is useful for checking whether the device is still operational.

Based on the functionality of a device, MLE+ can read and write properties to it. Fig 9(b) depicts how the `Present Value` of the `Analog Output` of zone 1 can be set to 10 by sending a BACnet `WriteProperty` request. The `Analog Output` object controls the heating element of the zone and a value of 10 that the heating element will operate at 10% of its maximum operating power. Similarly, as shown in Fig 9(c), using the `Read Property` request, the temperature value of a zone can be read. The temperature sensor is represented as an `Analog Input` object.



(a) Test bed with 4 zones

(b) Energy dashboard showing zone temperatures and power

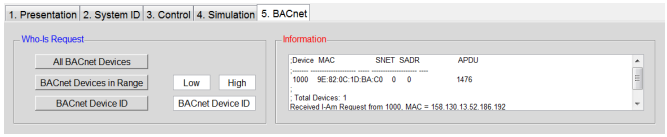
Figure 8. Building test-bed with BACnet connectivity to MLE+

Since MLE+ allows designing controllers in Matlab, a simple feedback controller can be specified for each zone. The controller periodically reads the temperature from each zone and switches the heating elements ON and OFF such that the temperature inside each zone stays within the upper and lower thresholds at all times.

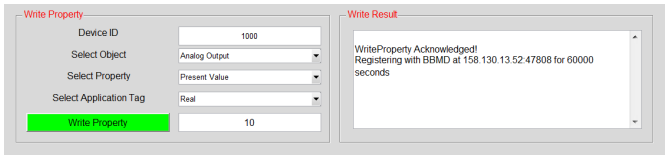
This is a simple example but it paves the way for using MLE+ for modern energy-efficient control methods such as MPC, as described in Section 4). Fig 8(b) shows the performance of two controllers implemented for the test-bed with MLE+. *Controller 1* controls each zone independently of the other zones. This uncoordinated behavior among the zones results in peaks in the total power consumption of the building. *Controller 2* coordinates the zone controls such that the peak power consumption for the building is minimized.

6 Related Work

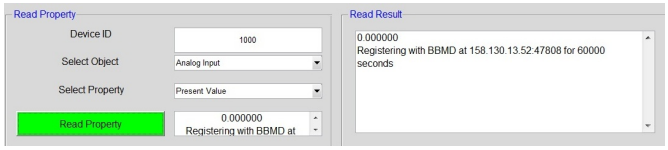
The building automation communities have explored the application of advanced control methods for buildings [14, 15, 16, 17, 18] using building energy simulation tools. In [14], the performance of a real building is analyzed and optimized using EnergyPlus models. In [18], a Siemens Apogee controller is used for demand response control and predictive modeling of a campus building using EnergyPlus. [14, 15, 17] present occupancy driven approaches for energy efficient building control but the proposed control methods cannot be implemented in EnergyPlus since EnergyPlus uses fixed schedules for occupants. In each of these cases, MLE+ can be used for performance evaluation of the proposed control methods with EnergyPlus.



(a) BACnet Who-IS request in MLE+



(b) BACnet Write Property in MLE+



(c) BACnet Read Property in MLE+

Figure 9. MLE+ BACnet Interface

HAMlab [19] provides a collection of tools suitable for simulation with Matlab/Simulink but is mainly suited for modeling the heat and moisture flows in a building. THERMOSYS [20] is another Matlab toolbox for analyzing the behavior of air-conditioning and refrigeration systems. It contains dynamic models of the basic components used in compression cycles but it cannot be used for whole building simulation. TRNSYS can as well interface with Matlab by running it as a process for simulating a design component. However, it does not provide whole-building simulation and does not exploit Matlab's environment. Specifically, it cannot use Matlab's built-in debugging capabilities.

A popular tool for building energy co-simulation is the Building Controls Virtual Test Bed (BCVTB) [21], but it has a few limitations which are discussed next.

6.1 Comparison with BCVTB

BCVTB is a software environment, based on Ptolemy II, for coupling different simulation programs [22, 23]. It can link different simulation programs for co-simulation, including EnergyPlus and Matlab. The co-simulation feature in EnergyPlus was originally developed for BCVTB and can be used by any program to perform co-simulation with EnergyPlus. MLE+ is an example of such a program.

Although Matlab can be coupled with EnergyPlus via BCVTB, its full functionality cannot be used because Matlab is only called by BCVTB as an executable client. Therefore, interactive execution and debugging of Matlab code is not possible. Furthermore, if the Matlab code or the Simulink model has an error, it is much more difficult to find and fix it with BCVTB than with MLE+, which runs in the standard Matlab environment. For users who mostly work with Matlab/Simulink and have never used Ptolemy, learning a new environment as Ptolemy is time-consuming.

These limitations of BCVTB are echoed in the case study presented in [24]. In the study, the authors acknowledged that BCVTB requires considerable know-how and effort in order to set up and operate the co-simulations. They also state how debugging became more difficult for them as adding breakpoints in Matlab code disrupted the co-simulations. They also experienced an increase in the simulation times when using BCVTB.

MLE+ has some distinct advantages over BCVTB:

1. It takes full advantage of the Matlab/Simulink environment, including interactive simulation, code debugging, code generation, and all available toolboxes. In other words, it integrates better with Matlab/Simulink.
2. It is easier to extend MLE+, using Matlab programming, for specialized applications and functions.
3. It is more familiar to users who mainly use Matlab/Simulink, i.e. control engineers.

7 Use cases

MLE+ has been successfully used by both industry and academia ([25, 26, 27, 18]). In [25], Siemens implemented an MPC-based Energy Management Controller (EMC) in Matlab that exchanged data with EnergyPlus via MLE+. The authors clearly assert in their work how they were able to take full advantage of the Matlab toolboxes and MLE+ to achieve this co-simulation. In [18], an intelligent automated demand response building management system using EnergyPlus is presented. According to the report the team explored the utilization of BCVTB but faced simulations problems caused due to communication between EnergyPlus and Matlab. They then switched to MLE+ for developing the demand response strategies. The authors state that MLE+ is a much more reliable way to synchronize the simulation between Matlab and EnergyPlus. Another MPC implementation for EnergyPlus using MLE+ is presented in [27] .

8 Discussion

While building simulation software tools can carry out accurate and realistic building simulations they only provide very basic control methods. On the other hand, control engineers and researchers have explored advanced control strategies for energy-efficient operation of a building, but more often than not, such methods are based on simplified physical models instead. MLE+ is intended to be used as a tool for building energy research and development by researchers who are familiar with Matlab and want to use realistic building simulation capability of building energy simulation software like EnergyPlus.

Being a Matlab toolbox, MLE+ gives the user complete access to all the toolboxes in Matlab and its computational power. The ability to pause the co-simulation with EnergyPlus and step through the code for debugging saves precious development and testing time. This helps control engineers to recognize problems, eliminate errors and take informed decisions early on in the design stage leading to fewer iterations in the development cycle. This feature is not available in tools like BCVTB. Workflows in MLE+ make it very easy to manage tasks like model identification, control design and optimization, post simulation analysis and integration between software applications and building equipment. MLE+ can also be used for tuning building parameters for both building simulation models inside EnergyPlus and for real building equipment.

Our current effort involves extending the tool to work with other building energy simulation tools like Radiance and OpenADR and optimization and modeling tools such as BLOM [28]. Our future work includes using MLE+ for synthesizing a control strategy to control the indoor illumination levels and HVAC system based on predicted and real time solar incident radiation measurements in real buildings.

9 References

[1] D.B. Crawley, L.K. Lawrie, C.O. Pedersen, and F.C. Winkelmann. Energy plus: energy simulation program. *ASHRAE journal*, 42(4), 2000.

[2] SA Klein and University of Wisconsin-Madison Solar Energy Laboratory. *TRNSYS: A transient simulation program*. Eng. Experiment Station, 1976.

[3] P. Strachan. Esp-r: Summary of validation studies. *Energy Systems Research Unit, University of Strathclyde, Scotland, UK*, 2000.

[4] equest: the quick energy simulation tool. `http://doe2.com/equest/`.

[5] F. C. Winkelmann et. al. DOE-2 Supplement: Version 2.1 e. Technical report, Lawrence Berkeley Lab., CA (United States); Hirsch (James J.) and Associates, Camarillo, CA (United States), 1993.

[6] A. Tindale. Designbuilder software. *Stroud, Gloucestershire, DesignBuilder Software Ltd*, 2005.

[7] *EnergyPlus External Interface(s) Application Guide*, May 2012.

[8] E.F. Camacho and C. Bordons. *Model Predictive Control*. Springer, 2004.

[9] G.J. Ward. The RADIANCE lighting simulation and rendering system. In *Conf. Computer graphics & interactive techniques*. ACM, 1994.

[10] M.A. Piette, G. Ghatikar, S. Kiliccote, E. Koch, D. Hennage, P. Palensky, and C. McParland. Open automated demand response communications specification (version 1.0). Technical report, LBNL, 2009.

[11] Truong X. Nghiem, M. Behl, Rahul Mangharam, and George J. Pappas. Green scheduling for radiant systems in buildings. In *Conference on Decision and Control*, Hawaii, USA, Dec. 2012.

[12] ASHRAE. Standard 135-1995: BACnet, Data Communication Protocol for Building Automation and Control Networks. *ASHRAE*, 1995.

[13] Steve Karg. Bacnet stack: An open source bacnet protocol stack for embedded systems. `http://bacnet.sourceforge.net/`.

[14] S. Narayanan, J. S. Apte, P. Haves, M. D. Sohn, and J. Elliott. Systems approach to energy efficient building operation. *ACEEE Summer Study on Energy Efficiency in Buildings*, 2010.

[15] Y. Agarwal et. al. Occupancy-driven energy management for smart building automation. In *BuildSys*, 2010.

[16] V. L. Erickson and A. E. Cerpa. Occupancy based demand response hvac control strategy. In *BuildSys*, pages 7–12, 2010.

[17] C. Liao, Y. Lin, and P. Barooah. Agent-based and graphical modelling of building occupancy. *Journal of Building Performance Simulation*, 2012.

[18] D. Culler, P.K. Wright, Y. Lu, and M Piette. A distributed intelligent automated demand response building management system. Technical report, University of California, Berkeley, 2011.

[19] AWM van Schijndel and JLM Hensen. Integrated heat, air and moisture modeling toolkit in matlab. In *Proc. of 9th International IBPSA Conference*, 2005.

[20] B.P. Rasmussen. Thermosys toolbox user's manual, 2002.

[21] Michael Wetter. Co-simulation of building energy and control systems with the building controls virtual test bed. *Journal of Building Performance Simulation*, 4(3):185–203, 2011.

[22] M. Wetter and P. Haves. A modular building controls virtual test bed for the integrations of heterogeneous systems. *SimBuild*, 2008.

[23] Xiufeng Pang, Michael Wetter, Prajesh Bhattacharya, and Philip Haves. A framework for simulation-based real-time whole building performance assessment. *Building and Environment*, 54(0):100 – 108, 2012.

[24] C. Sagerschnig, D. Gyalistras, A. Seerig, S. Prívara, J. Cigler, and Z. Vana. Co-simulation for building controller development: The case study of a modern office building. In *Proceedings of CISBAT*, 2011.

[25] K. Ji et. al. Prognostics enabled resilient control for model-based building automation systems. In *12th Building Simulation Conference*, 2011.

[26] I. Leobner, K. Ponweiser, G. Neugschwandtner, and W. Kastner. Energy efficient production - a holistic modeling approach. In *Sustainable Technologies (WCST), 2011 World Congress on*, pages 62 –67, nov. 2011.

[27] F. Sakellariou. Model predictive control for thermally activated building systems. *Masters Thesis*, 2011.

[28] BLOM: The Berkeley Library for Optimization Modeling and Nonlinear Model Predictive Control. `http://www.mpc.berkeley.edu/people/sergey-vichik/file/BLOM_paper.pdf`.