## Title

Building energy simulation in real time through an open standard interface

## Permalink

https://escholarship.org/uc/item/7zn730nb

## Authors

Pang, Xiufeng
Nouidui, Thierry S
Wetter, Michael
et al.

## Publication Date

2016-04-01

## DOI

10.1016/j.enbuild.2015.10.025

For official version, see http://doi.org/10.1016/j.enbuild.2015.10.025

# Building Energy Simulation in Real Time through an Open Standard Interface

Xiufeng Pang, Thierry S. Nouidui, Michael Wetter, Daniel Fuller, Anna Liao, Philip Haves

Lawrence Berkeley National Laboratory, Berkeley, CA

## Abstract

Building Energy Models (BEMs) are typically used for design and code compliance for new buildings and in the renovation of existing buildings to predict energy use. The increasing adoption of BEM as standard practice in the building industry presents an opportunity to extend the use of BEMs into construction, commissioning and operation. In 2009, the authors developed a real-time simulation framework to execute an EnergyPlus model in real time to improve building operation. This paper reports an enhancement of that real-time energy simulation framework. The previous version only works with software tools that implement the custom co-simulation interface of the Building Controls Virtual Test Bed (BCVTB), such as EnergyPlus, Dymola and TRNSYS. The new version uses an open standard interface, the Functional Mockup Interface (FMI), to provide a generic interface to any application that supports the FMI protocol. In addition, the new version utilizes the Simple Measurement and Actuation Profile (sMAP) tool as the data acquisition system to acquire, store and present data. This paper introduces the updated architecture of the real-time simulation framework using FMI and presents proof-of-concept demonstration results which validate the new framework.

## Introduction

Energy modeling is commonly used to help engineers, architects and owners design and integrate different building elements, such as the envelope, mechanical and electrical systems. As the standard for testing simulation tools [1] and the methods used to develop an energy model [2] evolve, a carefully created model can predict actual performance more accurately, which makes it possible to utilize energy simulation throughout the life cycle of a building. Pang et al. [3] reported the development of a framework that allows a comparison of building actual performance and expected performance in real time, as shown in Figure 1. The realization of the framework utilized EnergyPlus [4], the Building Controls Virtual Test Bed (BCVTB) [5, 6], and

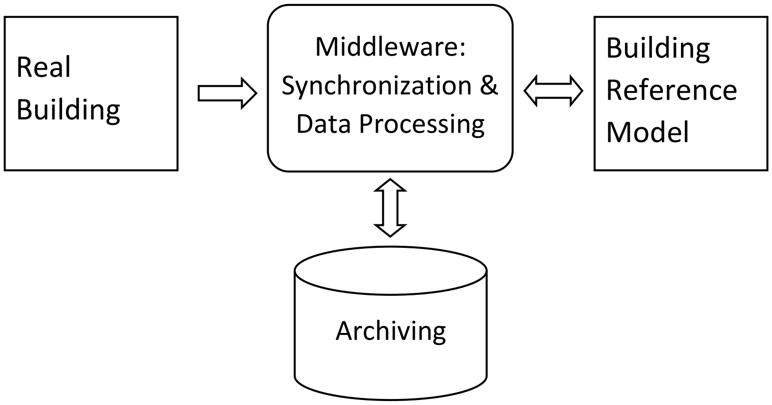


*Figure 1 Illustration of the framework*

a Building Automation System (BAS). EnergyPlus is used to model and simulate the building, the BCVTB acts as the middleware, and the BAS serves as the data acquisition system to collect the building and environment measurements. By real-time simulation, as opposed to off-line simulation, we mean the synchronization of the simulation time with real-time, as represented by the computer clock [3].

The BCVTB is a freely available, open-source, software environment that allows users to couple different simulation programs for co-simulation. The BCVTB also allows simulation programs to be coupled to actual hardware. The BCVTB is a special configuration of Ptolemy II, an open-source framework for actor-oriented design, with the addition of specific actors relevant for the building industry [5, 6, 7]. Ptolemy II provides the synchronization of simulation time to real-time.

EnergyPlus is a detailed whole building energy simulation program that calculates the building heating and cooling loads and outputs disaggregated energy end-use as well as many other simulation variables that are necessary for a detailed comparison of the simulated and actual building performance[8].

A few applications that use the previously published work have been reported [9, 10]. However, there are two main limitations associated with the previous realization of the real-time simulation framework:

- It only works with the models generated from the supported simulation tools, such as EnergyPlus, Dymola and TRNSYS. The running of the models requires the installation of the simulation software, or a runtime license, in the computer that runs the real-time simulation platform.
- It can only communicate with hardware through BACnet [11] or an analog Input/Output (I/O) device [12].

This paper introduces a major new realization of the real-time simulation framework that addresses these and other limitations.

## The New Platform

The BCVTB has been further developed to support the import of simulation models or tools that implement versions 1.0 and 2.0 of the Functional Mockup Interface (FMI), which provides a standardized method of encapsulating models and simulation tools (see below). The resulting encapsulated models and tools are referred to as Functional Mockup Units (FMU). This extension allows the BCVTB to be used to couple simulation models and tools that implement the FMI interface with simulation models and tools that do not implement it. This new capability has enabled the development of the new real-time simulation platform. Figure 2 shows the evolution of the realization of the real-time simulation framework. As can be seen in Figure 2, the new platform uses the Simple Measurement and Actuation Profile (sMAP) and the FMI to address the two main limitations described above.

The BCVTB uses its Synchronous Data Flow (SDF) director to control the communication between different actors. Actors are BCVTB software components that execute concurrently and share data with each other by sending messages via ports [13]. With respect to FMI, the SDF director is the master algorithm that coordinates the data-exchange between FMU actors and other actors. In the new platform, the data-exchange occurs at a fixed time step, which can be specified by the user. The SDF director is responsible for advancing and synchronizing its local time to the computer clock time for real-time applications. If variable time-stepping is required, Ptolemy II supports co-simulation with adaptive time steps, for example by using the Discrete Event director. However, EnergyPlus only supports fixed time step co-simulation, which is why we used the SDF director. The minimum time step which can be handled by the new platform is dictated by the minimum time step supported by the simulation tool that generates the FMU.

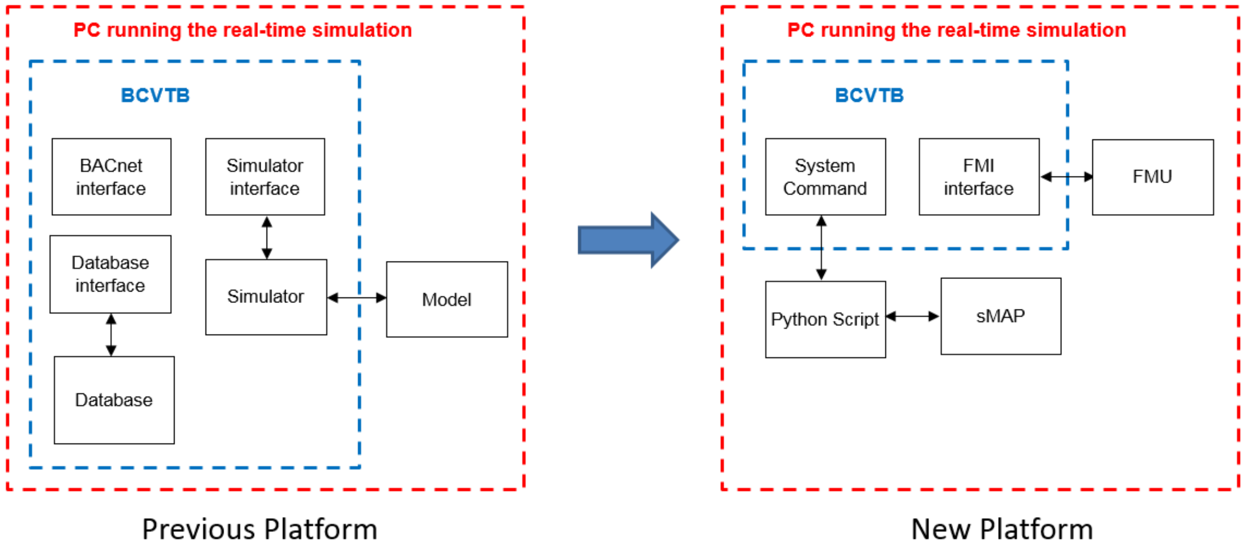


*Figure 2 Evolution of the real-time simulation platform*

### *sMAP*

sMAP is a protocol for publishing time-series data from a wide variety of sensors. The open-source, Python-based, sMAP library developed at the University of California, Berkeley includes software modules to interface to various meters and sensors, set up a backend database for data archiving and provide a front-end visualization plotter to display real-time and archived data. As shown in Figure 3, the sMAP architecture consists of several components that may be used together to capture, transmit, store, and present time-series data [14]:

- The sMAP Sources communicate with existing instrumentation, and collect time-series data using a wide variety of underlying protocols.
- The sMAP Archiver is a high-performance database for storing large volumes of data. It provides a simple and powerful interface for making use of historical data and real-time data and it locates and cleans the raw data.

- The Applications make use of historical and/or real-time data in order to achieve user objectives, for instance, providing attractive visualizations and computing optimal control strategies. For plotting data streams from the sMAP Archiver, sMAP uses the powerdb2 web-browser-based visualizer.

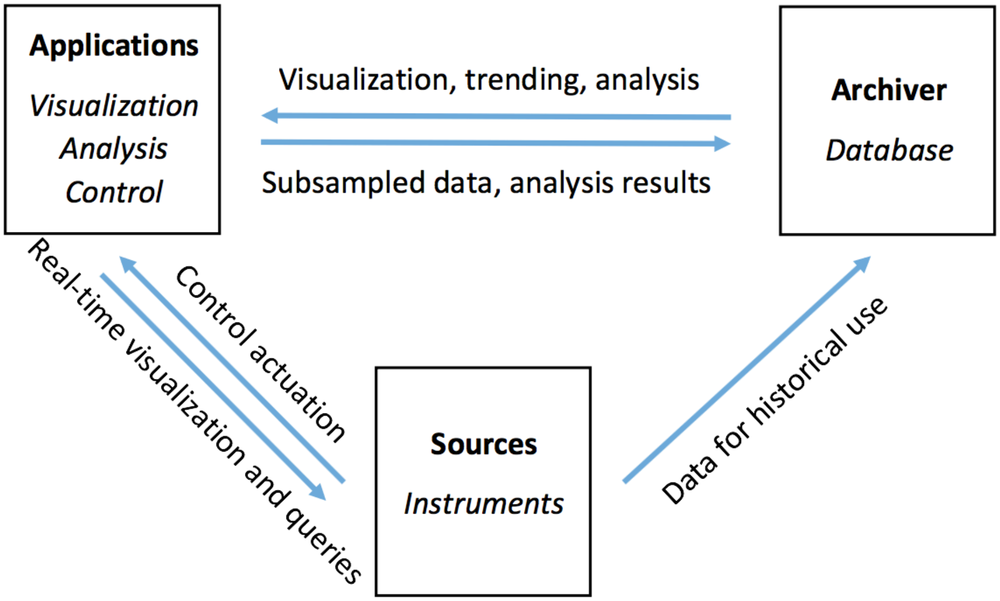


*Figure 3 The sMAP architecture*

Figure 4 shows how sMAP sources work. Each source contains one or more drivers, which contain code to communicate with the instruments. A list of available drivers can be found in [14], including Modbus, BACnet and HTTP/XML. Another feature of sMAP sources is that it is easy to write new drivers for any unsupported instruments. Description of other components in Figure 4 can be found in [14].

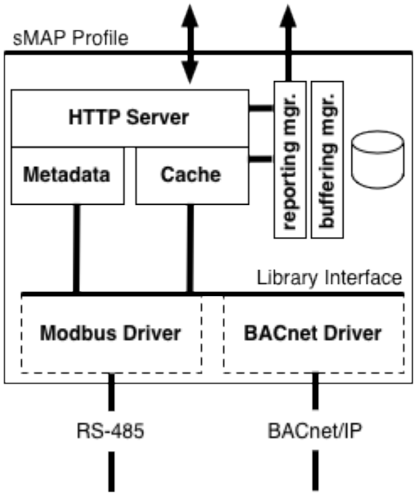


*Figure 4 Diagram of the sMAP Source*

The new real-time simulation platform invokes sMAP through the system command in the BCVTB using python scripts, which allows communication with a wide variety of instruments. In this way, the new real-time simulation platform successfully addresses the limitation regarding hardware communication noted above.

### *Functional Mockup Interface*

The Functional Mock-up Interface (FMI) standard is the result of the Information Technology for European Advancement (ITEA2) project MODELISAR [15, 16, 17]. The FMI standard is a platform-independent protocol to support the exchange of simulation models and simulation tools using a combination of an XML-file and C-code in source or binary form. The first version of the FMI standard – version 1.0., which is used in the tools reported here, was released in 2010. FMI 1.0 consists of three main parts:

1) the **FMI for model-exchange Application Programming Interfaces** (API), which provides a standardized API for dynamic simulation models that need to be imported into another modeling and simulation environment [15]. The dynamic models are described by differential, algebraic, and discrete equations with time- and state-events. Simulation models that implement the FMI for model-exchange use the solvers of the environment into which they are imported to integrate the differential equation in time.
2) the **FMI for co-simulation API** which provides a standard interface for coupling simulation models that contain their own solvers and simulation tools [16]. The FMI for co-simulation comes in two flavors: *Standalone* and *Tool Coupling*, as shown in Figures 5 and 6 respectively.
   a. In the *Standalone* API, the FMU contains the simulation model of a sub-system with the run-time environment provided by the simulation tool that has been used to generate the FMU.

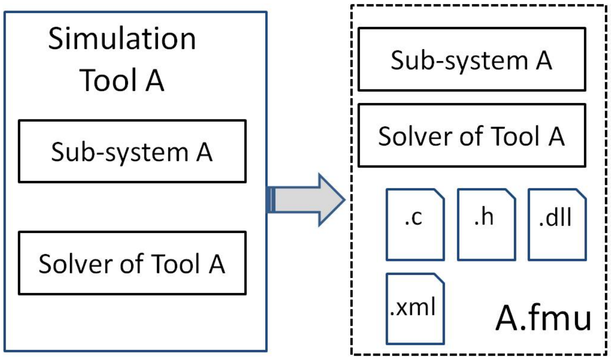


*Figure 5 FMU - the Standalone API*

   b. In the *Tool Coupling* API, the FMU implementation wraps the FMI functions calls to API which are provided by the simulation tool. In this configuration, the simulation tool needs to be present on, or accessible over the network from, the machine where the FMU will be executed.

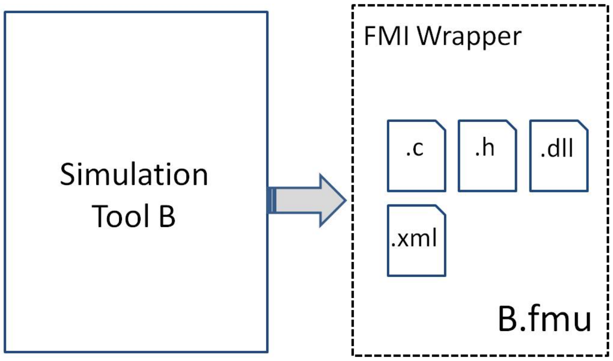


*Figure 6 FMU - the tool coupling API*

3) the **FMI for product life cycle management** which provides a generic way to handle FMI-related data needed in a simulation of systems in a product life cycle management [15].

The second and current version of FMI is FMI 2.0. This version was released in 2014 and is a major enhancement compared to FMI 1.0. FMI 2.0, the Model Exchange and Co-Simulation standards have been merged, and many improvements have been incorporated [18].

The FMI standard defines a set of C-functions to be implemented by a simulation model or a tool to facilitate its interoperability with other tools. For the co-simulation API, the FMI functions are commonly used to a) instantiate the model, initialize it, set the inputs, do time integration, retrieve outputs, terminate and free the model. A simulation model or tool which implements the FMI standard is called a Functional Mock-up Unit (FMU). An FMU is a zip file which contains:

- the FMU model description file. This is an XML-file which contains relevant model information such as the names of the input and the outputs of the model so that an import tool can interface with the model,
- C-sources of the model or compiled code which contain the model for the model-exchange API and the model with its run-time environment for the co-simulation API,
- additional model data such as tables, figures and documentation.

At the time of writing, there are more than 70 simulation programs that are capable of importing or exporting simulation models that implement the FMI standard.

The use of the FMI successfully addresses the limitation of the previous platform that only a limited set of simulation software tools were supported. In addition, for a FMU generated from the Standalone API, the new real-time simulation platform does not require the simulation software to be installed on the same computer.

## Field validation and result discussion

The objectives of the validation were to verify that 1) running an FMU encapsulating EnergyPlus in the BCVTB produces the same simulation results as running the EnergyPlus model directly; 2) the FMU can take the actual weather data as input and run in real time and produce the expected results.

### *Demonstration site information*

We chose the top floor of a 4-story building to conduct the field validation. The building is an office building located in Berkeley, California. The building was originally built with 3 stories and the 4th floor was added at a later time. As a result, the 4th floor is served by a different HVAC system than the rest of the building and so is essentially independent of the rest of the building. Figure 7 shows the floor plan and the HVAC ductwork layout. Two packaged units deliver conditioned air to the perimeter offices and share a common return air duct in the core area.

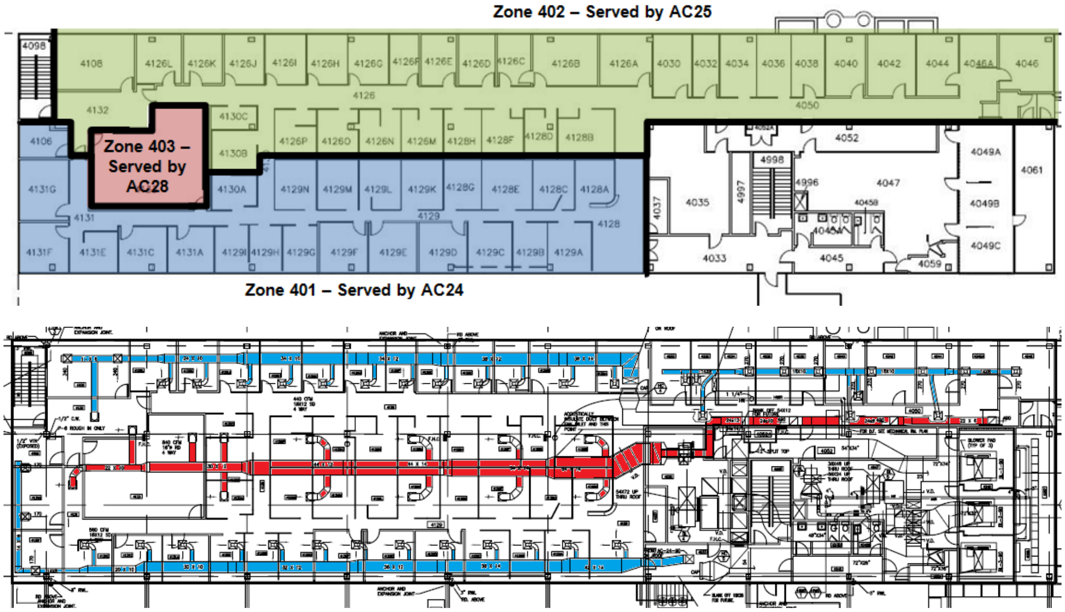


*Figure 7 Demonstration site floor plan and HVAC ductwork layout*

The 4th floor is divided into three conditioned thermal zones and the rest of the area is an unconditioned mechanical room. Zones 401 and 402 consist of private offices and zone 403 is a conference room. Zones 401 and 402 are served by dedicated single zone packaged heat pump units and Zone 403 is served by a split system. Since the usage of the conference room is hard to predict, we only used Zone 401 and Zone 402 for our validation. The two zones are occupied from 6:15am to 6:00pm on week days. The zone occupied cooling set-point is 75°F and the heating set-point is 71°F. The fan runs continuously during the occupied hours regardless of the operating mode of the compressor. The compressor on each unit has two stages of cooling/heating. During occupied hours, the compressor cycles the stages to maintain the zone temperature at the set-point.

## *Real-time simulation setup*

### Instrumentation

The minimum instrumentation needed for real-time simulation is for the weather data, i.e. the outdoor dry bulb temperature, outdoor relative humidity or dew point, diffuse solar radiation, direct or global horizontal solar radiation, wind speed and wind direction, which provide the actual weather boundary conditions that drive the simulation in real time. Depending on the actual use case, additional instrumentation may include the real-time plug load power, lighting power, occupancy and/or room temperature set-points as inputs and the whole building electricity and gas consumption and power consumption for HVAC equipment, for comparison with the simulation outputs.

The demonstration building has a complete set of high quality sensors for weather measurements and the power consumption of each heat pump unit is also monitored. All these measurements are acquired through the sMAP Source and stored in a sMAP Archiver dedicated to this demonstration.

### FMU generation

We chose EnergyPlus as the simulation tool for this demonstration. The geometry model is shown in Figure 8. Since only the fourth floor is modeled, the first three floors are not shown. The grey object behind the building represents a hill. The building model was created using the design documentation, the building envelope material properties, the HVAC equipment specifications and the control sequences, which represents the design intent. A recently-developed feature allows the EnergyPlus model to be exported as an FMU for co-simulation [19, 20]. A software module called EnergyPlusToFMU reads an EnergyPlus Input Data File (IDF), an Input Dictionary file (IDD) and a weather file, and creates an FMU that contains a zip-file as well as shared libraries that are used to interface with EnergyPlus. However, due to the complexity of EnergyPlus, the implementation uses the co-simulation tool coupling API, which means that to simulate the FMU, EnergyPlus has to be installed in the same computer. Since EnergyPlus is a free and open source software, the tool coupling API implementation is not a significant concern.

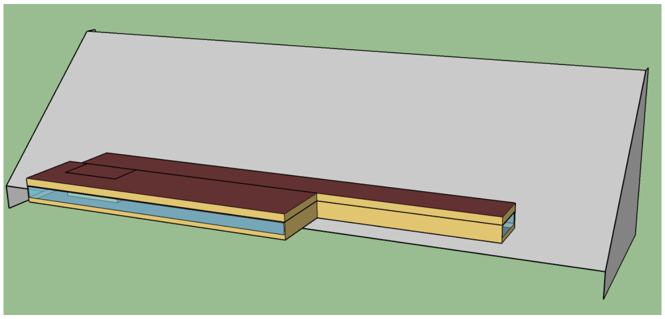
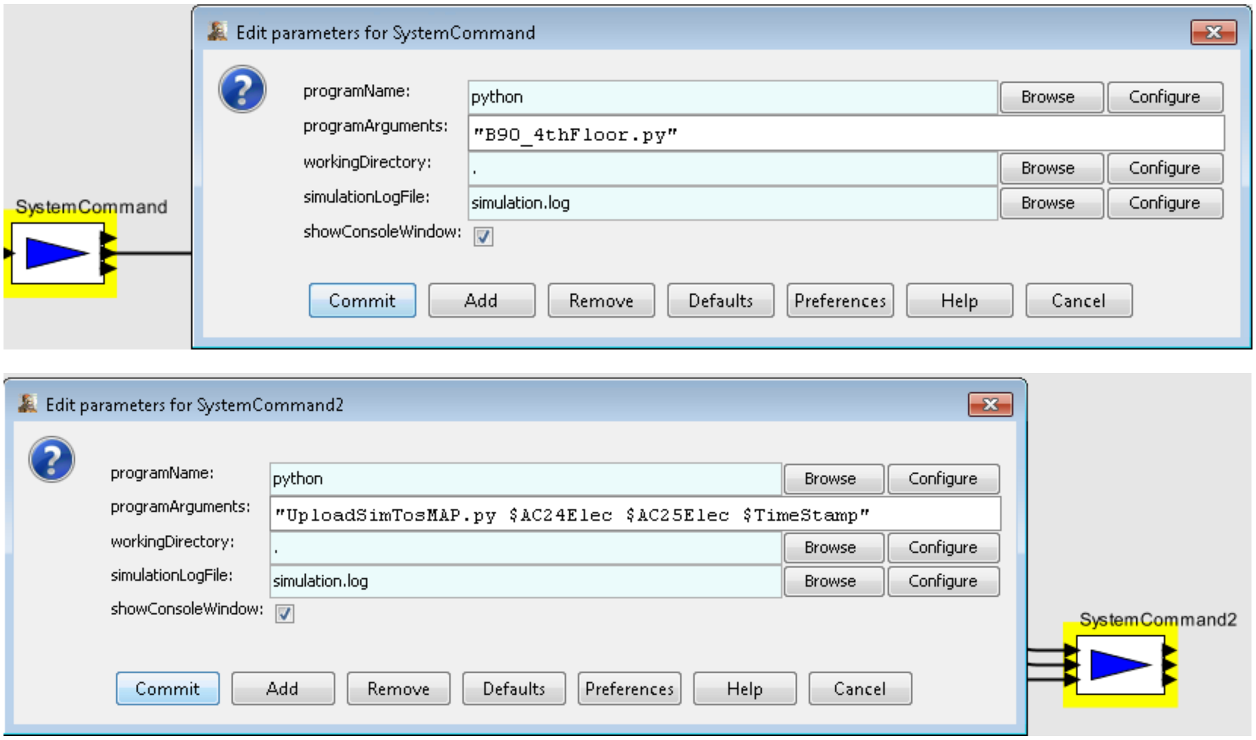


*Figure 8 Geometry model of the demonstration floor*

### BCVTB to sMAP data exchange

The BCVTB needs to exchange data with sMAP to acquire real-time model inputs and send the simulation results. This is achieved through the SystemCommand in the BCVTB, which executes a system command that is defined by the parameter *programName* and arguments *programArguments*, as shown in Figure 9. The upper image shows the SystemCommand that acquires the real-time model input (in this case, from the sMAP Archiver – in other cases directly from the real-time sMAP sources) by calling a python script. The lower image shows the SystemCommand that uploads the simulated result back to the sMAP Archiver through the python arguments.



Edit parameters for SystemCommand

programName: python

programArguments: "B90_4thFloor.py"

workingDirectory: .

simulationLogFile: simulation.log

showConsoleWindow: ☑

Edit parameters for SystemCommand2

programName: python

programArguments: "UploadSimTosMAP.py $AC24Elec $AC25Elec $TimeStamp"

workingDirectory: .

simulationLogFile: simulation.log

showConsoleWindow: ☑

*Figure 9 The BCVTB SystemCommand actor and its settings*

## Realization of the platform

Figure 10 shows the realization of the new real-time simulation platform in the BCVTB. The real-time simulation was launched by starting the BCVTB through the Graphical User Interface (GUI) or through the console. The simulation step time was 3 minutes in this demonstration. At each step time, the FMU receives the weather data as inputs and advances the model by one time step. Once the simulation is finished for that step time, the SystemCommand writes the results to the sMAP Archiver. It should be noted that any input available in the simulation software can be used as an input to the real-time simulation platform. For example, if a measurement of lighting power is available, it can be used to calculate the instantaneous lighting power density, which can then be used as a lighting internal heat gain input to the EnergyPlus model.

EnergyPlus contains a RunPeriod object to specify the *Begin and End days* of the simulation as shown in Figure 11. In the FMU, the *Begin and End days* are ignored and only the entry *Day of Week for Start Day* is used in the FMU. The *Begin and End days* for the real-time simulation are specified in the BCVTB GUI in seconds, as shown in Figure 10. For example, if the simulation begins on Jan 3rd and ends on Jan 5th, the *beginTime* and *endTime* are 86400*2 and 84600*5 respectively. As the EnergyPlus data input file can only specify the begin day and month, unless the simulation starts exactly at mid-night, a delay in the start of the simulation is needed in order to ensure the synchronization between the EnergyPlus simulation time and the computer clock time. In the BCVTB GUI, synchronization of the simulation time and real-time can be achieved by selecting the SDF Director and checking a check box, as shown in Figure 12.

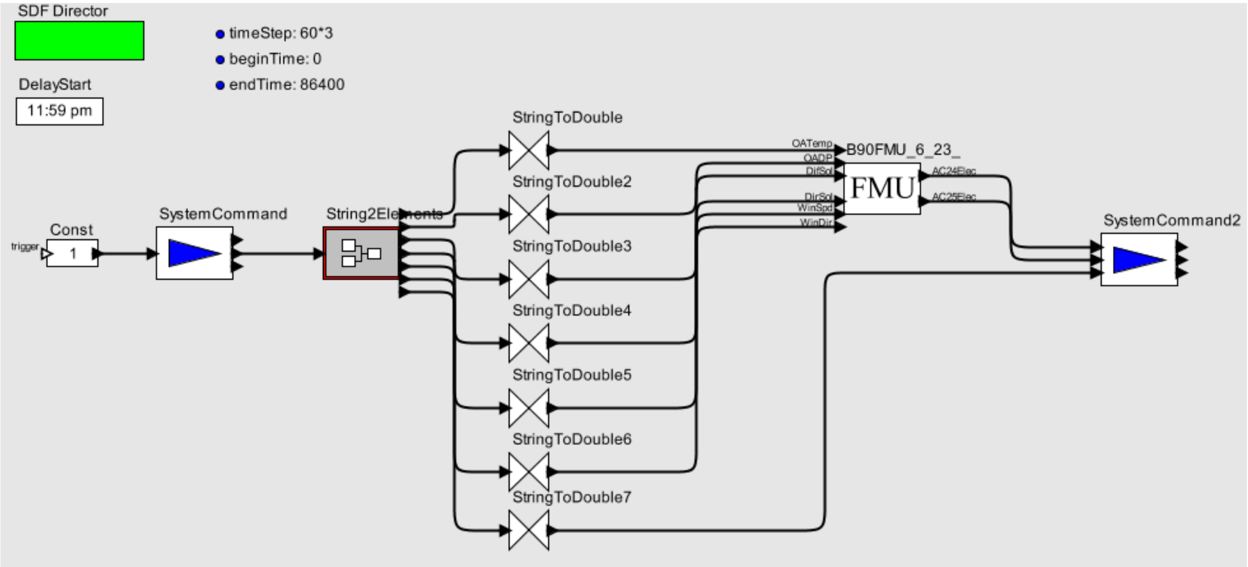
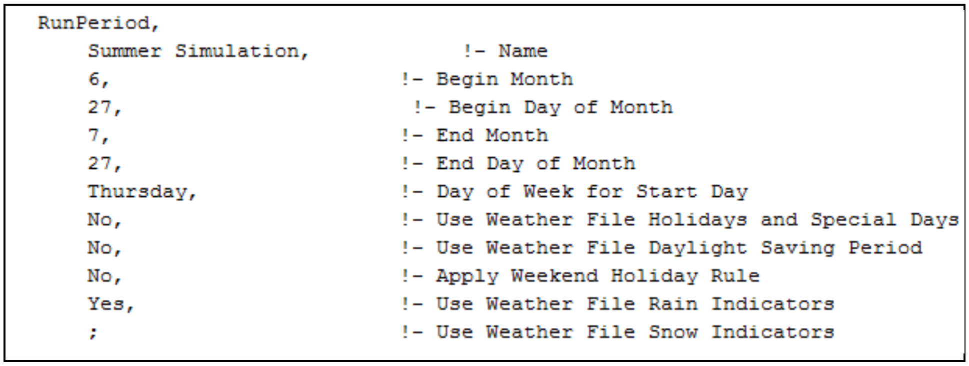


*Figure 10 Block diagram of the new real-time simulation platform*



```
RunPeriod,
    Summer Simulation,              !- Name
    6,                         !- Begin Month
    27,                         !- Begin Day of Month
    7,                         !- End Month
    27,                        !- End Day of Month
    Thursday,                  !- Day of Week for Start Day
    No,                        !- Use Weather File Holidays and Special Days
    No,                        !- Use Weather File Daylight Saving Period
    No,                        !- Apply Weekend Holiday Rule
    Yes,                       !- Use Weather File Rain Indicators
    ;                          !- Use Weather File Snow Indicators
```

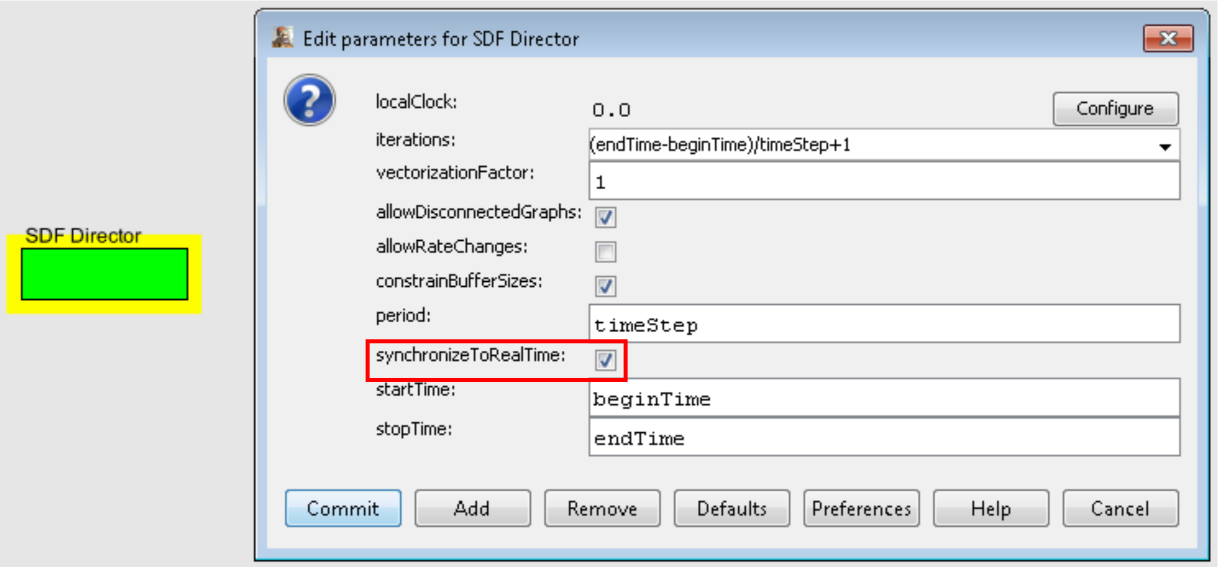
*Figure 11 The RunPeriod object in the EnergyPlus input file*

It should be noted that the simulation is stopped until real-time catches up. It is not guaranteed that the simulation executes within the allocated sampling time. The new platform is limited by the capability of the FMUs that are used with it. For instance, an FMU which supports a minimum time step of *x* will constraint the framework not to use a time step that is smaller than *x*. However, this is not a problem for EnergyPlus as the minimum time step in EnergyPlus is one minute and EnergyPlus is sufficiently fast to complete one step of simulation in one minute.

## *Discussion of experimental results*

### Execution time comparison

The first experiment was a comparison of the execution times of the EnergyPlus simulation for the two approaches that were used in the previous and current platforms, i.e. the BCVTB simulator interface and the BCVTB FMI interface. The simulation of the demonstration building was run for one year, using three different simulation time steps, i.e. 1 minute, 5 minutes and



localClock: 0.0
iterations: (endTime-beginTime)/timeStep+1
vectorizationFactor: 1
allowDisconnectedGraphs: ☑
allowRateChanges: □
constrainBufferSizes: ☑
period: timeStep
synchronizeToRealTime: ☑
startTime: beginTime
stopTime: endTime

*Figure 12 Realization of synchronization of simulation time to real-time in BCVTB*

10 minutes. The computer that ran the test has the Window 7 operating system with an i7-860 2.80 GHz processor. Table 1 presents the comparison results. As a reference, the execution time of running the model using EnergyPlus engine directly is also shown. As can be seen in Table 1, the execution times for the two approaches are comparable, with BCVTB FMI interface running slightly slower; this is to be expected because of the overhead of the FMI transaction layer. Since the simulation is run in real time, the effect of the difference in practice is negligible.

*Table 1 Execution time comparison*

| | Execution Time (second) | | |
|---|---|---|---|
| | BCVTB simulator | BCVTB FMI | Native EnergyPlus |
| 1-minute time step | 671.37 | 781.98 | 542.37 |
| 5-minute time step | 107.27 | 113.46 | 53.52 |
| 10-minute time step | 63.78 | 65.46 | 31.99 |

### FMU simulation results verification

In the second experiment, the FMU was run in the BCVTB and the EnergyPlus model was run directly, using the same weather data. The results of the two simulations were compared to verify the FMU performance. We used the Typical Meteorological Year 3 (TMY3) weather data for San Francisco to run the simulation. Figure 13 shows the electrical power consumption of the AC24 heat pump for both cases for the occupied hours for one day in summer. The results are the same at each step time, to the reported precision ($10^{-6}$ W).

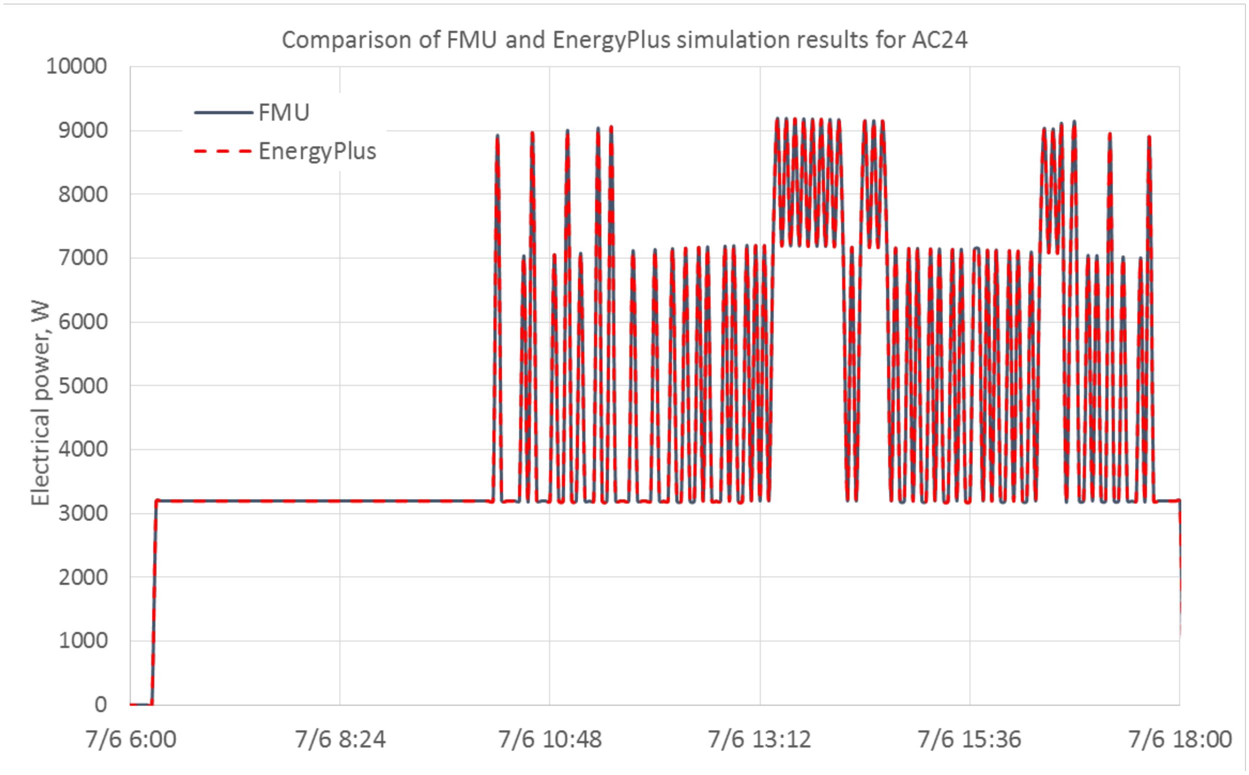


*Figure 13 Comparison of FMU and EnergyPlus simulation results of AC24 electrical power*

## Conclusion

A new realization of the real-time simulation framework has been developed and tested. The new realization features an open standard interface to simulation models and a wider range of protocols for communication with hardware, enabled by utilizing the FMI and sMAP. The configuration and setup of the new realization is much simplified in comparison to the previous implementation, which will make it easier to scale up the use of real-time simulation. The new realization could be used in various contexts to improve building energy performance, such as model-based fault detection and diagnosis, model predictive control, model-based commissioning and dynamic building performance benchmarking.

## Acknowledgement

This work was supported, in part, by the Assistant Secretary for Energy Efficiency and Renewable Energy, Office of Building Technology, State and Community Programs of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231, by the Energy Efficiency and Resources Core Technology Program of the Korea Institute of Energy Technology Evaluation and Planning (KETEP) granted financial resource from the Ministry of Trade, Industry and Energy, Republic of Korea (No. 20132010101800) and by the Environmental Security Technology Certification Program (ESTCP) of the U.S. Department of Defense (project EW09-29).

## References

[1] R. Judkoff and J. Neymark, International Energy Agency building energy simulation test (BESTEST) and diagnostic method, No. NREL/TP – 472-6231, National Renewable Energy Lab., Golden, CO, 1995

[2] COMNET: http://www.comnet.org/mgp

[3] X. Pang, M. Wetter, P. Bhattacharya, P. Haves, A framework for simulation-based real-time whole building performance assessment, Building and Environment **54** (2012), 100-108

[4] EnergyPlus: http://www.energyplus.gov

[5] Ptolemy Project: https://ptolemy.eecs.berkeley.edu/ptolemyII/index.htm

[6] BCVTB: https://simulationresearch.lbl.gov/bcvtb

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

[8] D.B. Crawley, L.K. Lawrie, F.C. Winkelmann, W.F. Buhl, Y.J. Huang, C.O. Pedersen, P.K. Strand, R.J. Liesen, M.J. Witte and J. Glazer, EnergyPlus: creating a new-generation building energy simulation program, Energy and Buildings 33 (2001) 319-331

[9] Y. Kwak, J. Huh and C. Jang, Development of a model predictive control framework through real-time building energy management system data, Applied Energy 155 (2015) 1-13

[10] H. Moon and M. S. Choi, Continuous commissioning through real-time simulation with a building energy monitoring system, Sensor Letters 12 (2014) 967-973

[11] BACnet: http://www.bacnet.org

[12] T. S. Nouidui, M. Wetter, Z. Li, X. Pang, P. Bhattacharya and P. Haves, BACnet and analog/digital interfaces of the Building Controls Virtual Test Bed, Proceedings of Building Simulation 2011: 12th Conference of International Building Performance Simulation Association, Sydney, November 14-16, 2011

[13] Claudius Ptolemaeus, Editor, System Design, Modeling, and Simulation using Ptolemy II, Book, available at: http://ptolemy.org/books/Systems

[14] sMAP 2.0 documentation, available: http://www.cs.berkeley.edu/~stevedh/smap2/index.html

[15] MODELISAR-Consortium, Functional mock-up interface for model-exchange, 2010, available: https://svn.modelica.org/fmi/branches/public/specifications/v1.0/FMI_for_ModelExchange_v1.0.pdf

[16] MODELISAR-Consortium, Functional mock-up interface for co-simulation, 2010, available: https://svn.modelica.org/fmi/branches/public/specifications/v1.0/FMI_for_CoSimulation_v1.0.pdf

[17] MODELISAR-Consortium, Functional mock-up interface for product lifecycle management, 2011, available: https://svn.modelica.org/fmi/branches/public/specifications/v1.0/FMI_for_PLM_v1.0.pdf

[18] Modelica Association, Functional mock-up interface for model-exchange and co-simulation, 2014, available: https://svn.modelica.org/fmi/branches/public/specifications/v2.0/FMI_for_ModelExchange_and_CoSimulation_v2.0.pdf

[19] T. S. Nouidui, D. M. Lorenzetti and M. Wetter, EnergyPlusToFMU (2013), available: http://simulationresearch.lbl.gov/fmu/EnergyPlus/export/userGuide/index.html

[20] Thierry S. Nouidui and Michael Wetter, Tool coupling for the design and operation of building energy and control systems based on the Functional Mock-up Interface standard, Proceedings of the 10th International Modelica Conference, pp. 311-320, Lund, Sweden, March 2014