Energy and Buildings 66 (2013) 88–103

Contents lists available at ScienceDirect

Energy and Buildings

journal homepage: www.elsevier.com/locate/enbuild

ROBESim: A retrofit-oriented building energy simulator based on EnergyPlus

Jun Wei Chuah [a,*], Anand Raghunathan [b], Niraj K. Jha [a]

[a] Department of Electrical Engineering, Princeton University, Princeton, NJ 08544, USA
[b] School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47907, USA

ARTICLE INFO

Article history:
Received 8 April 2013
Received in revised form 19 June 2013
Accepted 7 July 2013

Keywords:
Energy conservation
Green buildings
Building energy simulation
EnergyPlus
ROBESim

ABSTRACT

Buildings account for 21% of global energy consumption. Reducing building energy consumption can result in an immediate impact on energy expenditures and the environment. Retrofitting existing buildings is a means toward reducing their energy footprint. Currently, building owners have several retrofit choices. Building energy simulations can be used to assist in this retrofit selection process, providing energy savings estimates from different retrofits. Currently available simulators, however, are not retrofit-oriented and are not built to enable comparisons among retrofits. In order to perform these comparisons, users are required to manually create building models for each retrofit, which is obviously a cumbersome process. In this paper, we introduce ROBESim, a retrofit-oriented building energy simulator. ROBESim is based on the popular EnergyPlus framework, and relies on EnergyPlus for most of the supported computations. By using the retrofit modules in ROBESim, the user can quickly and easily generate building models to perform retrofit comparison simulations. ROBESim was designed to be modular and extensible. We outline these modules and show how developers can build upon our work in ROBESim. In addition, we describe the retrofit module development process and discuss additional ease-of-use enhancements that make building energy simulation possible for users of all experience levels.

1. Introduction

Energy expenditures account for 10% of global gross domestic product (GDP) in 2010, an increase from 8% in 2008 [1]. Energy expenditures are second only to healthcare costs in many countries. In 2010, 81% of the total global energy consumption was derived from fossil fuels, such as coal, oil, and natural gas [2]. Energy generation from these fossil fuels result in the release of greenhouse gases and other air and water pollutants, leading to environmental degradation and health concerns. In addition, increased energy expenditures contribute to a reduced standard of living. Reducing global energy consumption is critical to alleviating these concerns.

Buildings represent an important sector for energy conservation measures. Residential and commercial buildings represent 21% of total global energy consumption [3]. In the United States, buildings represent 40% of total national energy consumption [4,5]. Thus, a reduction in building energy consumption can have a significant impact on global energy consumption and, consequently, global energy expenditures.

Although advancements in building design and architecture can help reduce global building energy consumption, the long life cycle of buildings (and their corresponding slow replacement rate) results in a long period of time before these advancements become mainstream. In 2003, only 26% of the commercial buildings in the United States were constructed in the previous ten years [6]. Thus, to achieve building energy conservation, retrofitting existing buildings is an equally important step. Building retrofits are systems installed to augment or replace existing building systems, resulting in improved energy efficiency and/or occupant comfort. Some examples of building retrofits include light-emitting diode (LED) lighting, small-scale renewable energy generation through solar panels or wind turbines, and self-programming thermostats such as the Nest learning thermostat [7].

Building owners looking to retrofit their building have several choices. To help decide among these choices, an energy analysis of the current building is needed. A building energy analysis provides information on the building's energy consumption, *i.e.*, the energy consumed in building systems, such as heating, ventilation, air conditioning (HVAC), and lighting. This allows the building owner to select retrofits that can have the most significant impact. As part of this energy analysis, building energy simulators, such as EnergyPlus [8,9], eQuest [10], DOE-2 [11], and TRNSYS [12], can be used. These simulators provide a quick and easy estimate for the energy consumption in a given building, based on the corresponding

* Corresponding author. Tel.: +1 609 937 3804.
E-mail addresses: jchuah@princeton.edu, junwei.chuah@gmail.com (J.W. Chuah).

0378-7788/$ – see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.enbuild.2013.07.020

climate and building characteristics. Building energy simulation, in general, provides sufficient information for the building owner to decide which retrofits to install. Other options are available for more in-depth building energy analysis. These options will be further discussed in Section 2.

Although most of the building energy simulators mentioned above are freely available (EnergyPlus, eQuest, and DOE-2), they require a certain level of technical expertise to use. Thus, the typical building owner has to rely on trained professionals to perform this building energy analysis, resulting in additional costs. Although the analysis performed by trained auditors is, in general, more customized and, thus, more accurate, the additional costs involved could serve as a significant barrier for many building owners. This is especially true for residential home owners with a lower retrofit budget.

In addition to their complexity, most building energy simulators are not retrofit-oriented, *i.e.*, they do not contain built-in functionality to automatically apply retrofits and compare among different retrofit choices. Thus, users have to manually create building models to simulate across different retrofit choices. This is to be expected, since the primary purpose of these building energy simulators is to provide accurate simulation frameworks that can be built upon by additional software. Nonetheless, the lack of built-in retrofit support results in additional effort on the part of the building owner, providing another barrier to entry.

In this paper, we introduce ROBESim, a retrofit-oriented building energy simulator based on the EnergyPlus framework. The release version of ROBESim is compatible with EnergyPlus v8.0. We chose EnergyPlus because it is freely available, widely used in both academic and commercial applications, and has a comprehensive feature set. In addition, the core design philosophy of EnergyPlus is such that wrappers and add-ons can be easily developed around it; since EnergyPlus relies on text-based input and output files, this allows for easy external invocation by wrapper programs.

ROBESim comprises two parts: the core simulation framework and ease-of-use enhancements (in the form of user interfaces and input file generators). The core simulation framework extends EnergyPlus by allowing built-in definitions and separate computations for retrofits, *i.e.*, a retrofit developer can easily write a module that can be processed in ROBESim for energy analysis. For building owners, along with the release version of ROBESim, we provide a database of popular existing retrofits and some interesting retrofits that are under development, *e.g.*, better insulating windows, programmable thermostats, and smart thermostats [13–16]. This ensures that the user will have an immediately available retrofits database for comparison. We discuss the included retrofits in detail in Section 5.

Since a number of novel retrofits depend on room occupancy data and/or detailed occupant positioning information, the ROBESim core simulation framework also provides built-in support for occupant positioning information, in the form of occupant profiles. The occupant profiles contain information on the positions of the occupants throughout the simulation period. This information can be used to better gauge the energy savings that can be derived from occupant-aware retrofits. Occupant-aware retrofits are retrofits that rely on occupant information, such as the occupants' positions, in their function. Examples include the Nest learning thermostat [7] and smart lighting [17–19]. The energy savings enabled by these retrofits depend on the occupancy status of the building and/or exact occupant positions. Thus, to accurately evaluate these retrofits, occupant profiles must be incorporated during simulation. We discuss a number of additional retrofits that depend on occupant positioning information in Section 2.

To support the ROBESim simulation framework, we include a number of ease-of-use additions. We target all experience levels in our approach, *i.e.*, we have developed interfaces for casual building owners, retrofit developers, and simulation software developers. Specifically, for casual building owners, we have developed a user interface for quick and easy input, simulation, and comparison across our retrofits database. For retrofit developers, we provide offline software and libraries that expose all the functionality provided by the ROBESim core. This enables the retrofit developer to quickly develop retrofit modules for deployment in ROBESim. Apart from the development of modules for currently available retrofits, ROBESim can also be used in the preliminary development stages for novel retrofits. We detail examples of some retrofit modules we developed in Section 5. For software developers looking to extend our work, we provide ROBESim libraries, along with source code for the interfaces that enable batch application of retrofits and one-click simulation across multiple building models. These simulator usage scenarios are further explored in Section 4.

In addition to user interfaces, ROBESim includes a number of input file generators that, based on the provided information, can quickly and easily generate the required input files (namely, the building model and occupant profiles). The implementation details for these generators are further discussed in Section 7.

The ROBESim core simulation framework and ease-of-use enhancements were developed using the Java programming language (in comparison, the core EnergyPlus framework is written in Fortran). We chose Java since it is one of the most commonly taught programming languages. In addition, for users looking to extend our work, there is minimal effort required to include and invoke the available ROBESim external libraries.

In summary, we seek to achieve two goals in our ROBESim design. The first is to provide quick and easy retrofit development and retrofit-oriented simulation tools. The second goal is to support retrofits that depend on occupant positioning information. The key features of ROBESim are as follows:

- *Built-in support for building retrofits.* ROBESim automatically applies the chosen retrofits, and provides comparisons with the original building models.
- *Support for occupant profiles.* For applications that depend on occupant-positioning or occupancy status, ROBESim provides support for generating and utilizing occupant profiles.
- *User-friendly interfaces.* ROBESim includes ease-of-use additions in the form of a one-click web interface, along with easy input file generation.
- *Modular and extensible.* ROBESim follows the modular coding style of EnergyPlus. All components in ROBESim are modular, and can be easily extended. In addition, ROBESim is programmed in Java, allowing for easier extension for those familiar with the language.
- *EnergyPlus compatibility.* As far as possible, ROBESim retains the EnergyPlus input and output formats, with the exception of additions, such as retrofit and occupant profile definitions. Further, ROBESim relies on the EnergyPlus simulation framework as much as possible, only performing computations that are not supported under EnergyPlus.

The paper is organized as follows. In Section 2, we discuss some related work in the field of building energy analysis and simulation. We also discuss some interesting building retrofits that are currently available or under development. In Section 3, we provide an overview of ROBESim, our retrofit-oriented building energy simulator, describe the input and output files, and outline the simulation workflow for one run of ROBESim. In Section 4, we highlight some usage scenarios for ROBESim and discuss the wrapper functions that we have developed to assist in these usage scenarios. In Section 5, we describe how new retrofit modules can be developed and applied under ROBESim. We also present the built-in database of retrofit modules that we have developed for ROBESim. In Section 6,

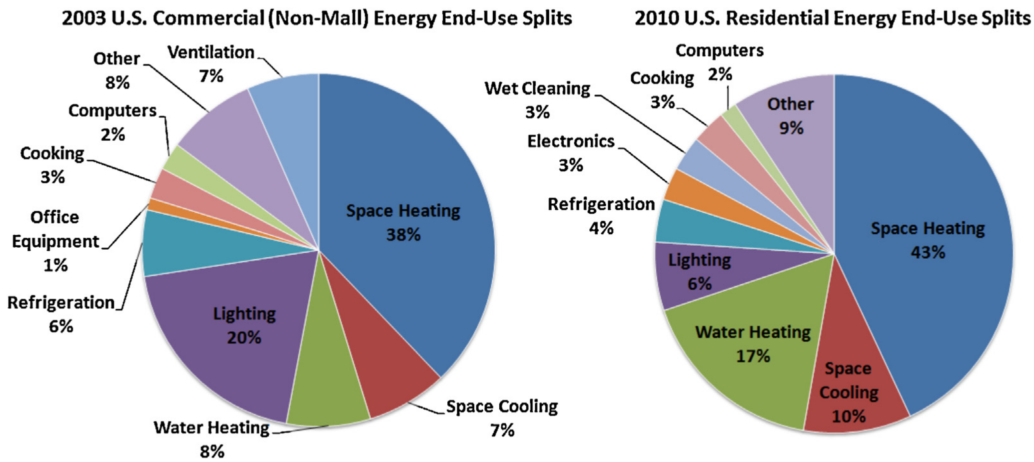

Fig. 1. Energy end-use splits for commercial and residential buildings in the United States [6,52].

we detail the components in the ROBESim core simulation framework. In Section 7, we outline some input file generators that we have developed for ROBESim to help with generating building models and occupant profiles. In Section 8, we outline and analyze some examples of simulations that can be run in ROBESim. We then conclude in Section 9.

2. Background

Building owners currently have several choices when considering retrofitting their buildings. These retrofit choices target one or more building systems. Fig. 1 illustrates the energy end-use splits for buildings in the United States. The end-use splits also highlight the building systems that are present in a typical building: space heating, space cooling, ventilation, lighting, water heating, refrigeration, computers, and other plug loads (electronic appliances).

Building energy analysis provides information on the energy consumption within a building. Typically, this information includes the total energy consumption, energy expenditures, and the breakdown of energy usage into their different end-uses, *i.e.*, information similar to Fig. 1, but tailored to the specific building under analysis. Using this information, building owners can then select the retrofits that are best suited for their respective buildings, *e.g.*, a building with a significant proportion of energy consumed through space heating will generally benefit more from improvements to the heating system, as opposed to another building with minimal energy consumption in space heating.

There are currently a number of approaches to building energy analysis, varying in terms of cost and precision of information. Smart meters can be used to capture the total energy consumption in a building. These meters can be provided by the utility or self-installed [20]. However, the granularity of information provided by smart meters is often insufficient. Building owners seeking to identify the major end-uses within their buildings are unable to do so with basic smart meters. Nonetheless, smart meters represent a cost-effective means to measure and track building energy consumption before and after retrofitting.

In order to obtain more fine-grain information through metering, more advanced techniques have to be implemented. These approaches can be classified into two broad categories: single-point metering [21,22] and multi-point metering [23–28]. Under single-point metering, appliances are classified based on their signatures, using a sensing system deployed at a single location within the building. In [22], the ElectriSense system is plugged into any power socket in the building. The sensor then tracks the electromagnetic interference in the power lines within the building, to determine which appliances are switched on. Under multi-point metering, sensors are deployed on all the appliances in the building. The sensors detect whether the appliances are switched on or off. Both metering systems result in information on the state (on/off) of all the appliances in the building. By correlating this information with a smart meter, the energy consumption attributed to each appliance can then be determined.

For a comprehensive evaluation of building energy consumption, an in-depth building energy audit can be performed [29–31]. Building energy audits involve a combination of metering and simulation to provide more accurate information. Understandably, this option is more costly than metering or simulation alone. However, most building energy audits include checks for buildings faults, such as air leakages and areas of poor or deteriorating insulation. Fixing these faults is often the most cost-effective means toward improving building energy performance.

For building owners looking for a quick energy analysis, building energy simulation is an attractive option. Several building energy simulators are currently freely available. We highlight EnergyPlus [8,9], eQuest [10], and DOE-2 [11] as some of the most popular options in building energy simulation. TRNSYS, although not free software, is also commonly used for building energy simulations. As previously discussed in Section 1, the information provided through building energy simulation is often sufficient for the building owner to make his retrofit choices. Unfortunately, existing building energy simulators are primarily focused on accurate computations for a single input set (a building model and weather data file). Further, each building model must be manually modified to consider additional retrofits. Our work improves on this situation by providing quick retrofit application, automated simulation across multiple building configurations, and comparison of the retrofit choices according to their expected impact on building energy performance.

In the previous section, we covered most of the reasons for our choice of EnergyPlus as the underlying simulation framework for ROBESim. Due to its flexibility, EnergyPlus has been used in the evaluation of unconventional building systems, *e.g.*, underfloor air distribution systems [32], dual airflow windows [33], thermal chimneys [34], and radiant cooling systems [35]. Standard EnergyPlus benchmark models have also been used to quickly estimate building energy consumption without the need for customized building models [36], highlighting the accuracy of the EnergyPlus simulation framework.

Several useful add-ons have been developed for the EnergyPlus simulation framework [37–42]. NewFacades [38] extends EnergyPlus to help analyze intelligent building facades in the early stages of the building design process. Using NewFacades, different intelligent facade combinations can be quickly generated and evaluated.

OpenStudio [39] extends EnergyPlus by providing a visual platform for building model generation and results viewing. Using OpenStudio, the process of building model generation is greatly accelerated. In addition, OpenStudio supports Radiance [43], a simulation tool commonly used to simulate indoor lighting conditions. The modularity of EnergyPlus is highlighted in OpenStudio, since the OpenStudio platform is able to effectively combine two different simulation tools, resulting in greater overall utility.

MLEPlus [40] provides a co-simulation platform between MATLAB [44] and EnergyPlus. This allows experienced MATLAB users to easily perform building energy simulations by invoking EnergyPlus from MATLAB. Such cross-platform tools are very useful as they enable users experienced in a given platform to easily invoke EnergyPlus, reducing the training period required for familiarization with EnergyPlus.

The use of text-based input and output files in EnergyPlus enables a suite of add-ons that provide graphical interfaces for the input and output files. xEsoView [41] provides a graphical user interface for the EnergyPlus standard output files, converting the text files into more readable graphs. These tools facilitate the use of EnergyPlus by simplifying the user interface. ROBESim includes a built-in building model generator for easy building file creation. Our building model generator is further discussed in Section 7.

ROBESim can be most closely compared to BEopt [42], a building energy optimization software used to analyze the performance of retrofits and building design choices. BEopt provides a simplified user interface for building modeling and retrofit selection. The user can then simulate across a number of retrofit choices. BEopt provides visual output of the energy savings and implementation cost that can be derived from the respective retrofit choices. Originally developed to consider the construction and material choices during building design (*e.g.*, to compare between double and triple-pane windows), BEopt has recently been updated to consider retrofits. However, the set of retrofits that are hard-coded in BEopt is limited to material and construction retrofits, basic thermostat schedules, and basic equipment efficiency improvements. Although the retrofits included are comprehensive, BEopt does not support novel retrofits. This is primarily due to the focus on building owners and designers, as opposed to retrofit developers. In contrast, we provide application programming interfaces (APIs) to access the basic modules included in ROBESim. Using these APIs, retrofit developers can quickly develop retrofit modules for their novel retrofits and perform batch simulations in ROBESim. This also allows building owners using ROBESim to simulate across a more complete list of retrofits, such as the Nest learning thermostat [7]. In addition, retrofit modules in ROBESim are not limited to functionality available in EnergyPlus, since the ROBESim co-simulator can perform computations that are not supported under EnergyPlus. This functionality is not available under BEopt. Nonetheless, we believe that the simplicity of the user interface presented in BEopt is unique in the field of building energy simulators, and serves as a model for future simulator development. In ROBESim, we aim to match this simplicity in our user interfaces.

Apart from supporting the automated exploration of retrofit choices, ROBESim also supports fine-grain occupant positioning information (in the form of occupant profiles). Once again, this differentiates ROBESim from BEopt, as occupant profile support is beyond the scope of BEopt. This is important for the analysis of novel, occupant positioning-based retrofits. In ROBESim, we provide built-in support for the following retrofits that depend on occupant profiles: the Nest learning thermostat [7], smart thermostats [13–16], smart lighting [17–19], and localized heating [45].

The Nest learning thermostat [7] improves on existing programmable thermostats in two ways. First, after a brief learning period, the Nest thermostat automatically manages the thermostat settings, based on the expected occupancy status of the building and the occupants' preferences. Second, the Nest thermostat contains a built-in motion sensor to detect the presence of occupants, adjusting the thermostat settings accordingly. Through this self-programming scheme, energy savings are realized when the building is vacant, allowing for a deeper setback in thermostat settings, with minimal discomfort (since the thermostat can predict when the building will be occupied again, it can preheat the building before the arrival of the occupants). Smart thermostats [13–16] operate in a similar fashion to the Nest thermostat, but rely on additional occupancy sensors, *e.g.*, motion sensors, door sensors, radio frequency identification tags, and low-power cameras. The combination of these additional sensors allows for better prediction, with the expense of increased implementation effort.

Smart lighting [17–19] relies on an occupant positioning system to determine the positions of all the occupants in a building. The positioning information is then used to determine the optimal lamp settings that satisfy the occupants' preferences. Localized heating [45] relies on ceiling-mounted directed radiant heaters to follow occupants around the building. This allows the thermostat to be set to a lower level and reduces heating energy in heating unoccupied spaces in the building.

The retrofits discussed here all rely, in varying degrees, on the occupant profiles supported in ROBESim. For the Nest and smart thermostats, coarse-grain occupancy status is sufficient, whereas smart lighting and localized heating rely on fine-grain occupant positioning information to fully analyze their performance. Although simulation of these retrofits is possible in EnergyPlus (requiring some careful schedule setting), these retrofits are natively supported in ROBESim, and can be easily deployed within the framework. A survey of other currently available building retrofits is provided in [46].

Previous work has incorporated building occupancy models and occupant preferences in building energy simulation [47–49]. The methods described in [47] and [48] can be used to generate coarse-grain occupancy information for simulation. In addition to this coarse-grain occupancy information, ROBESim supports fine-grain occupant positioning information. This is important for accurate simulation of retrofits that depend on such information, *e.g.*, smart lighting and localized heating. User preferences can also affect the operation of building systems [49]. Existing simulation tools do not take into account variations in user preferences during simulation. In the simulation of building HVAC, for example, fixed thermostat settings are used, regardless of the occupants that are in the building. Using the occupant profiles in ROBESim, the occupant preferences can be captured and simulated. For example, the thermostat setting can be made dependent on the preferences of the occupants that are in the building or zone. This better reflects real-world scenarios, where occupants frequently adjust the thermostat settings based on their personal preferences [49].

In summary, building energy simulation represents a quick and cost-effective means of estimating building energy consumption. However, existing simulators lack comprehensive support for building retrofits. Since one of the key uses of building energy simulation is to provide before-and-after information on the impact of retrofit choices, this is an important deficiency that we remedy with our ROBESim simulation framework. In addition, we provide native support for occupant profiles, an important component for occupant-aware retrofits.

3. Simulator overview

In this section, we discuss some of the key ROBESim components and their corresponding interactions. We provide an overview of the core simulation framework that ROBESim is based upon.

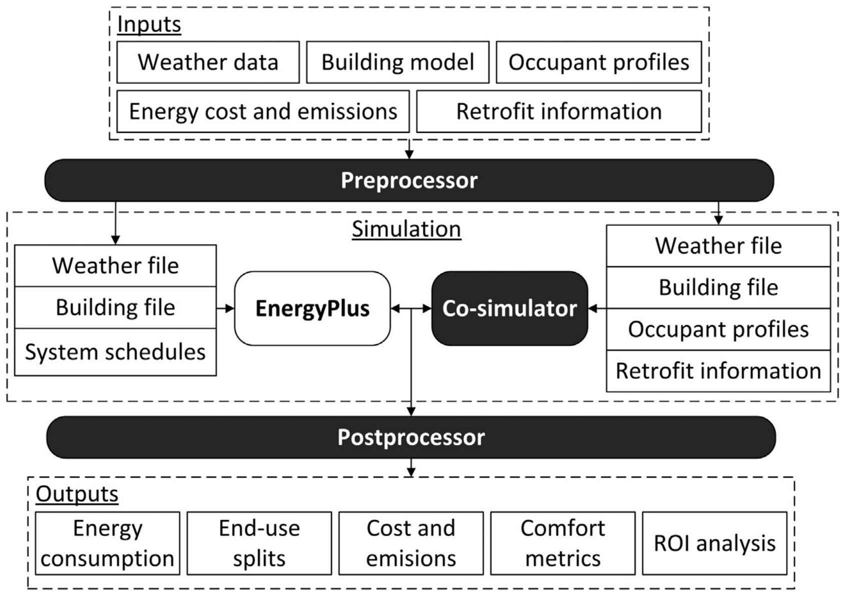

Fig. 2. Workflow for one ROBESim simulation run.

3.1. Simulator workflow

The workflow for one ROBESim simulation run is illustrated in Fig. 2. This simulation workflow is managed by the `SimulationManager` class in ROBESim. The simulation run is broken up into three phases: preprocessing, simulation, and postprocessing. In the preprocessing phase, the `Preprocessor` takes in the original inputs and produces two different sets of inputs. The first input set is EnergyPlus-compatible, and is used for all computations that are supported under EnergyPlus. The second input set is sent to the ROBESim co-simulator, for additional computations not supported by EnergyPlus. In the simulation phase, EnergyPlus is invoked externally (using the `EnergyPlusInvoker` class), based on the first set of EnergyPlus-compatible inputs. The `CoSimulator` is run either simultaneously with or after the completion of EnergyPlus, using the second set of inputs. The outputs from both EnergyPlus and the ROBESim co-simulator are then sent to the postprocessor. The `Postprocessor` summarizes these outputs in user-friendly terms, such as total energy consumption, end-use breakdown, and energy savings.

In general, depending on the applied retrofit modules, a ROBESim simulation run takes a slightly longer time than a single EnergyPlus simulation. Our analysis (explored further in Section 8) indicates a simulation overhead of less than one minute. Of course, the simulation runtime is extended for retrofit modules that involve more complex computations. Retrofit modules are software modules developed by retrofit developers to be used in ROBESim. In general, these modules reflect the functionality of the underlying retrofit. Retrofit modules are described in greater detail in Section 5.

The workflow shown in Fig. 2 represents a single ROBESim simulation run. Additional functions in ROBESim build upon this workflow. For example, it provides the means to run multiple instances of this simulation across different building models, climates, and retrofit choices; the outputs are then automatically combined and presented in comparison with one another. Another class, `StatsGenerator`, is used for this comparison step. We discuss these additional usage scenarios in greater detail in Section 4.

3.2. Inputs

The file-based input/output system of EnergyPlus ensures its modularity and ease of extension. To ensure compatibility and future extensibility, ROBESim follows the same file-based input/output system. In most cases, ROBESim inputs have identical formats as EnergyPlus inputs. For functionality beyond EnergyPlus, ROBESim relies on additional functions that are not supported in EnergyPlus. These functions and their definitions are clearly defined in the ROBESim documentation.

The main inputs of each ROBESim simulation run are as follows:

- *Weather data.* ROBESim accepts EnergyPlus weather files, since they are widely available. A comprehensive list of weather files can be found in [50]. The weather data are exposed to users of the ROBESim libraries through built-in functions.
- *Building model.* The building model comprises two parts: an EnergyPlus-compatible building file and an extended building file used in ROBESim. The extended building file contains information related to the specific appliances in the building, along with their positions. This information can be used to develop smart retrofit modules in ROBESim.
- *Occupant profiles.* For each occupant in the building, there is a corresponding occupant profile containing the positions of the occupant across the simulation period. This information is used in retrofit modules for occupant-aware retrofits.
- *Energy cost and emissions.* This information is primarily used to compute total energy cost and carbon emissions in the post-processing stage, and varies according to the location of the simulated building.
- *Retrofit information.* Retrofits are defined in the retrofits database. The input retrofit information is used primarily to determine which retrofits are to be applied. By changing this retrofit information, multiple ROBESim simulations can be performed without manually modifying the main building file. Note that although the retrofit modules are hard-coded into ROBESim (in the retrofits database), we support the use of input parameters, *e.g.*, different sets of triple-pane window panels may vary in cost and insulation values, but can still use the same retrofit modules with different input parameters, significantly reducing the time needed to define these retrofits for simulation. We describe retrofit modules in greater detail in Section 5.

The selected retrofits are automatically applied in the preprocessor, based on the input building model and retrofit information.

```
ROBESim 0.1beta,Building-4Z-Light-Res-1;
2,1,TimeStep
8000,1,Localized Heating-1:HVAC:Electricity:Consumption:
   Energy [J] !TimeStep
End of Data Dictionary
2,0
8000,146880.0
2,1
8000,146880.0
```

Listing 1. Co-simulator interim output.

For example, if the selected retrofit is a triple-pane window panel, the input building model can be the original building with single-pane windows. The preprocessor clones the original building model and applies the selected retrofit, yielding a new building model with triple-pane windows. The parameters for the new triple-pane window are obtained from the respective retrofit information file. In this case, simulation across multiple retrofit combinations is simplified, requiring only a single input building model.

The inputs sent to EnergyPlus are the EnergyPlus-compatible weather and building files, along with a set of schedule files. The schedule files are used in EnergyPlus to simulate the schedules of building systems, *e.g.*, thermostat settings across time for the HVAC system and the lighting schedules for lamps in the building. The ROBESim co-simulator inputs include the extended building file, occupant profiles, the relevant retrofit information, and the same EnergyPlus weather file. Based on the retrofit information, the co-simulator can then determine which computations to perform. This information is specified in the corresponding retrofit module and is further described in Section 5.

3.3. Outputs

Interim outputs from both EnergyPlus and the ROBESim co-simulator are files in their respective formats. For ease of processing (in the postprocessor), the ROBESim co-simulator output uses the EnergyPlus standard output format (full documentation for the EnergyPlus standard output format is provided on the EnergyPlus website [8]). Thus, the same output file parser can be used to process both sets of outputs.

Listing 1 shows a snapshot of the interim output from the ROBESim co-simulator for localized heating [45]. Under localized heating, heating energy is split into two parts: the central gas furnace and the electric radiant heaters. The thermostat controlling the gas furnace is set to a minimum temperature of 10 °C. The additional heating requirement is covered by radiant heaters directed at the occupants' positions. This additional energy is simulated in the ROBESim co-simulator, based on the occupant profiles provided. We further detail the localized heating retrofit module in Section 5.

In Listing 1, the first line describes the simulator version, along with the building name. The data dictionary is then defined, using the following format: [unique identifier], [number of output parameters], [description for each output parameter]. This format is identical to the EnergyPlus standard output format. The output data follow this data dictionary. In the example above, for each of the first two timesteps, the localized heating radiant heaters consumed 146,880 Joules (*J*) of electricity. We describe the computations involved within the co-simulator in Section 6.

The ROBESim postprocessor collates the interim outputs into a combined output and derives additional sets of outputs from this combined output. The postprocessor is invoked with a file containing the required processing parameters, primarily involved in the categorization of energy consumption and generation. The default

```
ROBESim 0.1beta, Building-4Z-Light-Res-1-localized_heating\
   Building-4Z-Light-Res-1-localized_heating.fso;
2,1,TimeStep
1000,1,Consumption [J] !TimeStep
1001,1,Production [J] !TimeStep
1002,1,Gas [J] !TimeStep
1003,1,Electricity [J] !TimeStep
1004,1,Heating [J] !TimeStep
1005,1,Cooling [J] !TimeStep
1006,1,Ventilation [J] !TimeStep
1007,1,Lighting [J] !TimeStep
1008,1,HVAC [J] !TimeStep
End of Data Dictionary
2,0
1000,929573.1779989412
1001,0.0
1002,728017.06756364
1003,201556.1104353012
1004,874897.06756364
1005,0.0
1006,47701.0864353012
1007,6975.024
1008,922598.1539989412
```

Listing 2. Postprocessor combined output.

file allows the categorization of the end-uses (Fig. 1). By modifying this parameters file, the user can choose to combine the individual HVAC components into a combined HVAC category, for example. This is especially useful for retrofits that impact multiple end-uses at the same time (such as the smart and Nest thermostats). Listing 2 shows a sample postprocessor combined output, using modified parameters files, for one timestep. In this case, based on the data dictionary, the following information is captured: energy consumption, energy production, energy derived from natural gas, energy from electricity, heating energy, cooling energy, ventilation energy, lighting energy, and combined HVAC energy consumption. This combined output follows the same format as the co-simulator and EnergyPlus interim outputs.

Using this combined output, along with the cost and emissions information from the inputs, the postprocessor computes the desired final output. In ROBESim, this final output comprises the following information: total energy consumption, breakdown of energy consumption by end-use, total cost and emissions information, retrofit-specific comfort metrics, and return-on-investment (ROI) analysis. To incorporate additional metrics, alternative postprocessors can be developed. The output file parser is made available to users of the ROBESim libraries, allowing for quick development and incorporation of alternative postprocessors.

3.4. Summary

We have provided an overview of a single ROBESim simulation run in this section, along with the corresponding inputs and outputs. In Section 4, we describe the wrapper functions we have developed for different simulator usage scenarios, based on this single simulation run. These wrappers are built into the release version of ROBESim, and represent the core functionality of the simulation tool. In Section 6, we discuss, in greater detail, the core components of the ROBESim simulation framework, *i.e.*, the `SimulationManager`, `Preprocessor`, `CoSimulator`, and `Postprocessor` classes. We provide implementation details and discuss the functions contained within each component.

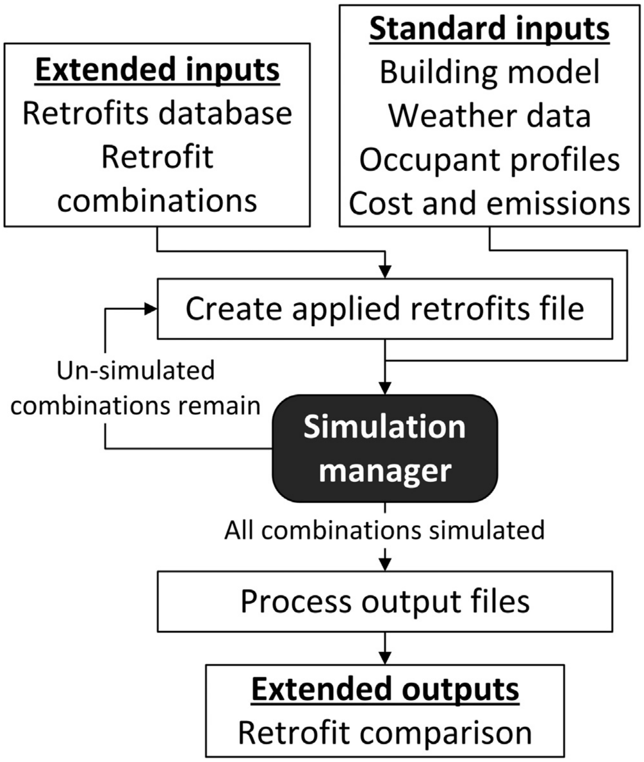

Fig. 3. Wrapper program for a building owner usage scenario (one building and climate zone to many retrofits).

4. Simulator usage scenarios

In this section, we discuss two usage scenarios for ROBESim: from the building or home owner's perspective, and from the retrofit developer's perspective. In addition, we describe the wrapper functions that we have developed for each of these usage scenarios.

4.1. Building owner

As discussed in Sections 1 and 2, building owners have several choices when retrofitting their buildings. One of the primary goals of ROBESim is to help these owners in their decision-making process. For these simulations, a single building model and weather file is used to simulate across different retrofit combinations, *i.e.*, one building and climate model to many retrofits. This can be achieved with the addition of a wrapper function.

Fig. 3 illustrates how this wrapper function works. In this case, the building model, weather data file, occupant profiles, and cost and emissions information remain the same for all simulation runs. The user provides a list of retrofits that are of interest; alternatively, he can use the default retrofits database provided in ROBESim. The wrapper function then applies these retrofits accordingly, invoking the simulation manager for each retrofit combination. When all the retrofit combinations have been explored, the wrapper function includes a summary report comparing the performance of the retrofit combinations across the output categories, *e.g.*, the total energy savings derived from each retrofit combination.

By using the wrapper function developed for building owners, the user can perform simulations across multiple retrofit combinations with minimal effort. To further simplify this process, we provide input file generators for the building file and occupant profiles (covered in Section 7). Using this suite of tools, the user can quickly and easily obtain an overview of the building's performance across different retrofit combinations. We provide simulation examples and results in Section 8.

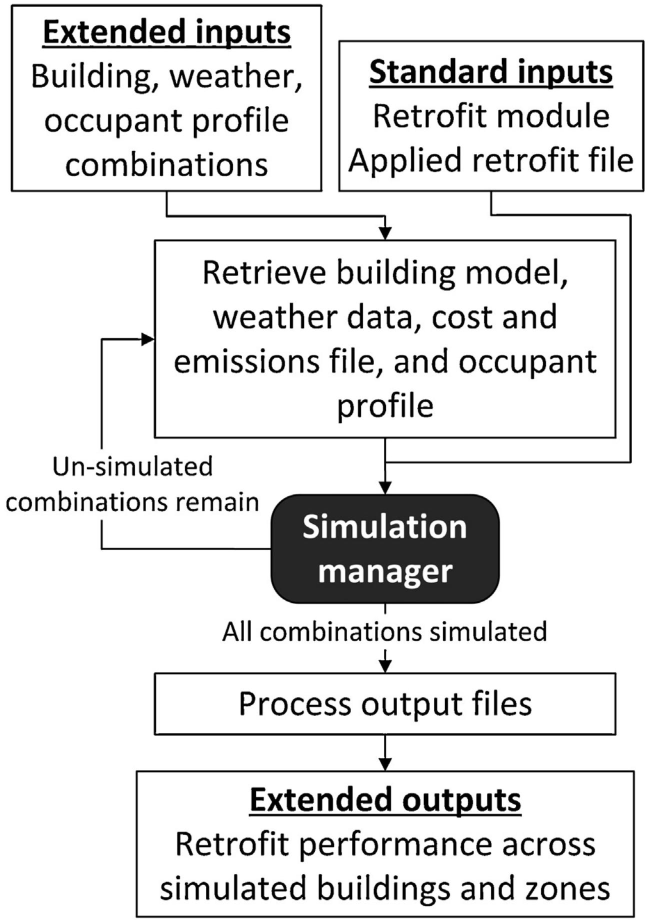

Fig. 4. Wrapper program for a retrofit developer usage scenario (one retrofit to many buildings and climate zones).

4.2. Retrofit developer

Whereas the building owner seeks to explore the performance of different retrofit combinations, the retrofit developer needs to evaluate the performance of his retrofit across different building models and climate zones, *i.e.*, one retrofit to many building models and climate zones. We developed a wrapper function that serves this purpose.

Fig. 4 illustrates the functionality of the wrapper developed for retrofit developers. This wrapper function is similar to that for building owners, with the inputs reversed. The retrofit information is sent straight to the simulation manager, whereas the building models and climate zones are retrieved accordingly. The occupant profile is linked to the respective building model, whereas the cost and emissions information is tied to the climate zone. Using this wrapper, a representative set of building models can be simulated across a given set of climate zones. The output from this wrapper highlights the performance of the retrofit in the given building-climate combinations.

To assist in building model and occupant profile generation, the input file generators that we have developed for ROBESim can be plugged into the wrapper function. In this case, the user only needs to input the relevant building characteristics and occupant profile heuristics, and the wrapper handles the generation and simulation process. We provide simulation examples and results for this wrapper in Section 8.

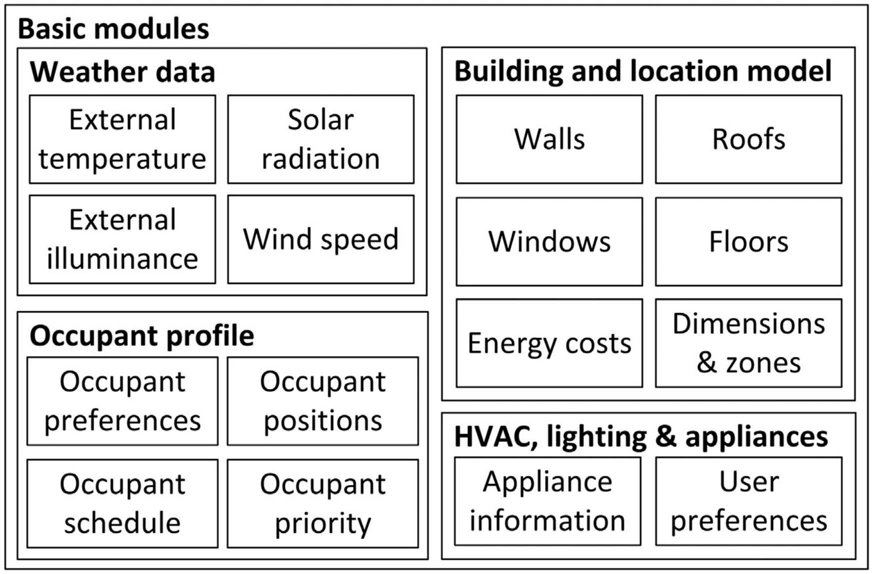

Fig. 5. Basic ROBESim modules.

4.3. Additional usage scenarios

Apart from the two primary usage scenarios above, we have developed ROBESim to be extensible to accommodate additional usage scenarios. Both wrapper functions build upon the single simulation workflow (presented in Section 3), with the simulation manager being run multiple times for each scenario. The general model for the wrapper functions is as follows:

- *Input file collection.* Different input file combinations are presented; the wrapper selects appropriate input files for each single simulation run.
- *Multiple simulations.* The simulation manager is invoked multiple times, with different sets of inputs.
- *Output processing.* The final set of outputs is collated and processed. The target information is then extracted from these outputs to provide the final output presented to the user.

This model can be used to develop additional wrappers that consider different input combinations and scenarios. For example, a multi-building, multi-retrofit wrapper can be developed to simulate across a diverse range of conditions. Beyond wrappers, we also provide external libraries for the core ROBESim simulation framework that can be used to incorporate additional usage scenarios, *e.g.*, combining ROBESim with other EnergyPlus add-ons, allowing ROBESim to be extended by simulation software developers for additional functionality.

5. Retrofit description and management

In this section, we define the retrofit modules used in ROBESim, and highlight the tools and methods available to retrofit developers who develop these retrofit modules. We first outline the basic modules present in ROBESim to provide an overview of the information available to the developer. Second, we present the retrofits database that is included in ROBESim. We then describe how retrofits are processed in the ROBESim preprocessor and co-simulator. To serve as a case study for retrofit developers, we describe the development of the localized heating retrofit module.

5.1. Basic modules

The set of basic modules available in the ROBESim libraries is presented in Fig. 5, along with some of the information that can be accessed from the modules. Each of these modules is a class within ROBESim, and the information is accessed through built-in methods for these classes.

File parsers are available from the ROBESim libraries for the input weather data, building model, and occupant profiles. Upon running the corresponding parser on the input file, a `WeatherDataset`, `BuildingModel`, or `OccupantProfile` object is created; these objects allow access to the data contained in the input files through their respective methods. For example, the `WeatherDataset` class comprises an array of `WeatherData` objects. Each `WeatherData` object stores the corresponding weather information for a single timestep. The set of methods used for accessing this information is presented in Listing 3. The data stored in the `OccupantProfile` class can be similarly accessed, allowing the retrofit developer to write retrofit modules that depend on occupant preferences and positions.

The `BuildingModel` class reflects the building models used in EnergyPlus. Most of the core components in EnergyPlus, such as `BuildingSurface`, `Construction`, `Material`, and `FenestrationSurface` have a corresponding ROBESim object. This allows for manipulation of the building model within ROBESim, as well as a built-in means to access the required building model information. For example, to apply a triple-pane window retrofit module

```
WeatherData:
    public double getTemperature();
    public double getHumidity();
    public double getGlobalHorizontalRadiation();
    public double getDirectNormalRadiation();
    public double getDiffuseHorizontalRadiation();
    public double getIlluminance();
    public double getWindDirectionDeg();
    public double getWindSpeed();
```

Listing 3. Methods for accessing information in `WeatherData`.

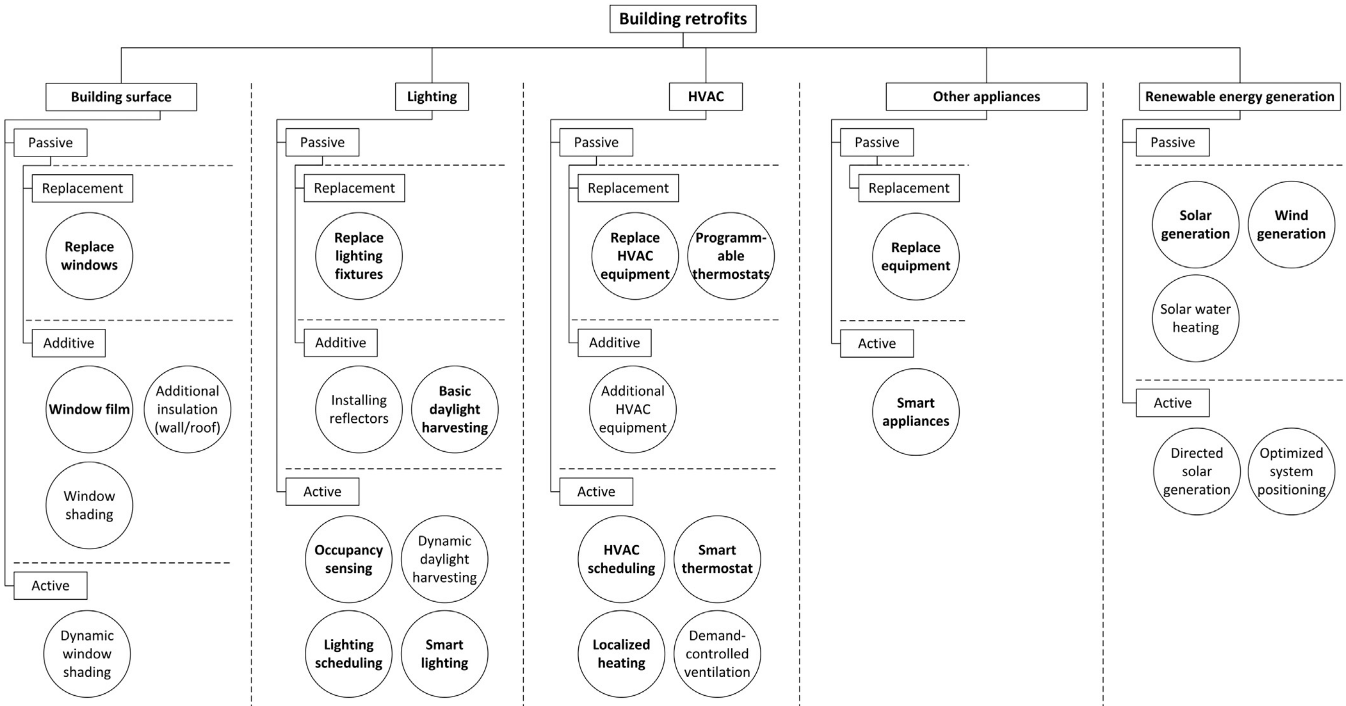

Fig. 6. Categorized set of currently available building retrofits. Retrofits in bold are included in the ROBESim retrofits database.

using ROBESim, the building model can first be accessed to determine the number and size of windows in the building. The retrofit module can then target the triple-pane window by replacing all the windows (fenestration surfaces) in the building model. We further detail retrofit processing under ROBESim later in this section.

As an added benefit to the mirroring of core EnergyPlus components, a developer can generate a building model within ROBESim, and use the built-in methods to convert this building model into an EnergyPlus-compatible file. This allows for automated building model generation. We discuss our version of this building model generator in Section 7.

The final component in the basic modules set involves information related to building systems such as HVAC, lighting, and appliances. EnergyPlus provides full support for HVAC definitions but limited support for both lighting and appliance definitions. In ROBESim, we incorporate most of the commonly used HVAC templates. For lighting and appliances, we include additional information, such as the positions of the appliances, that is not available in EnergyPlus. Since this extended lighting and appliance information is not supported in the EnergyPlus building file, it is exported to and read from a separate file. This file forms the extended building file used in ROBESim simulations (see Section 3).

5.2. Retrofits database

Before discussing the process involved in applying retrofits, we first describe the retrofits database that is provided in ROBESim. In addition, we categorize these retrofits and describe how the retrofit categories affect the simulation process. Fig. 6 illustrates a categorized set of currently available building retrofits, along with the retrofits (in bold) that are included in the retrofits database, *i.e.*, retrofit modules that are available along with the release of ROBESim. This list is non-exhaustive; for more information, a comprehensive survey of currently available building retrofits is provided in [46].

From Fig. 6, we identify retrofits in the following building categories: building surface, lighting, HVAC, appliances, and renewable energy generation. Building surface involves retrofits that affect the building envelope, typically based on the use of better insulating materials. Lighting, HVAC, and appliance retrofits involve improvements to the respective building systems. Renewable energy generation involves the small-scale generation of electricity through solar panels and wind turbines. ROBESim handles retrofits from different building categories separately. For example, for building surface retrofits, the input handler expects a `Material` and/or `Construction` object, along with the required information.

We classify retrofits into the *passive* and *active* types. In ROBESim, passive retrofits are retrofits that do not rely on the additional information in ROBESim, *i.e.*, they are natively supported under EnergyPlus. Conversely, active retrofits rely on additional information, such as occupant profiles and other relevant information (see Fig. 5). Active and passive retrofits are processed separately in ROBESim (discussed in the next subsection).

Passive retrofits are further classified into replacement and additive retrofits. Replacement retrofits replace existing building systems or constructions; only one replacement retrofit can be applied to each building system or construction, *e.g.*, a triple-pane window retrofit replaces existing windows and cannot be applied along with a double-pane window retrofit. Additive retrofits are additions to existing systems or constructions, and multiple additive retrofits can be applied to the same system or construction, *e.g.*, additional insulation added to the roofs of the building.

Each retrofit module comprises two parts: the code for the module in ROBESim and a parameters file. We use the parameters file to capture variations among similar building retrofits while preserving the core module code, *e.g.*, to support window retrofits of different materials and constructions, only one module needs to be written; the additional material and construction information is read from the parameters file. The parameters file for a triple-pane window retrofit is presented in Listing 4.

In Listing 4, the respective window materials and constructions are included in the parameters file. In this way, all replacement window retrofits can share the same code module in ROBESim,

```
Window,                  ! Category
Replacement,             ! Type
Triple Glazed Window,    ! Name
Window,                  ! Surface Type
100.0,                   ! Cost per window
0,                       ! Additional cost by area
15.0,                    ! Installation duration per window
0;                       ! Additional duration by area

! Materials used in current retrofit (EnergyPlus format)
WindowMaterial:Glazing,
    CLEAR 3MM,
    SpectralAverage,
    ,
    0.003,
    0.837,
    0.075,
    0.075,
    0.898,
    0.081,
    0.081,
    0,
    0.84,
    0.84,
    0.9;

WindowMaterial:Gas,
    AIR 6MM,
    Air,
    0.006;

! Construction that defines the retrofit (EnergyPlus format)
Construction,
    Trpl Clr 3mm/6mm Air,   ! Name of Construction
    CLEAR 3MM,              ! Layer 0 (Exterior)
    AIR 6MM,                ! Layer 1
    CLEAR 3MM,              ! Layer 2
    AIR 6MM,                ! Layer 3
    CLEAR 3MM;              ! Layer 4 (Interior)
```

Listing 4. Parameters file for a triple-pane window.

e.g., a double-pane window can be simulated by just modifying the `Construction` object in a new parameters file.

5.3. Retrofit processing

Retrofits are processed in two stages in ROBESim: the preprocessor and the co-simulator. Fig. 7 summarizes the operations that are performed in each of these two stages. The actual operations are defined in the respective retrofit modules.

In the preprocessing phase, the preprocessor takes, as input, the building model and retrofit parameters. Based on these parameters, the corresponding retrofit module in ROBESim is invoked to determine the expected implementation cost and duration. Listing 5 shows the required methods in the `Retrofit` interface in ROBESim. All retrofit modules will implement this interface, ensuring that the same functionality is available across different retrofit modules.

In addition to determining the retrofit implementation cost and duration, the preprocessor generates the retrofitted building model using the `applyRetrofit` function of the retrofit module on a cloned building model. Listing 6 shows the pseudo-code for the `applyRetrofit` function in a window retrofit module.

The function retrieves a list of all the windows in the building and sets their construction to match the new window construction.

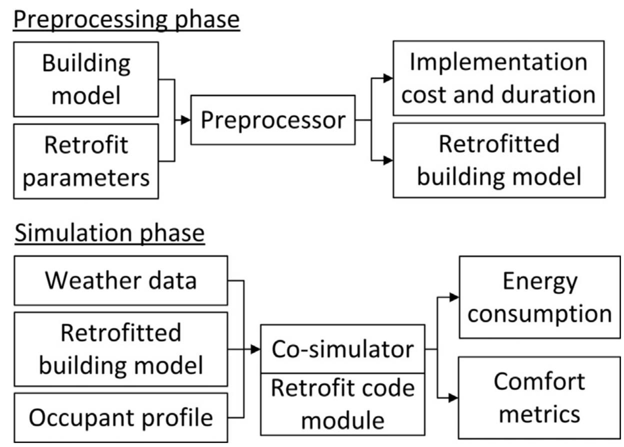

Fig. 7. Outline of retrofit input processing under ROBESim.

```
public interface Retrofit {
    // Retrofit parameters
    public String getName();
    public RetrofitType getType();
    public RetrofitCategory getCategory();

  // Implementation cost and duration

  public double getImplementationCost(BuildingModel
     building);
  public double getImplementationDuration(BuildingModel
     building);

  // Apply this retrofit to the given building model
  public void applyRetrofit(BuildingModel building);
}
```

Listing 5. Functions in the `Retrofit` interface.

In this example, the retrofitted building model is EnergyPlus-compatible, and there are no additional computations required in the co-simulator.

For active retrofits, such as localized heating, that require co-simulator computations, these computations are performed in the simulation phase. As shown in Fig. 7, the co-simulator invokes the retrofit module with the relevant information to determine the energy consumption and other metrics. Listing 7 shows the required functions in the `ActiveRetrofit` class.

In addition to energy consumption information, the `ActiveRetrofit` class supports easy generation of reports. A separate `ReportManager` class is updated at each timestep with the reports from all active retrofits. These reports form the co-simulator interim outputs in Listing 1 (see Section 3).

```
public void applyRetrofit(BuildingModel building) {
    // Apply construction to all windows in building
    for all windows in building {
        set window construction to construction described in
           parameters file
    }
}
```

Listing 6. Pseudo-code for `applyRetrofit` function in triple-pane window retrofit module.

```
public abstract class ActiveRetrofit implements Retrofit {
    // For retrofits with stored local information
    public abstract void init();

    // Get energy consumed or produced, based on occupant
       profile and weather data
    public abstract double getEnergy(OccupantData
       occupantData, WeatherData weatherData);

    // For generating reports
    public String getReportKey();

    // Format: [Number of report items],[Name]:[
       RetrofitCategory]:[FuelType]:[Consumption/Production]:
       Energy[J]
    public abstract String getReportHeader();
    public abstract String getReport(OccupantData
       occupantData, WeatherData weatherData);

}
```

Listing 7. Functions in the `ActiveRetrofit` class.

```
public void applyRetrofit(BuildingModel building) {
    // Change heating setpoint for all thermostats in the
       building
    for all thermostats in building {
        set heating setpoint to minimum temperature (10
            degrees celsius)
    }
}
```

Listing 8. Pseudo-code for `applyRetrofit` function in localized heating retrofit module.

5.4. Case study: the localized heating retrofit module

As a guided process through retrofit module development, we present a case study on the development of the localized heating retrofit module. Localized heating belongs to the HVAC building category, since it functions as an add-on to convective heating systems. Localized heating depends on the occupant positioning information from the occupant profiles, and is considered as an active retrofit. The co-simulator is tasked with determining the additional energy requirement for the electric radiant heaters at each simulation timestep.

In the preprocessing phase, the localized heating retrofit module changes the heating setpoint for all thermostats in the building to the minimum temperature of 10 °C. The pseudo-code for this operation is presented in Listing 8.

In addition, the pseudo-code used to determine the additional radiant heating energy, in the co-simulator, is shown in Listing 9.

```
public double getEnergy(OccupantData occupantData,
   WeatherData weatherData) {
    // Determine total energy consumption by radiant heaters
    for all occupants {
        if occupant is in the building {
            compute required radiant heating energy
        }
    }
    return sum of computed radiant heating energy
}
```

Listing 9. Pseudo-code for `getEnergy` function in localized heating retrofit module.

The retrofit module first determines which occupants are in the building, along with their respective positions. It then computes the required radiant heating energy for each occupant, based on his preferences. Finally, the total energy consumption is returned to the co-simulator for logging purposes.

From this example, we observe that, by exposing the building model components to the user, ROBESim allows automated retrofit application. In addition, retrofits that are not normally compatible with EnergyPlus can still be used under ROBESim. In the localized heating example, EnergyPlus is still used to compute the convective heating energy requirement. The co-simulator computes the part of the retrofit module that is not supported in EnergyPlus.

5.5. Summary

In this section, we discussed the information that the retrofit developer can use when developing retrofit modules in ROBESim. We also described some of the required methods for these modules. The localized heating retrofit module serves as a case study for the future development of retrofit modules.

6. ROBESim components

In Section 3, we provided an outline of the ROBESim simulator workflow. In this section, we discuss, in greater detail, the components of the ROBESim core simulation framework: the simulation manager, preprocessor, co-simulator, and postprocessor.

6.1. Simulation manager

The `SimulationManager` class is used to manage the single-run simulation workflow shown in Fig. 2. It is initialized with a building file, corresponding set of occupant profiles and weather data, energy cost and emissions, and the retrofits that need to be applied, along with their parameters. As previously discussed, the application of the retrofit is performed automatically in ROBESim; a retrofitted building model is generated in the process. In this way, the retrofit developer has control over the retrofit application process, and the user does not have to concern himself with manual retrofit application.

The tasks performed in the simulation manager are as follows:

1. Invoke preprocessor with given inputs.
2. Invoke EnergyPlus with compatible building and weather files.
3. Invoke co-simulator with extended building file, weather file, and occupant profiles.
4. Upon completion of EnergyPlus and co-simulator simulations, invoke postprocessor with interim simulator outputs.
5. Terminate.

The simulation manager serves as a base that additional usage scenarios build upon, as previously discussed in Section 4.

6.2. Preprocessor

The preprocessor first performs a preliminary check of the inputs. This is to ensure the existence of input files and their compliance with the requirements of the simulator. Next, the selected retrofits are applied to a clone of the base building model to yield a retrofitted building model. Finally, the preprocessor reads and parses the input files into objects that can be accessed within the ROBESim framework. The methods used to access these objects are presented in Listing 10.

In Listing 10, the energy cost and emissions information is stored in an `ExtendedInputs` object. This class can be expanded to contain additional information, as required. The preprocessor essentially

```
WeatherData:
    public BuildingModel getBuilding();
    public WeatherDataset getWeatherDatabase();
    public OccupantProfile getOccupantProfile();
    public RetrofitDatabase getRetrofitDatabase();
    // Energy cost and emissions information
    public ExtendedInputs getExtendedInputs();
```

Listing 10. Methods for accessing information in `Preprocessor`.

serves as an input file processor and storage unit for the input information.

6.3. Co-simulator

The co-simulator is used to compute additional energy information not supported in EnergyPlus. This primarily affects active retrofits; since active retrofits rely on additional information, they typically require some computation in the co-simulator. However, not all active retrofits require co-simulator computations. For example, the smart thermostat module uses the occupant profile information to determine the correct thermostat schedules for simulation. This computation is performed in the preprocessing phase, when the `applyRetrofit` method is called. Once the schedules are determined, they can be used in the building model sent to EnergyPlus, since EnergyPlus provides native support for thermostat schedules.

For other active retrofits, such as localized heating and smart lighting, computations are required in the co-simulator. The co-simulator invokes a secondary class, `ReportManager`, to collate and store the output reports for this phase. After all computations are complete, the report manager writes the interim outputs to an interim output file (see Listing 1 in Section 3).

6.4. Postprocessor

The postprocessor reads the interim outputs from both EnergyPlus and the ROBESim co-simulator to produce a combined output file. This is based on an input parameters file that is used to sort the relevant outputs into different categories.

Listing 11 shows a portion of the input parameters file, used to classify electricity, lighting, and HVAC energy consumption. The

```
Electricity,
    Cooling:Electricity,
    Fans:Electricity,
    InteriorLights:Electricity,
    Electricity:Consumption;
Lighting,
    InteriorLights:Electricity;
HVAC,
    Heating,
    Cooling,
    Fans:Electricity;
```

Listing 11. Snapshot of input parameters file used in `Postprocessor`.

postprocessor cross-references the data dictionaries of the interim outputs with the input categories defined in the parameters file to determine the correct categorization. In this example, the `HVAC` category covers dictionary items with the following terms: `Heating`, `Cooling`, and `Fans:Electricity`. Note that dictionary items can belong to several different categories, *e.g.*, the `Fans:Electricity` dictionary item also belongs to the main `Electricity` category. The result of this classification step is a combined output file (see Listing 2 in Section 3) that tracks the energy transfer in the respective categories.

We have presented some key information on the core ROBESim components. Apart from these components, there are a number of helper modules that we have developed to simplify retrofit development and extension of our simulation tool, *e.g.*, input parsers, output tools, and data storage classes. Additional information on these helper modules can be found in the API documentation provided for ROBESim.

7. Input file generators

One of the key goals for our design of ROBESim was accessibility for users of all experience levels. With this in mind, we have developed a number of ease-of-use modules to help improve user experience. These modules primarily assist with building model and occupant profile generation.

7.1. Building model generator

We have developed an automated building model generator capable of quickly generating multiple building models offline. This is in contrast to the EnergyPlus example file generator [51] available online. Using the online tool, users generate building models one at a time; the generated files are then sent to the user through email, usually after a period of time (from approximately fifteen minutes to up to an hour). Conversely, the building model generator in ROBESim can be generated directly on the users' machines; a typical building model can be generated in under one minute (the output building model file is fully compatible with EnergyPlus v8.0). Our building model generator contains several default building system choices, especially for users that are not aware of the systems installed in their building, *e.g.*, the default choice for HVAC system is the `HVACTemplate:Zone:Unitary` template defined in EnergyPlus, referring to the unitary HVAC systems most commonly used in residential and small commercial buildings.

An overview of the different stages in the building model generator is presented in Fig. 8. The key inputs are as follows: building dimensions, window fraction (proportion of walls covered by windows), materials, and climate zone. For building materials, we use the preset constructions provided in EnergyPlus, *i.e.*, `Light` for wooden buildings, `Medium` for brick buildings, and `Heavy` for concrete buildings. We find that these preset constructions provide a good approximation for most buildings of their type. The use of preset constructions presents a more convenient abstraction for the average user.

Based on the inputs above, the building model generator generates the building envelope. This envelope is then divided into different zones. We approximate this process by defaulting to a four-zone per floor scheme. The next stage involves generating the necessary HVAC templates for the building model. The generator supports two templates: `HVACTemplate:Zone:Unitary` and `HVACTemplate:Zone:IdealLoadsAirSystem`. The latter represents an ideal loads air system, serving as a high-level abstraction for the actual HVAC system in the building. The output from the generator comprises the EnergyPlus-compatible building model,

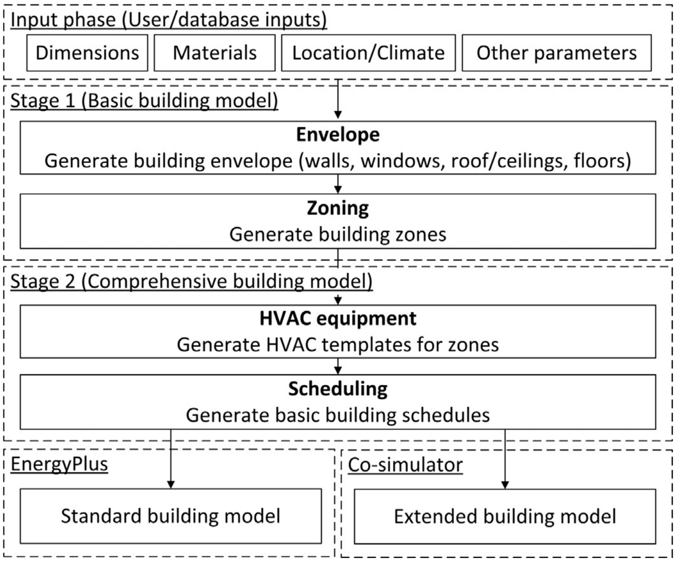

Fig. 8. Building model generator outline.

and the extended building model containing additional information about the appliances in the building.

We provide the source code for our building model generator in ROBESim. It can be extended to include other additional considerations that are not reflected in our generator.

7.2. Occupant profile generator

The occupant profile generator is used to automatically generate occupant profiles for a given building. Fig. 9 provides an overview of the occupant profile generator provided in ROBESim.

We have developed two versions of occupant profile generators: a random profile generator and a schedule-based generator. The inputs to the generator are as follows: the building model (or file), the number of occupants to generate, heuristics (for the random profile generator), and schedules (for the schedule-based generator).

The random profile generator is the easiest way to generate a working occupant profile. It is based on the following heuristics:

- *Hours unoccupied.* The time interval when the occupant is out of the building.
- *Movement probability.* Occupants in a building generally do not move around too much. This probability approximates the movements in real-world situations.

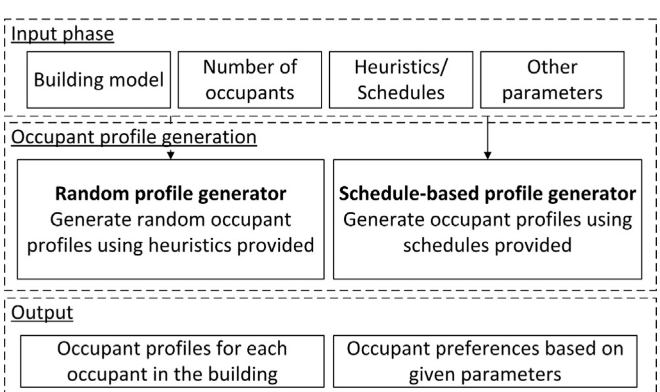

Fig. 9. Occupant profile generator outline.

```
DataDay,
!Day of the week (1=Monday)
1 2 3 4 5,
!Start,End,Activity,Location (Out=not in building)
0000,0900,Sleep,Bed,
0900,0930,Shower,Bath,
0930,1800,Work,Out,
1800,1900,Meal,Dining,
1900,2000,TV,Sofa,
2000,2130,Computer,Desk,
2130,2200,Shower,Bath,
2200,2400,Sleep,Bed;

!Map locations to points
LocationMap,
!Name,X,Y,Z
Bed,1,2,0,
Bath,1,1,0,
Dining,12,2,0,
Sofa,14,2,0,
Desk,2,2,0,
Out,-1,-1,-1;
```

Listing 12. Sample weekday schedule data.

- *Sleeping hours.* For residential buildings, this represents the time interval when the occupant is sleeping and, thus, not moving.

In general, we find that the random profile generator works well, especially considering that it requires no additional effort on the part of the user. Thus, we expect it to be more commonly used.

For users that require a more deterministic occupant profile, we provide a schedule-based profile generator. Since it is not reasonable to expect the user to input a full year of schedules, we require a week of schedules in our generator. These schedule data are then replicated to cover a full calendar year. Sample weekday schedule data are presented in Listing 12.

The schedule data include the following information: occupant schedule for all seven days of the week and map of given locations to actual coordinates. Based on the given schedule, the corresponding occupant profile is then generated.

As in the case of the building model generator, we provide source code for both versions of our occupant profile generator. Our examples serve as reference for future versions of occupant profile generators.

8. Simulation examples

We have discussed the core ROBESim components, along with some of the ease-of-use extensions that we have developed. In this section, we present some simulation examples for ROBESim.

8.1. Building owner usage scenario

In Section 4.1, we covered the one building to many retrofits usage scenario that most building owners, and users of ROBESim, will follow. We performed one simulation run using the wrapper program for the building owner usage scenario. The input file to the wrapper program is presented in Listing 13.

```
Building,
    Example-4Z-108-1-Res-Light;

Weather,
    USA_NJ_Trenton-Mercer.County.AP.724095_TMY3.epw;

Occupant,
    occ_example_basic;

Retrofit,
    localized_heating,
    programmable_thermostat,
    smart_thermostat,
    triple_pane_windows;
```

Listing 13. Input to wrapper program for building owner usage scenario.

The input file is parsed in the wrapper program to determine the input building file, weather file, occupant profiles, and the retrofits to consider. The input building file is a single-story 108 m^2 residential apartment with light construction, *i.e.*, built of wood. The input weather file, in this case, is for Trenton, NJ. The occupant profiles used are for three regular workers, *i.e.*, they leave for work on weekdays. These profiles were generated using the schedule-based generator described previously. In this example, we consider the following retrofits: localized heating, programmable thermostat, smart thermostat, and triple-pane windows. We seek to determine the impact on HVAC energy consumption of implementing each of these three retrofits.

Fig. 10 illustrates the simulation results. The energy savings results for the smart thermostat and for localized heating are in line with the results presented in [13–16] and [45], respectively. The savings results for the programmable thermostat and triple-pane windows are in line with our manual simulation results; in our validation test, we created retrofitted building models manually for both retrofits to obtain these simulation results.

8.2. Parametric analysis example

In addition to performing comparisons between different retrofits, ROBESim can also be used to perform parametric analysis on a specified retrofit. In this case, the base retrofit remains the same; variants of this base retrofit are generated from user-specified parameters. These retrofit variants are then automatically applied to the building model for simulation.

As an example of parametric analysis using ROBESim, we evaluated the impact of wall insulation thickness on HVAC energy

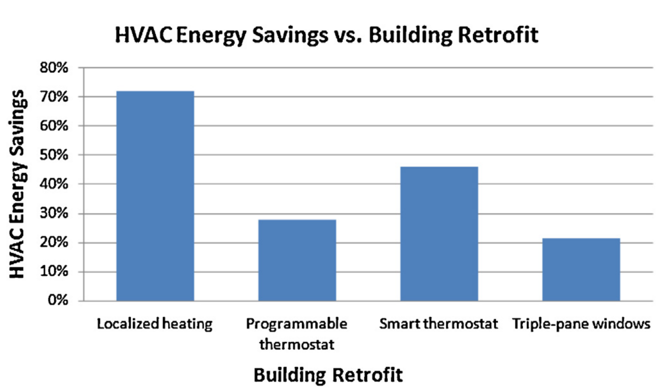

Fig. 10. HVAC energy savings for each retrofit.

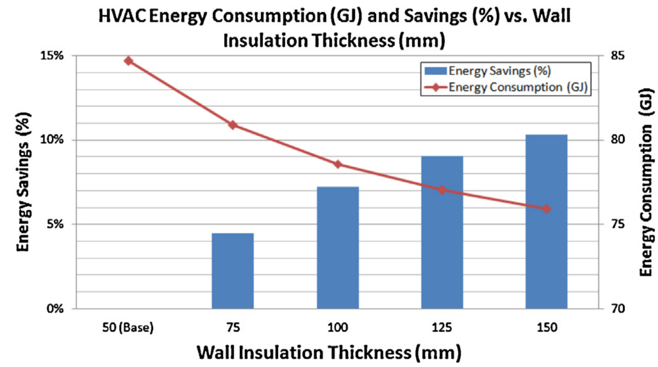

Fig. 11. HVAC energy consumption and savings across different wall insulation thicknesses.

consumption. In this analysis, we used the same building model and climate zone (single-story 108 m^2 residential apartment with light construction in Trenton, NJ). In our simulations, we used the following wall insulation thicknesses: 50 mm, 75 mm, 100 mm, 125 mm, and 150 mm. The wall insulation thickness in the original building model was 50 mm. This was used as our base case for energy savings comparisons. The results of this analysis are illustrated in Fig. 11.

From Fig. 11, we observe that the energy savings increase as additional insulation is added to the walls. A wall insulation thickness of 150 mm enables HVAC energy savings of around 10%, compared to the base case (50 mm). In this example, ROBESim automatically applies the new wall constructions to the base building model to create new building models with the appropriate wall insulation thicknesses. In addition, the output from ROBESim allows quick and easy comparison between the different parameters. Thus, the effort required from the user is significantly reduced.

8.3. Retrofit developer usage scenario

In this example, we explore the retrofit developer's perspective: a one retrofit to many buildings usage scenario. For this example, the wrapper program input is as shown in Listing 14.

The input instructs the wrapper program to perform several ROBESim simulations with the triple-pane window retrofit, using the same building model, across different climate zones. The

```
Building,
    Example-4Z-108-1-Res-Light;

Weather,
    USA_NJ_Trenton-Mercer.County.AP.724095_TMY3.epw,
    USA_IL_Chicago-OHare.Intl.AP.725300_TMY3.epw,
    USA_CO_Boulder-Broomfield-Jefferson.County.AP.724699_TMY3
       .epw,
    USA_AZ_Phoenix-Sky.Harbor.Intl.AP.722780_TMY3.epw,


    USA_WA_Seattle-Tacoma.Intl.AP.727930_TMY3.epw,
    USA_GA_Atlanta-Hartsfield-Jackson.Intl.AP.722190_TMY3.epw
       ;

Occupant,
    occ_example_basic;

Retrofit,
    triple_pane_windows;
```

Listing 14. Input to wrapper program for retrofit developer usage scenario.

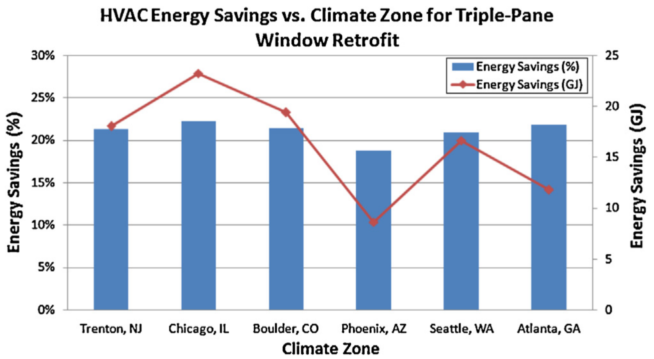

Fig. 12. HVAC energy savings for triple-pane window retrofit across different climate zones.

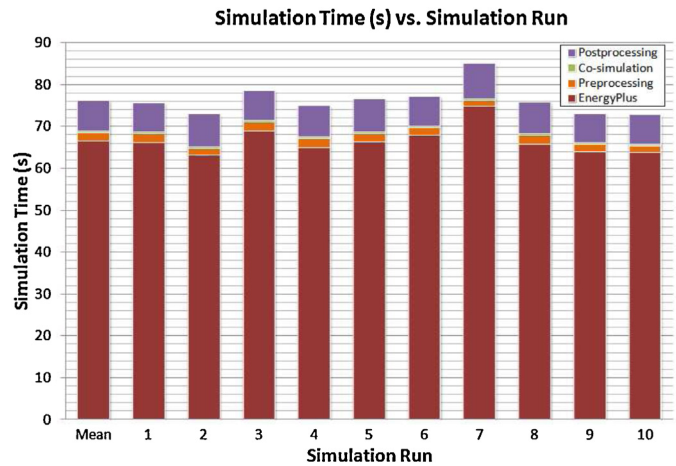

Fig. 13. Simulation time for each simulation run, according to simulation phase.

climate zones selected are as follows: Trenton (New Jersey), Chicago (Illinois), Boulder (Colorado), Phoenix (Arizona), Seattle (Washington), and Atlanta (Georgia). The simulation results are illustrated in Fig. 12. From the figure, we observe that the HVAC energy savings, as a percentage, are approximately the same for the six climate zones (around 20%), although the energy savings, when represented in gigajoules (*GJ*), vary significantly. This is primarily due to the varying HVAC requirement across the different climate zones. Thus, although the proportion of HVAC energy savings is similar in Chicago and Phoenix (22% vs. 19%), the total energy and cost savings will be significantly higher in Chicago than in Phoenix (23.2 *GJ* vs. 8.6 *GJ*).

In the two examples above, we described the structure of the wrapper program inputs. Upon receiving these inputs, the rest of the wrapper programs is automated, *i.e.*, the programs do not require any user intervention. The output from the simulations is a comma-separated values (CSV) file that can be easily processed to produce the figures shown in Figs. 10 and 12. Thus, using the wrapper programs included in ROBESim, the user can quickly and easily perform simulations for the different usage scenarios.

8.4. Simulation overhead analysis

In this section, we discuss the simulation overhead attributed to ROBESim. In our analysis, we performed ten identical ROBESim simulation runs, using the 108 m^2 apartment building model in Trenton, NJ. We used localized heating as an example, since localized heating requires additional computations in the co-simulator to determine the energy required in radiant heating (see Section 5.4).

Fig. 13 illustrates the results of this analysis. From the figure, we observe that EnergyPlus represents the bulk of the simulation run time in our example. On average, the additional simulation overhead from ROBESim (preprocessing, co-simulation, and postprocessing) is around 14.5%, compared to running EnergyPlus alone.

In our example, the computations in the co-simulator, using the localized heating retrofit module, do not require much additional time (approximately 0.4 seconds). This is not true for all retrofit modules, since some retrofit modules depend on control systems using complex computations (*e.g.*, smart lighting requires more co-simulation time due to its dependency on genetic algorithms for control), whereas other retrofit modules, such as simple wall or window retrofits, do not require any computations in the co-simulator. Thus, the actual simulation overhead depends on the retrofit modules that are applied.

9. Conclusion

In this paper, we have motivated the need for more information when making decisions to retrofit buildings. In addition, we have highlighted building energy simulation as a quick and inexpensive means to obtain this information. We anticipate that, as more retrofits are developed and made commercially available, retrofitting old buildings will be an increasingly popular solution to reduce energy consumption. Existing simulation tools, however, are too complex for most building owners to operate. Our solution, ROBESim, is a retrofit-oriented simulation tool that supports built-in functions for retrofit modules. This enables a host of user-friendly add-ons that simplify the building energy simulation process. We are confident that ROBESim will be easy to use and useful to both building owners and retrofit developers of all experience levels.

In ROBESim, we have made a deliberate decision in our design process for our simulation framework to be modular and extensible. We have also made the relevant libraries, source code, and API available. We hope that ROBESim can serve as a base for additional modules and interface enhancements to improve the building energy simulation experience.

Acknowledgment

This work was supported by the Agency for Science, Technology and Research (A*STAR), Singapore, under the National Science Scholarship.

References

[1] Leonardo Energy, World Energy Expenditures, 2013 http://www.leonardo-energy.org/world-energy-expenditures
[2] IEA, International Energy Agency, 2013 http://www.iea.org/
[3] EIAIEO, U.S. Energy Information Administration (EIA): International Energy Outlook, 2011 http://www.eia.gov/forecasts/ieo/
[4] BEDB, U.S. Department of Energy Office of Energy Efficiency and Renewable Energy (EERE): Buildings Energy Data Book, 2011 http://buildingsdatabook.eren.doe.gov/
[5] EIA, U.S. Energy Information Administration, 2013 http://www.eia.doe.gov
[6] EIACBECS, U.S. Energy Information Administration (EIA): Commercial Buildings Energy Consumption Survey, 2003 http://www.eia.doe.gov/emeu/cbecs/
[7] Nest, Learning Thermostat, 2013 http://www.nest.com/
[8] EnergyPlus, Energy Simulation Software, 2013 http://apps1.eere.energy.gov/buildings/energyplus/
[9] D.B. Crawley, L.K. Lawrie, F.C. Winkelmann, W. Buhl, Y.J. Huang, C.O. Pedersen, R.K. Strand, R.J. Liesen, D.E. Fisher, M.J. Witte, J. Glazer, EnergyPlus: Creating a new-generation building energy simulation program, Energy and Buildings 33 (2001) 319–331.
[10] eQuest, The Quick Energy Simulation Tool, 2013 http://doe2.com/equest/
[11] DOE-2, Building Energy Use and Cost Analysis Tool, 2013 http://doe2.com/DOE2/

[12] TRNSYS, Transient System Simulation Tool, 2013 http://www.trnsys.com/
[13] J. Lu, T. Sookoor, V. Srinivasan, G. Gao, B. Holben, J. Stankovic, E. Field, K. Whitehouse, The smart thermostat: Using occupancy sensors to save energy in homes, in: Proc. ACM Conf. Embedded Networked Sensor Systems, 2010, pp. 211–224.
[14] J. Scott, A.J.B. Brush, J. Krumm, B. Meyers, M. Hazas, S. Hodges, N. Villar, PreHeat: Controlling home heating using occupancy prediction, in: Proc. ACM Int. Conf. Ubiquitous Computing, 2011, pp. 281–290.
[15] Y. Agarwal, B. Balaji, S. Dutta, R.K. Gupta, T. Weng, Duty-cycling buildings aggressively: the next frontier in HVAC control, in: Proc. Int. Conf. Information Processing in Sensor Networks, 2011, pp. 246–257.
[16] V.L. Erickson, M.A. Carreira-Perpinan, A.E. Cerpa, Observe Occupancy-based system for efficient reduction of HVAC energy, in: Proc. Int. Conf. Information Processing in Sensor Networks, 2011, pp. 258–269.
[17] V. Singhvi, A. Krause, C. Guestrin, J.H. Garrett Jr., H.S. Matthews, Intelligent light control using sensor networks, in: Proc. Int. Conf. Embedded Networked Sensor Systems, 2005, pp. 218–229.
[18] Y.-J. Wen, A.M. Agogino, Personalized dynamic design of networked lighting for energy-efficiency in open-plan offices, Energy and Buildings 43 (2011) 1919–1924.
[19] D.T. Delaney, G.M.P. O'Hare, A.G. Ruzzelli, Evaluation of energy-efficiency in lighting systems using sensor networks, in: Proc. ACM Wkshp. Embedded Sensing Systems for Energy-Efficiency in Buildings, 2009, pp. 61–66.
[20] TED, The Energy Detective, 2013 http://www.theenergydetective.com/
[21] A.G. Ruzzelli, C. Nicolas, A. Schoofs, G.M.P. O'Hare, Real-time recognition and profiling of appliances through a single electricity sensor, in: Proc. IEEE Conf. Sensor Mesh and Ad Hoc Communications and Networks, 2010, pp. 1–9.
[22] S. Gupta, M.S. Reynolds, S.N. Patel, Electrisense: Single-point sensing using EMI for electrical event detection and classification in the home, in: Proc. ACM Int. Conf. Ubiquitous Computing, 2010, pp. 139–148.
[23] Y. Kim, T. Schmid, Z.M. Charbiwala, M.B. Srivastava, Viridiscope: Design and implementation of a fine grained power monitoring system for homes, in: Proc. ACM Int. Conf. Ubiquitous Computing, 2009, pp. 245–254.
[24] D. Jung, A. Savvides, Estimating building consumption breakdowns using on/off state sensing and incremental sub-meter deployment, in: Proc. ACM Conf. Embedded Networked Sensor Systems, 2010, pp. 225–238.
[25] X. Jiang, M. Van Ly, J. Taneja, P. Dutta, D. Culler, Experiences with a high-fidelity wireless building energy auditing network, in: Proc. ACM Conf. Embedded Networked Sensor Systems, 2009, pp. 113–126.
[26] T. Schmid, D. Culler, P. Dutta, Meter any wire, anywhere by virtualizing the voltage channel, in: Proc. ACM Wkshp. Embedded Sensing Systems for Energy-Efficiency in Buildings, 2010, pp. 25–30.
[27] Z.C. Taysi, M.A. Guvensan, T. Melodia, Tinyears: Spying on house appliances with audio sensor nodes, in: Proc. ACM Wkshp. Embedded Sensing Systems for Energy-Efficiency in Buildings, 2010, pp. 31–36.
[28] A. Rowe, M. Berges, R. Rajkumar, Contactless sensing of appliance state transitions through variations in electromagnetic fields, in: Proc. ACM Wkshp. Embedded Sensing Systems for Energy-Efficiency in Buildings, 2010, pp. 19–24.
[29] EnergyStar, Home Energy Audits, 2013 http://www.energystar.gov/index.cfm?c=home_improvement.hm_improvement_audits
[30] SCIenergy, Official Website, 2013 http://www.scienergy.com/
[31] Retroficiency, Official Website, 2013 http://www.retroficiency.com/
[32] S. Schiavon, K.H. Lee, F. Bauman, T. Webster, Simplified calculation method for design cooling loads in underfloor air distribution (UFAD) systems, Energy and Buildings 43 (2011) 517–528.
[33] J. Wei, J. Zhao, Q. Chen, Energy performance of a dual airflow window under different climates, Energy and Buildings 42 (2010) 111–122.
[34] K.H. Lee, R.K. Strand, Enhancement of natural ventilation in buildings using a thermal chimney, Energy and Buildings 41 (2009) 615–621.
[35] Z. Tian, J.A. Love, Energy performance optimization of radiant slab cooling using building simulation and field measurements, Energy and Buildings 41 (2009) 320–330.
[36] N. Fumo, P. Mago, R. Luck, Methodology to estimate building energy consumption using EnergyPlus benchmark models, Energy and Buildings 42 (2010) 2331–2337.
[37] EnergyPlus, Third-Party Software Products, 2013 http://apps1.eere.energy.gov/buildings/energyplus/other_tools.cfm
[38] C.E. Ochoa, I.G. Capeluto, Advice tool for early design stages of intelligent facades based on energy and visual comfort approach, Energy and Buildings 41 (2009) 480–488.
[39] OpenStudio, National Renewable Energy Laboratory, 2013 http://openstudio.nrel.gov/
[40] T.X. Nghiem, MLE+: A Matlab-EnergyPlus Co-simulation Interface, 2013 http://www.seas.upenn.edu/nghiem/mleplus.html
[41] xEsoView, eso file viewer, 2013 http://sourceforge.net/projects/xesoview/
[42] BEopt, Building Energy Optimization Software, 2013 http://beopt.nrel.gov/home
[43] Radiance, A Validated Lighting Simulation Tool, 2013 http://www.radiance-online.org/
[44] MATLAB, The Language of Technical Computing, 2013 http://www.mathworks.com/products/matlab/
[45] J.W. Chuah, C. Li, A. Raghunathan, N.K. Jha, Localized heating for building energy efficiency, in: Proc. Int. Conf. VLSI Design, 2013, pp. 13–18.
[46] S. Basuroy, J.W. Chuah, N.K. Jha, Making buildings energy-efficient through retrofits: A survey of available technologies, in: Proc. IEEE Power and Energy Society General Meeting, 2013.
[47] J. Page, D. Robinson, N. Morel, J.-L. Scartezzini, A generalised stochastic model for the simulation of occupant presence, Energy and Buildings 40 (2008) 83–98.
[48] I. Richardson, M. Thomson, D. Infield, A high-resolution domestic building occupancy model for energy demand simulations, Energy and Buildings 40 (2008) 1560–1566.
[49] P. Hoes, J. Hensen, M. Loomans, B. de Vries, D. Bourgeois, User behavior in whole building simulation, Energy and Buildings 41 (2009) 295–302.
[50] EnergyPlus, Weather Data, 2013 http://apps1.eere.energy.gov/buildings/energyplus/weatherdata_about.cfm
[51] EnergyPlus, Example File Generator, 2013 http://apps1.eere.energy.gov/buildings/energyplus/file_generator_about.cfm
[52] EIARECS, U.S. Energy Information Administration (EIA): Residential Energy Consumption Survey, 2009 http://www.eia.doe.gov/emeu/recs/contents.html