sustainability

MDPI

Article

EplusLauncher: An API to Perform Complex EnergyPlus Simulations in MATLAB® and C#

Germán Campos Gordillo [1], Germán Ramos Ruiz [2,*], Yves Stauffer [3], Stephan Dasen [3] and Carlos Fernández Bandera [2]

1 Aurea Consulting, Sustainable Architecture and Engineering, 20110 Pasaia, Spain; gc@ecoeficiente.es
2 School of Architecture, University of Navarra, 31009 Pamplona, Spain; cfbandera@unav.es
3 Centre Suisse d'Electronique et de Microtechnique, 2002 Neuchâtel, Switzerland; Yves.STAUFFER@csem.ch (Y.S.); Stephan.DASEN@csem.ch (S.D.)
* Correspondence: gramrui@unav.com; Tel.: +34-948-425-600 (ext. 802751)

Received: 4 November 2019; Accepted: 10 January 2020; Published: 16 January 2020

Abstract: There is a growing concern about how to mitigate climate change, in which the production and use of energy has a great impact as one of the largest sources of global greenhouse gases (GHG). Buildings are responsible for a large percentage of these emissions. Therefore, there has been an increase in research in this area, in order to reduce their consumption and increase their efficiency. One of the major simulation programs used in optimization research is EnergyPlus. The purpose of this software is the complete energy simulation of a building, although it lacks tools to analyze its results and, above all, to manage and edit its simulations. For this reason, we developed an application programming interface (API) that serves to merge two areas which are highly demanded by researchers: energy building simulation (using EnergyPlus) and tools for the management and design of research experiments (in this case, MATLAB®). The developed API allows the user to perform complex simulations using EnergyPlus in a simple way, as it allows the editing of each simulation and the analysis of the simulation results through MATLAB®. In addition, it enables the user to simultaneously run multiple simulations, using either all computer core processors or a selection of them (i.e., allowing parallel computing), reducing the simulation time. The API was developed in the C# language, such that it can be used with any software that can import *.NET* libraries.

Keywords: energyPlus; MATLAB®; C#; building energy simulation; parameterization; parallelization

1. Introduction

In February 2019, the Pew Research Center published a report on the top global threats [1], in which it stated that concern about global climate change has climbed, such that it is now at the top of the eight international threats, along with concerns about ISIS, cyberattacks, North Korea's nuclear program, and others. This research center "*is a nonpartisan fact tank that informs the public about the issues, attitudes and trends shaping America and the world*" [2]. The report also highlighted that "*people around the world agree that climate change poses a severe risk to their countries*", as well as the concern of the Intergovernmental Panel on Climate Change (IPCC) about its possible impacts in both the near and distant future.

As is well-known, buildings and construction play an important role in mitigating global warming, as they are large consumers of final energy (36%) and, therefore, are large producers of CO_2 emissions (39%) [3]. This is one of the main reasons there have been many studies on how to improve the building energy performance, how to optimize their HVAC systems, how to obtain cost-optimal retrofit solutions, and so on. In 2014, Nguyen et al. [4] reviewed the simulation-based optimization

methods currently used to analyze building energy performance. In the review, they stated that EnergyPlus [5] and TRNSYS (Transient System Simulation Tool) are the main simulation tools used in the field of building optimization research, with values of 37.2% and 35.3%, respectively. TRNSYS, as explained in [6], has "*a modular structure that implements a component-based approach*". These components may be as simple as a weather file or part of an HVAC system, or as complex as an entire building. The software calculates and solves different equations representing the building energy model. In contrast, EnergyPlus is an entire building energy simulation software, which serves to calculate energy consumption taking into account many aspects that thermally influence the building. It has a lot of features and capabilities, such as different heat balance algorithms, advanced fenestration models, a large number of HVAC possibilities, Functional Mockup Interfaces (FMI) that enable co-simulation with other software, and so on.

In terms of optimization software, Attia et al. [7] performed a literature review by reviewing of 165 publications and 28 interviews with optimization experts about the trends in simulation-based building performance optimization (BPO) tools. In the review, they found that GenOpt [8] and MATLAB® were the most used tools, and MATLAB® the most popular tool. This is because MATLAB® has a optimization toolbox which provides a large set of algorithms to minimize or maximize objectives taking into account different constraints, including solvers for linear and non-linear programming, with continuous and discrete variables, and so on. In addition, it features tools which enable the user to analyze the optimization results. As Yan et al. [9] highlighted in their conclusions: "*although there have been many software tools developed specially for building optimization design, Matlab is still the most popular tool for optimal solution searches*".

Although coupling TRNSYS and MATLAB® have also been commonly used to simulate and optimize energy buildings [10,11], we focus on coupling EnergyPlus and MATLAB® in this article. Within this group, almost all studies can be organized into two large groups: those that used co-simulation to perform optimization and those that used MATLAB® to launch EnergyPlus simulations and analyze their results.

- Co-simulation between EnergyPlus and MATLAB®

  Co-simulation is a simulation strategy which allows different simulation programs to run simultaneously and exchange data during the simulation. Each software package expects data from the other, which allows the programs to complement missing features in the other or to optimize the building operation with information obtained during the simulation. EnergyPlus uses the Building Controls Virtual Test Bed (BCVTB) as the software environment that enables the user to couple different elements of EnergyPlus for co-simulation with other software (e.g., MATLAB®, Modelica, Radiance, TRNSYS, ESP-r, and so on). It is based on the Ptolemy II software environment and has the ability to couple the simulation program with the actual hardware. This communication protocol is open and can be used by other software to perform co-simulation with EnergyPlus; this is the case of MLE+ [12]. The latter is executed in the standard environment of MATLAB®, which facilitates the finding and fixing of errors (debugging). This is more complicated to do with the direct implementation of BCVTB in MATLAB®, which requires knowledge of the Ptolemy II programming language. All of the examples that use co-simulation between EnergyPlus and MATLAB® use one of these (i.e., BCVTB or MLE+).

  For the cases that use BCVTB as the software environment to perform the co-simulation between EnergyPlus and MATLAB®, the optimizations carried out are focused on Model Predictive Control (MPC) optimizations [13–18]; on building energy consumption improvements [19,20]; on thermal comfort using the Human and Building Interaction Toolkit [21]; on considering the condensation risk of a thermally activated building (TAB) in the cooling operation [22]; on how to integrate the ability to simulate double-skin facades into EnergyPlus [23]; on thermal bridges using MATLAB® [24]; and so on.

For the cases that use MLE+, the optimizations carried out are similar: Model Predictive Control (MPC) optimizations [25–28]; building energy performance improvements using phase change materials [29]; analyzing daylight conditions with Radiance [30]; or making decisions based on the thermal comfort levels [31]. It has even been used to create a black-box model with the information provided by EnergyPlus during co-simulation [32].

- Management of EnergyPlus simulations using MATLAB®

  On the other hand, other studies on optimizing or analyzing the energy performance of a building did not require the use of co-simulation, but focused on managing the simulations and their results. In these cases, it is necessary to develop a personal script that manages EnergyPlus through MATLAB®, as there exists no dedicated application that does this in a simple way. The research topics that use EnergyPlus in this way are broader: for example, optimizing building energy retrofits using genetic algorithms [33–38] or a specific methodology, such as the "Simulation-based Large-scale uncertainty/sensitivity Analysis of Building Energy performance" (SLABE) method [39]; improving the building energy performance when designing the envelope [40]; performing thermostatic optimization [41]; or using Agent-Based Modeling (ABM) to capture the dynamic energy of occupants [42]. In addition, more specific topics have been studied using this strategy, such as: optimizing the control of blind systems [43]; using urban energy models to analyze the urban heat island effect [44]; developing an EnergyPlus meta-model [45]; performing MPC and thermal comfort optimization [46]; developing a sensitivity analysis using the Simlab software [47,48]; optimizing the renewable energy mix of a building [49]; finding an optimal architectural design using BIM at the design stage [50]; or performing automated random parametric simulations [51].

  To make things even more complicated, researchers sometimes prefer to develop a personal script to manage other software, which, in turn, manages EnergyPlus, for example using GenOpt to analyze the energy saving potential of adaptive glazing technologies [52] or the optimal control of a switchable glazing façade [53]; using jEPlus [54] to perform sensitivity analyses, uncertainty quantification, or multi-objective optimizations [55–59]; or, with Rhinoceros and Grasshopper, using Ladybug and Honeybee to evaluate energy saving strategies in commercial buildings with integrated data collection [60].

After analyzing all of these studies, the need for an application that links an optimization tool (e.g., MATLAB®) with a building energy simulation program (e.g., EnergyPlus) is evident, as highlighted by Attia et al. [7], "*for less simulation efforts and feasible optimization it is essential to develop the link between existent building simulation tools and trusted optimization tools*". In particular, the necessity of such integration with MATLAB® was highlighted by Leow et al. [61]: "*There exist other high-performance building simulation applications e.g., TrnSys, ESP-r, EnergyPlus but these offer a less direct integration path with popular mathematical computing and simulation environments e.g., MATLAB*". The authors of Thieblemont et al. [62] also highlighted that, without this application, the possibilities for the optimization of building energy models are reduced "*(...) at the moment, the building simulation community mainly uses software having no optimization process like TRNSYS or EnergyPlus does. In consequence, building model software tools must be linked to some optimization tools (Matlab, GenOpt, etc.), which may be difficult to achieve or reduce the modeling possibilities*".

This article explains the development of this application, in API format, such that its use by other types of software is both simple and versatile. Figure 1 shows the general scheme of simulation in EnergyPlus. As can be seen, the general scheme of an EnergyPlus simulation only requires two files: a weather file ($*.epw$) and a building energy model (BEM) file ($*.idf$). Then, after the simulation, the software generates different outputs (i.e., a $*.csv$ file and an $*.html$ file). Figure 2 shows the same process, but performed with the EplusLauncher API. In this case, the EnergyPlus input files are managed by the API, which is responsible for running the simulations. Then, when the process is

finished, MATLAB® can analyze the output files, making it possible to make new decisions about the input files.

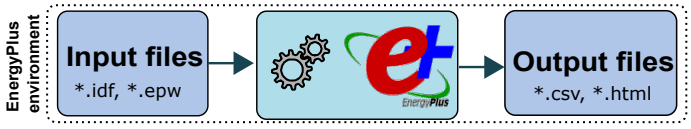

Figure 1. General EnergyPlus simulation scheme.

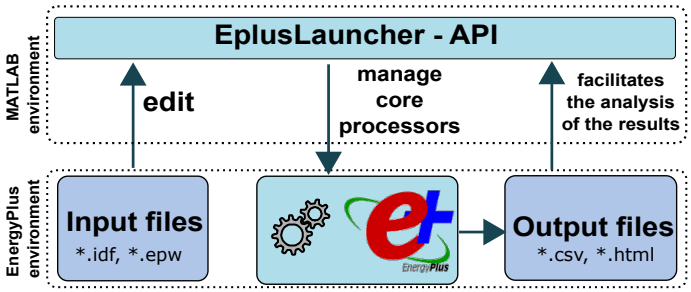

Figure 2. Management of EnergyPlus simulations using the EplusLauncher API.

The advantages of EplusLauncher API can be summarized in three main aspects: (i) it allows the editing of the EnergyPlus input files before the simulation process begins; (ii) it manages the EnergyPlus simulation, allowing the use of a selection of computer processors (parallelization); and (iii) it facilitates the analysis of results through MATLAB®, as all of the operations are performed within the same platform. In the following section, each of these advantages are explained in depth.

The document is structured as follows. Section 2 describes how to use the API and the possibilities of all its options. Section 3 gives an example of its use in a real case scenario. This case was developed for a European project of the program Horizon 2020 under Grant No. 731211, whose name is "project SABINA" (https://sabina-project.eu) [63] (SmArt BI-directional multi eNergy gAteway). Finally, the conclusions are summarized in Section 4.

2. Description of the API

The EplusLauncher API can be downloaded from the SABINA project repository (https://sabina-project.eu/resources/API/index.html), and the documentation can be consulted at this link (https://sabina-project.eu/resources/API/api/index.html).

To explain the use and features of the API, we use a simple parameterization example performed with EnergyPlus and managed by MATLAB® through the EplusLauncher API. As explained above, the API was developed in C#, thus it can be used by any other software that can import *.NET* libraries, as is the case for MATLAB®. The structure of this example could be used easily in other studies, adapting the code to the needs of the particular experiment. It is organized into four MATLAB® files: "`launcher.m`", "`EnergyPlusMessage.m`", "`SimulationFinished.m`", and "`JobsFinished.m`".

- `launcher.m`
  This is the main file, where almost all of the options must be defined (see Listing 1). First, the paths of the files, including the path of the EplusLauncher API, are defined [`API Loading`]. Next, it is necessary to configure the API [`Launcher configuration`]. In this part, it is possible to select the number of cores to perform the simulation (`Launcher.MaxCores`), where "0" is the maximum. This function is useful as, in some studies, many simulations need to be performed, such as parameterization studies or in the analysis of models that use optimization algorithms, where a group of simulations must be performed before making a decision about the model (for example, a genetic algorithm and its population simulation). For these cases, having the ability to perform a group of simulations at the same time (i.e., parallelization of the simulations)

can make these studies feasible. In addition, when the number of cores is high, it is possible to perform several optimizations of different problems at the same time by defining specific cores for each optimization.

Each of these simulations will be stored inside a folder called (`OutputJob#`), where # is the number defined by (`Launcher.OutputFolderNumber`). In the cases where different optimizations are carried out at the same time, it is good practice to identify each of them. Initially, this folder name may seem too simple; however, it satisfies the Windows® file and path length limitations (due to the maximum character number limitation). The disadvantage is that the file does not give us information about the type of simulation that has been carried out. This Windows® limitation seemed more important to us than the information that could be given in the folder name. However, this information is collected in the log file, which is created by the command (`Launcher.JobsLogFile`) and can be accessed to obtain the details of each of the simulations performed by EplusLauncher. As shown below, the folder name (`OutputJob#`) is defined in the [`EnergyPlus configuration`] section of `launcher.m`, whereas the number (#) is defined in the [`Launcher configuration`] section. The reason for this is to avoid cases where it is necessary to perform different simulations with different configurations (e.g., several jobs with different weather or input files), in which the results will be overwritten because they are assigned the same folder name. Thus, each folder is identified by a unique name and number, preventing errors.

Listing 1: `launcher.m` file.

```
%% =============== API Loading ========================================
folder = pwd; % Path of the files
NET.addAssembly(fullfile(folder,'EplusLauncher.dll'));
%% =============== Launcher configuration ==============================
global Launcher;
Launcher = EplusLauncher.Launcher();
Launcher.MaxCores = 0;
Launcher.OutputFolderNumber = 0;
Launcher.JobsLogFile = fullfile(folder,'jobs.csv');
%% =============== EnergyPlus listeners ================================
addlistener(Launcher,'OutputMessageReceived',@EnergyPlusMessage);
addlistener(Launcher,'SimulationCompleted',@SimulationFinished);
addlistener(Launcher,'JobsFinished',@JobsFinished);
%% =============== EnergyPlus configuration ============================
config = EplusLauncher.Configuration();
config.EnergyPlusFolder = 'C:\EnergyPlusV9-1-0\';
config.InputFile = fullfile(folder,'in.idf');
config.WeatherFile = fullfile(folder,'in.epw');
config.RviFile = fullfile(folder,'rviFile.rvi');
config.OutputFolder = fullfile(folder,'OutputJob');
%% =============== Building Energy Model (BEM) edition - Example =======
Tags = {'#Mat_Roughness#','#Thick_Insulation#','#Mat_Density#'};
Material_Roughness = {'MediumSmooth','Smooth','Rough'};
Thickness_Insulation = [0.02,0.04,0.10];
Material_Density = [1000, 1200, 1400];
%% =============== List of jobs ========================================
for mat_roughness = Material_Roughness
  for thick_insulation = Thickness_Insulation
    for mat_density = Material_Density
      job = EplusLauncher.Job(config);
      job.AddTags(Tags);
            job.AddValues([mat_roughness,thick_insulation,mat_density]);
            Launcher.Jobs.Add(job);
    end
  end
end
%% =============== Run simulations =====================================
Launcher.Run();
```

Next, [`EnergyPlus listeners`] are defined. These listeners report each EnergyPlus message to MATLAB®, as well as when a simulation ends or all of the jobs end. They make calls to the other MATLAB® files (`EnergyPlusMessage.m`, `SimulationFinished.m`, and `JobsFinished.m`) and allow for adding in new code (scripts) for other functionalities that have not yet been developed in the API (e.g., reporting the simulation time or the ongoing results, modify the simulation file during the optimization process using the results, change the optimization strategy if a local minimum has been reached, etc.).

Then, it is necessary to perform the [`EnergyPlus configuration`]. In this part, all the files needed to run EnergyPlus are defined: the installation folder of EnergyPlus (`config.EnergyPlusFolder`); the building energy model file (∗*.idf*)(`config.InputFile`); the weather file (∗*.epw*)(`config.WeatherFile`); the RVI file (∗*.rvi*), which is the responsible for reorganizing the EnergyPlus results in the output files (∗*.csv*)(`config.RviFile`); and the name of the output folder (`config.OutputFolder`). This name will be followed by the number defined in the (`Launcher.JobsLogFile`) stage.

Finally, editing of the Building Energy Model (BEM) file and definition of the simulation list is done. EplusLauncher can edit the simulation file (∗*.idf*) before each simulation. As ∗*.idf* files are text-based files, EplusLauncher searches for tags (i.e., identifiers) in the file and replaces them with other variables. These variables can be text (strings) or values (numbers). These identifiers are defined in the variable (`Tags`) and their values are variables created by the user. The example given below is the parameterization of a building envelope, where the roughness and density of one of its layers and the insulation thickness are parameterized. In this case, the job list is defined by several "`for`" loops, in order to select all the variation possibilities of the three parameters. This list must be defined for each analyzed problem, and will be different depending on the type of problem. For example, in the case of optimization using a genetic algorithm, this list of jobs will be the population of each generation, where the different identifiers will be first added to the list of jobs (`job.AddTags(Tags)`) and, then, the values of these identifiers (`job.AddValues([...])`).

The file (`launcher.m`) ends with the execution command (`Launcher.Run()`), which, in the example, executes 27 simulations ($3 \cdot 3 \cdot 3$) giving all possibilities in the parameterization example (i.e., a brute force technique).

- `EnergyPlusMessage.m`

  This file displays all EnergyPlus messages in the MATLAB® command window. It is very helpful, as it reports all possible errors during the simulation process, identifying which simulation has caused the error (see Listing 2).

Listing 2: `EnergyPlusMessage.m` file.

```
function EnergyPlusMessage(~,args)
disp([char(args.SimulationFolder) ': ' char(args.Message)]);
end
```

- `SimulationFinished.m`

  This file summarizes the entire simulation process of each simulation, reporting "`correct simulation: True`" if the simulation has run without any errors (see Listing 3).

Listing 3: `SimulationFinished.m` file.

```
function SimulationFinished (~,args)
Result = 'True';
if args.Success == 0
Result = 'False';
end
disp([char(args.SimulationFolder) ' - correct simulation: ' Result])
end
```

- `JobsFinished.m`

  Finally, this file is executed when all simulations are finished (see Listing 4). In an optimization process, this file is used to add additional operations to extract valuable information from the results obtained, which can be used to edit the EnergyPlus files and repeat the process.

  In general, simulations are carried out on a large scale in all optimization processes; thus, it is necessary to manage the simulation files. EnergyPlus generates an enormous amount of information throughout the simulation process. Many of these files are not used in the study being carried out and only take up space. For this reason, the API has an option to delete files, which is executed at the end of the simulation process. In the example, this function is useful, as the only files that are used after the parameterization are the results files. This removal process is accessed by defining the filenames with their extensions (`cleaner.SetFiles()`) or by selecting files with the same extension (`cleaner.SetFilesExtensions()`). Once the files have been defined, the cleaning process is executed in all the folders created during the simulation process using the command (`cleaner.CleanFromLogFile()`).

Listing 4: `JobsFinished.m` file.

```
function JobsFinished(~,~)
global Launcher;
cleaner = EplusLauncher.Cleaner();
% Files to delete by name
cleaner.SetFiles({'eplusout.bnd','eplusout.eso'});
% Files to delete by extension
cleaner.SetFilesExtensions({'mtr','eso','mtd','sql'});
cleaner.CleanFromLogFile(Launcher.JobsLogFile);
disp('All done. :)');
end
```

The use of this API facilitates the management of EnergyPlus simulations within the MATLAB® environment, which increases the simulation and optimization possibilities. The API not only allows easy editing of EnergyPlus files, but also allows the user to manage the use of all or part of the processing cores (parallelization) and facilitates the development of complementary scripts to perform analysis of the simulation results at the end of the process. The following section describes the use of this API in a real case scenario, developed for the European project SABINA (https://sabina-project.eu) [63].

3. Example of Use

In the scope of the European project SABINA [63], a model predictive control (MPC) that drives various HVAC systems and takes advantage of a building's thermal inertia was developed in MATLAB®. This algorithm needs to be run in conjunction with the EnergyPlus simulated model, in order to validate its performance. To interface the EnergyPlus model with the MATLAB® environment, some functions were written, following the guidelines mentioned above. Doing so allowed us to quickly test the behavior of the modeled building and interface it with the control algorithms.

The parameters we wanted to automatize through MATLAB® were the start and end date of the simulation (the run period) and the temperature set points of each room of the simulated building. To do this, we tagged those parameters with an identifiable text string, as described above, in the corresponding $*.idf$ file.

Following that, some structures could be defined to control the parameters. An example is shown below (see Listing 5).

Listing 5: Sabina_Launcher_Example.m file.

```
%% ================ Loading API ============================================
folder = pwd; % Path of launcher.m
NET.addAssembly(fullfile(folder,'EplusLauncher.dll'));
%% ================ Launcher configuration ================================
global Launcher;
Launcher = EplusLauncher.Launcher();
Launcher.MaxCores = 4;
Launcher.OutputFolderNumber = 0;
Launcher.JobsLogFile = fullfile(folder,'jobs.csv');
%% ================ EnergyPlus listeners ==================================
addlistener(Launcher,'OutputMessageReceived',@EnergyPlusMessage);
addlistener(Launcher,'SimulationCompleted',@SimulationFinished);
addlistener(Launcher,'JobsFinished',@JobsFinished);
%% ================ EnergyPlus configurations =============================
config = EplusLauncher.Configuration();
config.InputFile = fullfile(folder,'Sabina_BEM.idf');
config.WeatherFile = fullfile(folder,'Sabina_weather.epw');
config.RviFile = fullfile(folder,'launcher.rvi');
config.EnergyPlusFolder = 'C:\EnergyPlusV9-0-1\';
config.OutputFolder = fullfile(folder,'OutputJob');
%% ================  Parameters definition ================================
beginMonth = 1; % Defining runperiod for simulation
beginDayOfMonth = 18;
endMonth = 1;
endDayOfMonth = 20;
setpoint_base = zeros(1,144); % Generating setpoints vector (base)
setpoint_base(1:36) = 17;
setpoint_base(37:72) = 25;
setpoint_base(73:108) = 17;
setpoint_base(109:144) = 25;
x = rand(10,144); % Generating setpoints vectors (random)
for j = 1:144
    x(:,j) = 17+8*x(:,j);
end
All_values = [];
All_values = [All_values beginMonth beginDayOfMonth endMonth endDayOfMonth];
All_values = [All_values setpoint_base];
for k = 1:10
    All_values = [All_values beginMonth beginDayOfMonth endMonth endDayOfMonth];
    All_values = [All_values x(k,:)];
end
All_values = transpose(reshape(All_values,[148,11]));
[numRows,numCols] = size(All_values);
% Tags definition
All_tags = {};
All_tags = [All_tags,'#RP_BM#','#RP_BD#','#RP_EM#','#RP_ED#'];
for i = 1:144
    h = floor(i / 6);
    m = mod(i, 6) * 10;
    All_tags = [All_tags, ['#' num2str(h, '%02d') '_' num2str(m, '%02d') '#']];
end
%% ================ Simulation environment ================================
for j = 1:numRows
    job = EplusLauncher.Job(config);
    job.AddTags(All_tags);
    job.AddValues(All_values(j,:));
    Launcher.Jobs.Add(job);
end
%% ================ Run simulations =======================================
Launcher.Run();
```

In the [`parameters definition`] of this example, the simulation will start on 18 January and finish on 20 January (see `Defining runperiod for simulation`). Therefore, the simulation was carried out during the heating season.

Using the same methodology, a set of room temperature set points over 24 h, which control the simulation, were defined as follows. In this case, one new temperature set point value was used for each step of 10 min. No algorithm was implemented at this stage, as the development is currently ongoing. Therefore, a step function, varying from a room thermal set point of 17 °C to 25 °C, was applied in each room. For the development of this example, two kinds of set point vectors were defined: a baseline vector that used these limit values every six hours and a set of 10 random set point vectors between these limit values (see `Generating set points vector (base)` and `Generating set points vectors (random)`). These tests served to analyze how the simulated building followed the given set points. In the baseline case, it is noted that these important steps were applied to test the building's behavior in extreme conditions. Indeed, it is neither realistic nor efficient to apply such temperature changes in real conditions.

This temperature scheduler can now be used for any weekly/monthly scheduler to test each set point vector obtained from the MPC algorithms developed in SABINA.

The use of the EplusLauncher API's multi-core features enables multiple simulation jobs to be launched at the same time. This makes MPC feasible, due to its capacity to perform batches of simulations within reasonable time scales. In fact, in the European project the simulations are used to feed a neural network that will be in charge of obtaining the different MPC solutions. For the training, the parameterization of the simulations is the most appropriate strategy since the behavior of the model is taken into account given different external variables.

Finally, thanks to the fact that the simulations are managed by MATLAB®, it is possible to access the results directly and analyze them during the simulation process, using the toolboxes in MATLAB®. This facilitated the development of the SABINA project algorithms for the MPC.

Figure 3 illustrates the thermal control result using the baseline case vector. As explained above, the simulation consisted of three days of simulation during the heating season, using the set points of 17 °C and 25 °C as limit values every six hours. As mentioned above, this simulation serves exclusively to verify that the model follows the set points established in an extreme case, and does not make sense in real conditions.

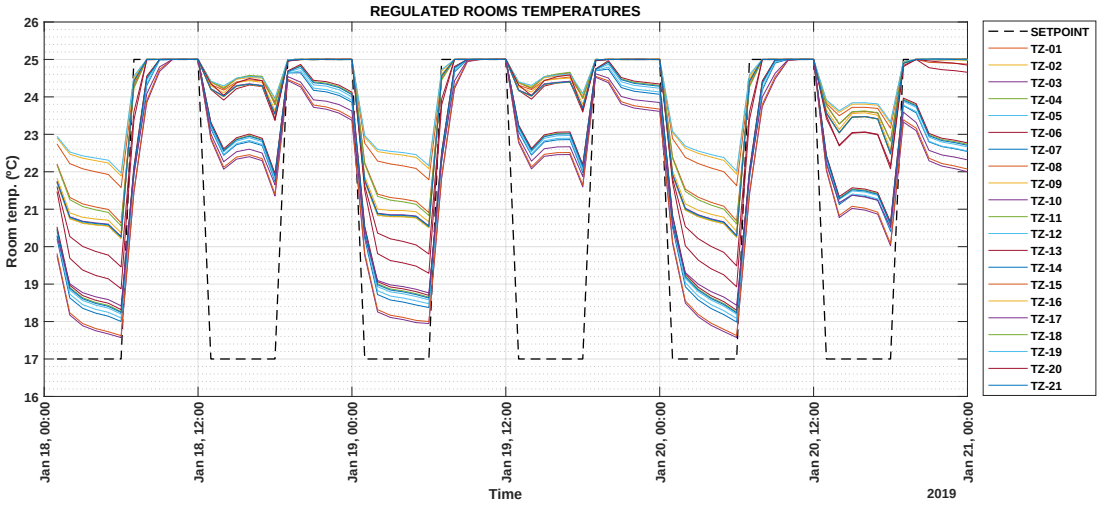

Figure 3. Example of thermal control of a building, simulated by EnergyPlus and controlled by MATLAB®.

4. Conclusions

This article describes the development and use of an application programming interface (API), which links an optimization tool (MATLAB®) and a building energy simulation software (EnergyPlus). This type of application is in high demand by the scientific community due to the rise of research

focused on improving buildings in order to combat climate change (e.g., energy conservation measures, improvements in HVAC systems, strategies for optimizing the use of buildings, and so on). In fact, this tool was born from a real need within an optimization process developed by the European project "SABINA" [63]. This project aims to develop new technologies and financial models to connect, control, and actively manage generation and storage assets to exploit synergies between electrical flexibility and the thermal inertia of buildings.

The tool was designed to be easy to use and can be easily improved using MATLAB® scripts that can complement its features. Its most important features are highlighted below:

- It allows the editing of EnergyPlus input files ($*.idf$) before performing simulations. These modifications are made through a process of searching and replacing tags, which provides great versatility to the editing process. This, together with other EnergyPlus utilities (such as Parametric:Logic, which allows changes to the $*.idf$ file according to certain values) gives this feature great potential for the optimization of building energy models.
- It allows (in a simple way) for selection of the quantity of cores to be used in each optimization process. This not only reduces the total time needed to perform an optimization process but, in cases where a high number of cores are available (i.e., a cluster), several optimizations can be performed at the same time.
- The API is structured in such a way that it is easy to introduce improvements (through scripts) in each of the phases of the simulation/optimization process. It allows for combination of the analysis capabilities of all MATLAB® toolboxes with the large amount of results that EnergyPlus provides, by allowing both to work under the same platform.

On the other hand, this API is only in its first version and, therefore, is in the testing process. Depending on demand and use, improvements will be developed in one or another direction. To date, the tool has yielded very good results and has proven to be highly suitable for many of the processes developed in the SABINA project.

Author Contributions: The conceptualization and the methodology were developed by C.F.B. and G.R.R. and G.C.G. developed the API. G.R.R. wrote the manuscript and performed the API testing. Y.S. and S.D. developed the examples to test the API in a real environment. All authors have read and agreed to the published version of the manuscript.

Funding: The work was funded by the research and innovation program Horizon 2020 of the European Union under the Grant No. 731211, project SABINA.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

ABM	Agent-Based Modeling
API	Application Programming Interface
BCVTB	Building Controls Virtual Test Bed
BEM	Building Energy Model
BPO	Building Performance Optimization
ECM	Energy Conservation Measures
FMI	Functional Mockup Unit
GHG	Green House Gases
HVAC	Heating Ventilation Air Conditioning
IPCC	Intergovernmental Panel on Climate Change
MPC	Model Predictive Control
SABINA	SmArt BI-directional multi eNergy gAteway
SLABE	Simulation-based Large-scale uncertainty/sensitivity Analysis of Building Energy performance
TAB	Thermally Activated Building

References

1. Poushter, J.; Huang, C. Climate change still seen as the top global threat, but cyberattacks a rising concern. *Pew Research Center*, 10 February 2019, pp. 1–37.
2. Carle, J. Climate change seen as top global threat. *Pew Research Center*, 14 July 2015, pp. 1–18.
3. Dean, B.; Dulac, J.; Abergel, T. *Towards Zero-Emission Efficient and Resilient Buildings*; Global Status Report; UN Environment and International Energy Agency: Paris, France, 2017.
4. Nguyen, A.T.; Reiter, S.; Rigo, P. A review on simulation-based optimization methods applied to building performance analysis. *Appl. Energy* **2014**, *113*, 1043–1058. [CrossRef]
5. Crawley, D.B.; Lawrie, L.K.; Winkelmann, F.C.; Buhl, W.F.; Huang, Y.J.; Pedersen, C.O.; Strand, R.K.; Liesen, R.J.; Fisher, D.E.; et al. EnergyPlus: Creating a new-generation building energy simulation program. *Energy Build.* **2001**, *33*, 319–331. [CrossRef]
6. Crawley, D.B.; Hand, J.W.; Kummert, M.; Griffith, B.T. Contrasting the capabilities of building energy performance simulation programs. *Build. Environ.* **2008**, *43*, 661–673. [CrossRef]
7. Attia, S.; Hamdy, M.; O'Brien, W.; Carlucci, S. Assessing gaps and needs for integrating building performance optimization tools in net zero energy buildings design. *Energy Build.* **2013**, *60*, 110–124. [CrossRef]
8. Wetter, M. GenOpt-A generic optimization program. In Proceedings of the Seventh International IBPSA Conference, Rio de Janeiro, Brazil, 13-15 August **2001**; pp. 601–608.
9. Yan, D.; O'Brien, W.; Hong, T.; Feng, X.; Gunay, H.B.; Tahmasebi, F.; Mahdavi, A. Occupant behavior modeling for building performance simulation: Current state and future challenges. *Energy Build.* **2015**, *107*, 264–278. [CrossRef]
10. Hu, M.; Xiao, F.; Jørgensen, J.B.; Wang, S. Frequency control of air conditioners in response to real-time dynamic electricity prices in smart grids. *Appl. Energy* **2019**, *242*, 92–106. [CrossRef]
11. Alibabaei, N.; Fung, A.S.; Raahemifar, K.; Moghimi, A. Effects of intelligent strategy planning models on residential HVAC system energy demand and cost during the heating and cooling seasons. *Appl. Energy* **2017**, *185*, 29–43. [CrossRef]
12. Bernal, W.; Behl, M.; Nghiem, T.X.; Mangharam, R. MLE+: A tool for integrated design and deployment of energy efficient building controls. In Proceedings of the Fourth ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings, Toronto, ON, Canada, 6 November 2012; pp. 123–130.
13. Smarra, F.; Jain, A.; de Rubeis, T.; Ambrosini, D.; D'Innocenzo, A.; Mangharam, R. Data-driven model predictive control using random forests for building energy optimization and climate control. *Appl. Energy* **2018**, *226*, 1252–1272. [CrossRef]
14. Kwak, Y.; Huh, J.H.; Jang, C. Development of a model predictive control framework through real-time building energy management system data. *Appl. Energy* **2015**, *155*, 1–13. [CrossRef]
15. Bianchini, G.; Casini, M.; Pepe, D.; Vicino, A.; Zanvettor, G.G. An integrated model predictive control approach for optimal HVAC and energy storage operation in large-scale buildings. *Appl. Energy* **2019**, *240*, 327–340. [CrossRef]
16. Corbin, C.D.; Henze, G.P.; May-Ostendorp, P. A model predictive control optimization environment for real-time commercial building application. *J. Build. Perform. Simul.* **2013**, *6*, 159–174. [CrossRef]
17. Knudsen, M.D.; Petersen, S. Model predictive control for demand response of domestic hot water preparation in ultra-low temperature district heating systems. *Energy Build.* **2017**, *146*, 55–64. [CrossRef]
18. Hedegaard, R.E.; Pedersen, T.H.; Petersen, S. Multi-market demand response using economic model predictive control of space heating in residential buildings. *Energy Build.* **2017**, *150*, 253–261. [CrossRef]
19. Reynolds, J.; Ahmad, M.W.; Rezgui, Y.; Hippolyte, J.L. Operational supply and demand optimisation of a multi-vector district energy system using artificial neural networks and a genetic algorithm. *Appl. Energy* **2019**, *235*, 699–713. [CrossRef]
20. Li, X.; Wen, J. Building energy consumption on-line forecasting using physics based system identification. *Energy Build.* **2014**, *82*, 1–12. [CrossRef]
21. Langevin, J.; Wen, J.; Gurian, P.L. Quantifying the human–building interaction: Considering the active, adaptive occupant in building performance simulation. *Energy Build.* **2016**, *117*, 372–386. [CrossRef]
22. Chung, W.J.; Lee, Y.S.; Lim, J.H. Cooling Operation Guidelines of Thermally Activated Building System Considering the Condensation Risk in Hot and Humid Climate. *Energy Build.* **2019**, *193*, 226–239. [CrossRef]

23. Kim, D.W.; Park, C.S. A heterogeneous system simulation of a double-skin faÇade. In Proceedings of the 12th International IBPSA Building Simulation Conference, Sydney, Australia, 14–16 November 2011.
24. Ibrahim, M.; Biwole, P.H.; Wurtz, E.; Achard, P. Limiting windows offset thermal bridge losses using a new insulating coating. *Appl. Energy* **2014**, *123*, 220–231. [CrossRef]
25. Zhao, J.; Lam, K.P.; Ydstie, B.E.; Loftness, V. Occupant-oriented mixed-mode EnergyPlus predictive control simulation. *Energy Build.* **2016**, *117*, 362–371. [CrossRef]
26. Tabares-Velasco, P.C.; Speake, A.; Harris, M.; Newman, A.; Vincent, T.; Lanahan, M. A modeling framework for optimization-based control of a residential building thermostat for time-of-use pricing. *Appl. Energy* **2019**, *242*, 1346–1357. [CrossRef]
27. Feng, J.D.; Chuang, F.; Borrelli, F.; Bauman, F. Model predictive control of radiant slab systems with evaporative cooling sources. *Energy Build.* **2015**, *87*, 199–210. [CrossRef]
28. Zhao, J.; Lam, K.P.; Ydstie, B.E. EnergyPlus model-based predictive control (EPMPC) by using MATLAB/SIMULINK and MLE+. In Proceedings of the 13th International IBPSA Building Simulation Conference, Chambéry, France, 25–28 August 2013.
29. Rouault, F.; Bruneau, D.; Sebastian, P.; Nadeau, J.P. Use of a latent heat thermal energy storage system for cooling a light-weight building: Experimentation and co-simulation. *Energy Build.* **2016**, *127*, 479–487. [CrossRef]
30. Yi, Y.K. Dynamic coupling between a Kriging-based daylight model and building energy model. *Energy Build.* **2016**, *128*, 798–808. [CrossRef]
31. Lee, Y.S.; Malkawi, A.M. Simulating multiple occupant behaviors in buildings: An agent-based modeling approach. *Energy Build.* **2014**, *69*, 407–416. [CrossRef]
32. Royer, S.; Thil, S.; Talbert, T.; Polit, M. A procedure for modeling buildings and their thermal zones using co-simulation and system identification. *Energy Build.* **2014**, *78*, 231–237. [CrossRef]
33. Ascione, F.; Bianco, N.; De Stasio, C.; Mauro, G.M.; Vanoli, G.P. A new methodology for cost-optimal analysis by means of the multi-objective optimization of building energy performance. *Energy Build.* **2015**, *88*, 78–90. [CrossRef]
34. Ascione, F.; Bianco, N.; De Stasio, C.; Mauro, G.M.; Vanoli, G.P. Multi-stage and multi-objective optimization for energy retrofitting a developed hospital reference building: A new approach to assess cost-optimality. *Appl. Energy* **2016**, *174*, 37–68. [CrossRef]
35. Ascione, F.; Bianco, N.; De Masi, R.F.; Mauro, G.M.; Vanoli, G.P. Resilience of robust cost-optimal energy retrofit of buildings to global warming: A multi-stage, multi-objective approach. *Energy Build.* **2017**, *153*, 150–167. [CrossRef]
36. Ascione, F.; Bianco, N.; De Stasio, C.; Mauro, G.M.; Vanoli, G.P. CASA, cost-optimal analysis by multi-objective optimisation and artificial neural networks: A new framework for the robust assessment of cost-optimal energy retrofit, feasible for any building. *Energy Build.* **2017**, *146*, 200–219. [CrossRef]
37. Ascione, F.; Bianco, N.; De Masi, R.F.; Mauro, G.M.; Vanoli, G.P. Energy retrofit of educational buildings: Transient energy simulations, model calibration and multi-objective optimization towards nearly zero-energy performance. *Energy Build.* **2017**, *144*, 303–319. [CrossRef]
38. Ascione, F.; Bianco, N.; Mauro, G.; Napolitano, D.; Vanoli, G. A Multi-Criteria Approach to Achieve Constrained Cost-Optimal Energy Retrofits of Buildings by Mitigating Climate Change and Urban Overheating. *Climate* **2018**, *6*, 37. [CrossRef]
39. Mauro, G.M.; Hamdy, M.; Vanoli, G.P.; Bianco, N.; Hensen, J.L. A new methodology for investigating the cost-optimality of energy retrofitting a building category. *Energy Build.* **2015**, *107*, 456–478. [CrossRef]
40. Ascione, F.; Bianco, N.; De Masi, R.; Mauro, G.; Vanoli, G. Design of the building envelope: A novel multi-objective approach for the optimization of energy performance and thermal comfort. *Sustainability* **2015**, *7*, 10809–10836. [CrossRef]
41. Surles, W.; Henze, G.P. Evaluation of automatic priced based thermostat control for peak energy reduction under residential time-of-use utility tariffs. *Energy Build.* **2012**, *49*, 99–108. [CrossRef]
42. Papadopoulos, S.; Azar, E. Integrating building performance simulation in agent-based modeling using regression surrogate models: A novel human-in-the-loop energy modeling approach. *Energy Build.* **2016**, *128*, 214–223. [CrossRef]
43. Kim, D.W.; Park, C.S. Manual vs. optimal control of exterior and interior blind systems. In Proceedings of the Eleventh International IBPSA Conference, Glasgow, Scotland, 27–30 July 2009; pp. 27–30.

44. Santos, L.G.R.; Afshari, A.; Norford, L.K.; Mao, J. Evaluating approaches for district-wide energy model calibration considering the Urban Heat Island effect. *Appl. Energy* **2018**, *215*, 31–40. [CrossRef]
45. Rackes, A.; Melo, A.P.; Lamberts, R. Naturally comfortable and sustainable: Informed design guidance and performance labeling for passive commercial buildings in hot climates. *Appl. Energy* **2016**, *174*, 256–274. [CrossRef]
46. Ascione, F.; Bianco, N.; De Stasio, C.; Mauro, G.M.; Vanoli, G.P. Simulation-based model predictive control by the multi-objective optimization of building energy performance and thermal comfort. *Energy Build.* **2016**, *111*, 131–144. [CrossRef]
47. Joint Research Centre of the European Commission. SimLab: Software Package for Uncertainty and Sensitivity Analysis. Available online: https://ec.europa.eu/jrc/en/publication/simlab-software-uncertainty-and-sensitivity-analysis (accessed on 16 January 2020).
48. Jin, Q.; Overend, M. Sensitivity of façade performance on early-stage design variables. *Energy Build.* **2014**, *77*, 457–466. [CrossRef]
49. Ascione, F.; Bianco, N.; De Masi, R.F.; De Stasio, C.; Mauro, G.M.; Vanoli, G.P. Multi-objective optimization of the renewable energy mix for a building. *Appl. Therm. Eng.* **2016**, *101*, 612–621. [CrossRef]
50. Oh, S.; Kim, Y.J.; Park, C.S.; Kim, I.H. Process-driven BIM-based optimal design using integration of EnergyPlus, genetic algorithm, and pareto optimality. In Proceedings of the IBPSA Building Simulation 2011 Conference, Sydney, Australia, 14–16 November 2011; pp. 894–901.
51. Calafiore, G.; Tommolillo, C.; Novara, C.; Fabrizio, E. APSEplus: A MATLAB toolbox for parametric energy simulation of reference buildings. In Proceedings of the 6th International Conference on Software and Computer Applications, Bangkok, Thailand, 26–28 February 2017; pp. 267–271.
52. Favoino, F.; Overend, M.; Jin, Q. The optimal thermo-optical properties and energy saving potential of adaptive glazing technologies. *Appl. Energy* **2015**, *156*, 1–15. [CrossRef]
53. Favoino, F.; Fiorito, F.; Cannavale, A.; Ranzi, G.; Overend, M. Optimal control and performance of photovoltachromic switchable glazing for building integration in temperate climates. *Appl. Energy* **2016**, *178*, 943–961. [CrossRef]
54. Zhang, Y.; Korolija, I. Performing complex parametric simulations with jEPlus. In Proceedings of the SET2010-9th International Conference on Sustainable Energy Technologies, Shanghai, China, 24–27 August 2010; pp. 24–27.
55. Delgarm, N.; Sajadi, B.; Kowsary, F.; Delgarm, S. Multi-objective optimization of the building energy performance: A simulation-based approach by means of particle swarm optimization (PSO). *Appl. Energy* **2016**, *170*, 293–303. [CrossRef]
56. Pang, Z.; O'Neill, Z. Uncertainty quantification and sensitivity analysis of the domestic hot water usage in hotels. *Appl. Energy* **2018**, *232*, 424–442. [CrossRef]
57. Li, H.; Wang, S.; Cheung, H. Sensitivity analysis of design parameters and optimal design for zero/low energy buildings in subtropical regions. *Appl. Energy* **2018**, *228*, 1280–1291. [CrossRef]
58. Delgarm, N.; Sajadi, B.; Delgarm, S. Multi-objective optimization of building energy performance and indoor thermal comfort: A new method using artificial bee colony (ABC). *Energy Build.* **2016**, *131*, 42–53. [CrossRef]
59. Delgarm, N.; Sajadi, B.; Delgarm, S.; Kowsary, F. A novel approach for the simulation-based optimization of the buildings energy consumption using NSGA-II: Case study in Iran. *Energy Build.* **2016**, *127*, 552–560. [CrossRef]
60. Barbosa, J.D.; Azar, E. Modeling and implementing human-based energy retrofits in a green building in desert climate. *Energy Build.* **2018**, *173*, 71–80. [CrossRef]
61. Leow, W.; Larson, R.C.; Kirtley, J.L. Occupancy-moderated zonal space-conditioning under a demand-driven electricity price. *Energy Build.* **2013**, *60*, 453–463. [CrossRef]

62. Thieblemont, H.; Haghighat, F.; Ooka, R.; Moreau, A. Predictive control strategies based on weather forecast in buildings with energy storage system: A review of the state-of-the art. *Energy Build.* **2017**, *153*, 485–500. [CrossRef]
63. Sabina-project. SABINA. SmArt BI-Directional Multi eNergy gAteway. Available online: https://sabina-project.eu/ (accessed on 16 January 2020).