A Distributed Multi-Model Platform to Cosimulate Multi-Energy Systems in Smart Buildings

A Distributed Multi-Model Platform to Co-Simulate Multi-Energy Systems in Smart Buildings

Daniele Salvatore Schiera, Luca Barbierato, Andrea Lanzini, Romano Borchiellini, Enrico Pons, Ettore Bompard, Edoardo Patti, Enrico Macii and Lorenzo Bottaccioli

***Abstract*—Nowadays, buildings are responsible for large consumption of energy in our cities. Moreover, buildings can be seen as the smallest entity of urban energy systems. On these premises, in this paper, we present a flexible and distributed co-simulation platform that exploits a multi-modelling approach to simulate and evaluate energy performance in smart buildings. The developed platform exploits the *Mosaik* co-simulation framework and implements the Functional Mock-up Interface (FMI) standard in order to couple and synchronise heterogeneous simulators and models. The platform combines in a shared simulation environment: *i*) the thermal performance of the building simulated with EnergyPlus; *ii*) a heat pump integrated with a PID control strategy modelled in Modelica to satisfy the heating demand of the building; iii) an electrical energy storage system modelled in MATLAB Simulink; and *iv*) different Python models used to simulate household occupancy, electrical loads, photovoltaic production and smart meters, respectively. The platform guarantees a plug-and-play integration of models and simulators, in which one or more models can be easily replaced without affecting the whole simulation engine. Finally, we present a demonstration example to test the functionalities, capability and usability of the developed platform and discuss future developments of our framework.**

***Index Terms*—Co-simulation, Functional Mock-up Interface, Mosaik, Building Energy System, Heat Pump, Photovoltaics, Electrical Energy Storage, Distributed Computing, Cyber-Physical Multi-Energy System, Systems Simulation.**

NOMENCLATURE

Acronyms

API	Application Programming Interface
BCVTB	Building Controls Virtual Test Bed
BES	Building Energy System
BIM	Building Information Modelling
CAD	Computer-Aided Design
COE	Co-simulation Orchestration Engine
CV	Control Valve
EHP	Electric Heat Pump
ESI	Energy System Integration
FMI	Functional Mock-up Interface
FMU	Functional Mock-up Unit
GIS	Geographic Information System
ICT	Information and Communications Technology
MES	Multi-Energy System
MPC	Model Predictive Control
PID	Proportional-Integral-Derivative
PV	Photovoltaics
RES	Renewable Energy Sources
TCP/IP	Transmission Control Protocol/Internet Protocol
UES	Urban Energy System

Variables & Symbols

Δt	Simulator time-step
A	Battery amperage
COP	EHP Coefficient Of Performance
GHI	Global Horizontal Irradiance
HH_{Behav}	Household behaviour variables vector
HH_{ELoad}	Household electrical load
HP_{ELoad}	EHP electrical consumption
PV_{EGen}	PV electrical generation
SOC	Battery State Of Charge
T_{DryBul}	Outdoor Dry-Bulb Temperature
T_{indoor}	Indoor Temperature
T_{set}	Indoor Set-point Temperature
V	Battery voltage
$Weather$	Weather variables vector

D. S. Schiera, A. Lanzini, R. Borchiellini, E. Pons and E. Bompard are with the *Department of Energy*, Politecnico di Torino, Turin, Italy.

L. Barbierato, E. Patti and L. Bottaccioli are with the *Department of Control and Computer Engineering*, Politecnico di Torino, Turin, Italy.

E. Macii is with the *Interuniversity Department of Regional and Urban Studies and Planning*, Politecnico di Torino, Turin, Italy.

All authors are with the *Energy Center Lab* research group, Politecnico di Torino, Turin, Italy (e-mail: name.surname@polito.it).

I. INTRODUCTION

URBAN Energy System (UES) mainly contribute to climate change due to high levels of energy consumption and greenhouse gas emission [1], [2]. To address these issues, efficient machines, processes, and ICT solutions have been proposed to convert, store, and manage energy, e.g., decentralised multi-energy production with high Renewable Energy Sources (RES) penetration, energy storage systems deployment, advanced cyber-physical energy infrastructures, and adoption of new energy vectors like green hydrogen. These new disruptive technologies will make UES more reliable, efficient, and less polluting. However, to push the growth towards more sustainable communities, it will be furtherly important to encourage Energy System Integration (ESI), as highlighted by the European Green Deal's action plan [3] and Clean Energy for all Europeans package [4]. ESI means the coordinated planning and operation of the energy system across multiple energy carriers, energy markets, end-use sectors, and above mentioned technologies at different functional layers to maximise efficiency, minimise waste, powering a climate-neutral circular economy, and achieve a low carbon future [3], [5].

UES falls under the concept of Multi-Energy Systems (MES), which are complex systems compliant with the ESI solutions where heterogeneous energy vectors (e.g. electricity, heat exchanging fluids, natural gas) are coupled together in such a multi-faceted way that they are challenging to be analysed comprehensively [6]. MES complexity is difficult to be understood without exploiting models merging cyber-

physical, economic, and social perspectives. Moreover, such analysis must also consider constraints and feedback from regulators, economic drivers, social and environmental behaviours [7]. Therefore, the major challenges associated with design, simulation, and management of UES use cases lie in an effective holistic analysis based on collaboration among different domain experts and integration of their specialised modelling and simulation tools. Indeed, experts from different domains may misunderstand each other due to different terminology and backgrounds and, usually, cannot communicate and exchange information with each other due to data formats incompatibility and license restrictions of tools.

In the last few decades, a robust research effort has been given to develop domain-specific simulation tools that have been used to simulate the behaviour of energy systems and solve the problems of a particular domain with high efficiency and accuracy [8]. A growing effort appears to focus on combining two or more modelling frameworks to integrate aspects of different domains and functional layers with the exploitation of novel methodologies, standards, and tools [9], such as co-simulation platforms. Co-simulation has been widely applied to integrate several models in order to represent and describe the complexity of UES [10], [11]. In particular, co-simulation platforms have been developed for studying applications such as new strategies for city energy supply and demand, urban energy planning, or distribution networks analysis and stability for an efficient RES penetration in a new smart citizen-centric energy system [12]–[14]. Moreover, new enabling technologies have been adopted to support the co-simulation techniques, such as Functional Mock-up Interface (FMI) [15] and *Mosaik* co-simulation framework [16]. Nevertheless, due to the complexity and heterogeneity of UES, the co-simulation platforms in literature seem to still face technical challenges in providing an easy-to-use and cross-domain scenario composition and co-simulation environment.

The great potential in energy-saving and grid balancing that smart buildings could offer has attracted many researchers in developing co-simulation frameworks tools specialised to model in details the Building Energy System (BES) by describing individual components and their complex interactions [18]–[25]. Indeed, buildings play a crucial role as they are responsible for roughly 40% of the overall energy consumption [26] and BES can be considered as the smallest entity of a larger UES. Table I reports a comparison of reviewed co-simulation frameworks for BES, highlighting the co-simulation techniques, the scenario design and composition procedures, and the use of stand-alone models. Commonly, the co-simulation techniques were used to enhance the modelling of individual components of BES by using EnergyPlus, which is one of the most widely used building energy modelling software [18]–[20], [22], [23]. Fewer researchers have performed co-simulation with EnergyPlus by use of a middleware coupling [20], [23], such as Building Controls Virtual Test Bed (BCVTB) [27] that allows to connect different simulation programs, such as Modelica, Radiance, and MATLAB/Simulink. However, BCVTB is not standardised and limited to the use of the ad-hoc integrated modules. Many other researchers have chosen a more flexible and standardised approach using FMI coupled with EnergyPlus and various software and tools [18], [19], [22] or FMI coupled with another building energy simulation tool in a Modelica-based environment [21]. However, while these reviewed co-simulation frameworks are suitable for analysing specific BES solutions, they seem to lack compelling flexibility and usability on easy and collaborative integration of several stand-alone domain-specific models for broader analysis of BES use cases [18]–[25]. Indeed, these frameworks generally use circumscribed co-simulation master algorithms or delimited co-simulation environments that make demanding and effort the integration of models and scenario composition [18], [19], [21], [22]. Moreover, in most cases [18], [19], [22], the implemented models are not stand-alone. Still, they are mainly inside of a few comprehensive simulators, thus partly losing the potential and advantages of the co-simulation framework. Furthermore, they mainly focus on coupling occupancy model with thermal models [18], [22] or thermal models with external heating/cooling system models [19], [21], or thermal models with MPC to test the effectiveness of the strategy [20]. Hence, none of these literature solutions integrate different domain-specific models together, such as users' occupancy, thermal and electrical energy demand and production of a building, heating and control systems.

On these premises, this paper presents a flexible and distributed co-simulation platform that exploits a multi-modelling approach in order to simulate and assess energy performance in smart buildings. The presented platform integrates heterogeneous simulation models by exploiting the flexible *Mosaik* co-simulation framework [16] that has been extended to embed Functional Mock-up Interface [15] allowing the interoperability among various simulation engines and tools. The main focus of this paper is on the infrastructure of the distributed co-simulation platform to demonstrate its functionalities, capability, and usability. In particular, the proposed platform permits a modular integration of several domain-specific models and simulation engines with a plug-and-play fashion through an effective cross-domain scenario configuration procedure. This solution tackles the challenges associated with the combination of software and knowledge across various domains, enhancing cooperation among the domain experts and making it easier to implement their models. Furthermore, our solution allows to locate each model in different network nodes which communicate with the master node (i.e. *Mosaik*) through Internet protocol suite (i.e. TCP/IP), providing a distributed co-simulation environment.

With respect to literature solutions compared in Table I, our platform simultaneously integrates different heterogeneous and distributed models. In our previous work [17], the platform has been assessed with a scenario that simulates the electrical and thermal energy demand and behaviour of a smart building energy system, which include: *i)* models for household occupancy, thermal demand, and indoor temperature scheduling, *ii)* models to realistically emulate appliances' load consumption, *iii)* photovoltaic simulator based on geographic information system, *iv)* an electric heat-pump with integrated control strategy, *v)* smart meters, and *vi)* weather information provided by third-party services through web-service communication.

TABLE I
PREVIOUS CO-SIMULATION FRAMEWORKS APPLIED TO THE MODELLING OF BUILDING ENERGY SYSTEMS.

Refs	CF	FMI	COE	DC	SDC	TM	EM	BM	PM	SM	HCCS	TPS	Are stand-alone models?
This work	×	×	Mosaik	×	Scenario Builder	×	×	×	×	×	×	×	Yes
Schiera et al. [17]	×	×	Mosaik		Single YAML setup	×	×	×	×		×	×	Yes
Chapman et al. [18]	×	×	No-MASS/FMI		Individual models setup	×		×			×		Partially
Thomas et al. [19]	×	×	CitySim/FMI		Individual models setup	×		×			×	×	Partially
Kwak et al. [20]	×		BCVTB		Individual models setup	×	×	×			×	×	Yes
Nicolai et al. [21]	×	×	SimulationX		Individual models setup	×		×			×		Yes
Wang et al. [22]	×	×	EnergyPlus/FMI		Individual models setup	×		×			×		Partially
Jia et al. [23]	×		BCVTB		Individual models setup	×		×			×		Yes

Acronyms used in table. CF: Co-simulation Framework, FMI: Functional Mock-up Interface, COE: Co-simulation Orchestration Engine, DC: Distributed Conmputing, SDC: Scenario Design & Composition, TM: Thermal Model, EL: Electrical Model, BM: Behavioural Model, PM: energy Production Model, SM: energy Storage Model, HCCS: Heating and Cooling Control System, TPS: Third-Party Services integration.

This paper extends our previous work [17] by adding the following novel features:

- The MATLAB Simulink engine was embedded in the co-simulation platform through FMI increasing the flexibility of our co-simulation platform and allowing the design and development of newer models exploiting the MATLAB simulation engine and its advanced simulation libraries. In order to demonstrate the MATLAB Simulink integration, the Simulink model of an electrical energy storage system has been included in the previous scenario.

- The *Scenario Builder* module was implemented to simplify the front-end input data and scenario configuration procedures. The new module facilitates cross-domain experts collaboration and assists in deploying their simulation models in the distributed co-simulation environment enhancing the plug-and-play integration. Moreover, the data I/O and visualisation were improved and integrated with the *Scenario Builder* functionalities.

- A new building energy system scenario was designed and implemented to assess energy performance and self-consumption evaluating a combined PV-battery system. The scenario was simulated with a demonstration example of a realistic house, and two different control strategies for regulating the heating system were evaluated to demonstrate the flexibility and usability of the co-simulation platform by easily substituting the models.

The remainder of the paper is structured as follows. Section II provides a detailed description of the different layers that compose our co-simulation platform. Section III shows the design and composition of a building energy system scenario with the description of each integrated model. Section IV presents the simulation results of the scenario to demonstrate the capability and usability of the proposed platform. Finally, Section V reports concluding remarks and future works.

II. CO-SIMULATION PLATFORM

The proposed distributed co-simulation infrastructure lays the foundations of a hybrid multi-modelling framework able to integrate the subsystems of a UES coming from different functional layers and domains of knowledge in order to pursue the ESI solutions as mentioned in the introduction section. To join these perspectives and help researchers in design, develop and test new components or solutions in such a complex energy system, a system architecture model was firstly developed in [28] by using a model-based system engineering approach to perform a holistic and systematic analysis of the UES use case, which is subsequently translated into an executable scenario and simulated on the distributed co-simulation platform.

This section focuses on the software infrastructure of the proposed co-simulation platform and its embodied elements, as depicted in Fig. 1, by showing its functionalities, capability, and usability of easily performing scenario design and composition as illustrated in Fig. 2, integrating domain-specific models, and simulating complex energy system scenarios. Indeed, each technological field expert could cooperate by deploying its own simulation models shared among the platform research community. In this view, the co-simulation platform is able to integrate already existing models performed by domain-specific simulation software in a shared and distributed co-simulation environment where the different models can be added/removed/replaced in plug-and-play fashion through a cross-domain scenario configuration procedure. From a software platform perspective, each model is seen as a grey-box model automatically connected with the other models based on the designed scenario. Therefore, each domain expert shall develop its model only, ensuring that it exposes the input/output variables and parameters necessary to build the scenario by establishing a common understanding with the co-simulation expert and the other domain experts.

To achieve the above-mentioned purposes, the platform was based on three main enabling technologies outlined in the

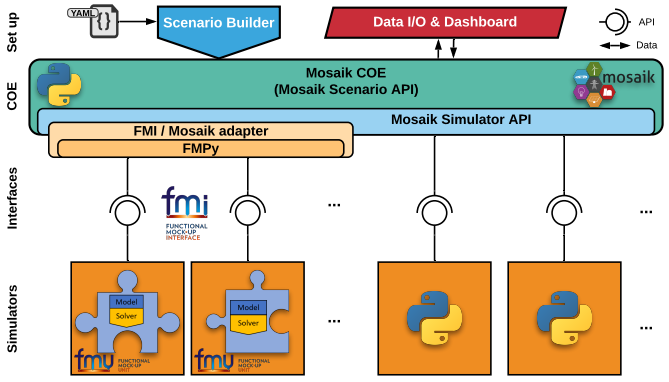

Fig. 1. Scheme of the distributed co-simulation infrastructure. It reports the enabling technologies and main elements of the platform: Scenario Builder, Data I/O & Dashboard interface, *Mosaik* COE including Scenario and Simulator APIs, FMI/Mosaik adapter, simulation blocks and their interfaces.

following paragraphs:

1) Co-simulation approach: Co-simulation has been identified as a flexible approach to integrate sub-models coming from different domain-specific models and simulation tools in a shared and distributed simulation environment. Essentially, co-simulation techniques allow integration of system of systems, each one simulated by a different simulator engine (or solver). Indeed, the domain-specific subsystems are modelled by the domain experts and usually simulated by their specialised solvers and modelling tools. Therefore, this approach preserves the use of efficient and suitable subsystems' solvers that are coupled to obtain a more complex dynamic system of systems simulation in terms of scalability, variety and composability of models. Moreover, co-simulation enhances performances in respect to stand-alone simulations thanks to the ability to distribute and parallelise the computation either in a cluster of servers and computers or in a cloud environment. Finally, co-simulation facilitates the hybrid multi-modelling approach, in which different modelling paradigms (e.g., agent-based, equation-based, discrete-event based) can be easily coupled into the shared environment.

2) Functional Mock-up Interface Standard: Commonly, domain-specific energy modelling tools and their solvers cannot communicate and exchange information with each other due to data formats incompatibility and license restrictions. They also often do not exchange information between different instances of the same simulation engine. To face these issues, Functional Mock-up Interface (FMI) [15] has been proposed as a tool-independent standard that allows *i*) to encapsulate model and its simulation engine, *ii*) to support direct control of the model through a standardised interface, and *iii*) to exchange data among different models [15]. In practice, FMI defines an interface based on a set of C-functions with the model that is implemented by an executable, called Functional Mock-up Unit (FMU). In a nutshell, FMU is a ZIP file that contains all the equations used by the model, its resources, documentation, and an XML file that describes the model structure and defines the variables used by FMU.

3) Co-simulation Orchestration Engine: The co-simulation approach requires a master algorithm to create instances of models and manage time evolution and regulation in a shared simulation environment, so-called *Orchestrator*. In recent years, different Co-simulation Orchestration Engines (COE) have been developed [9] based on specific application cases. Among them, the open-source co-simulation framework *Mosaik* [16] provides good performance, high usability, and flexibility. Indeed, *Mosaik* can be integrated with several power grid simulators (e.g. PYPOWER, Opal-RT, PowerFactory) and any other simulators written with various programming languages, such as Python and Java. Moreover, *Mosaik* allows the distribution of simulators exploiting TCP/IP communications used for data exchange.

As illustrated in Fig. 1, the software infrastructure of the co-simulation platform can be divided into different framework layers, where each layer can be distributed into different network nodes to improve co-simulation performances through such a distributed computation environment. The platform framework layers are: *i)* the set-up layer consisting of a *Scenario Builder* and *Data I/O & Dashboard* modules, *ii)* the *Mosaik* COE layer with the FMI/*Mosaik* adapter, *iii)* the interfaces layer between COE and simulators through *Mosaik* API, and *iv)* the layer of the attached simulators to perform the required scenario. The platform framework layers are outlined in the following subsections.

A. Scenario Builder, Data I/O & Dashboard

Scenario Builder module is a wrapper that simplifies and tailors the interface to *Mosaik* COE by translating input data and processing information through the different COE levels. The module composes the overall energy scenario through an effective cross-domain scenario configuration procedure that

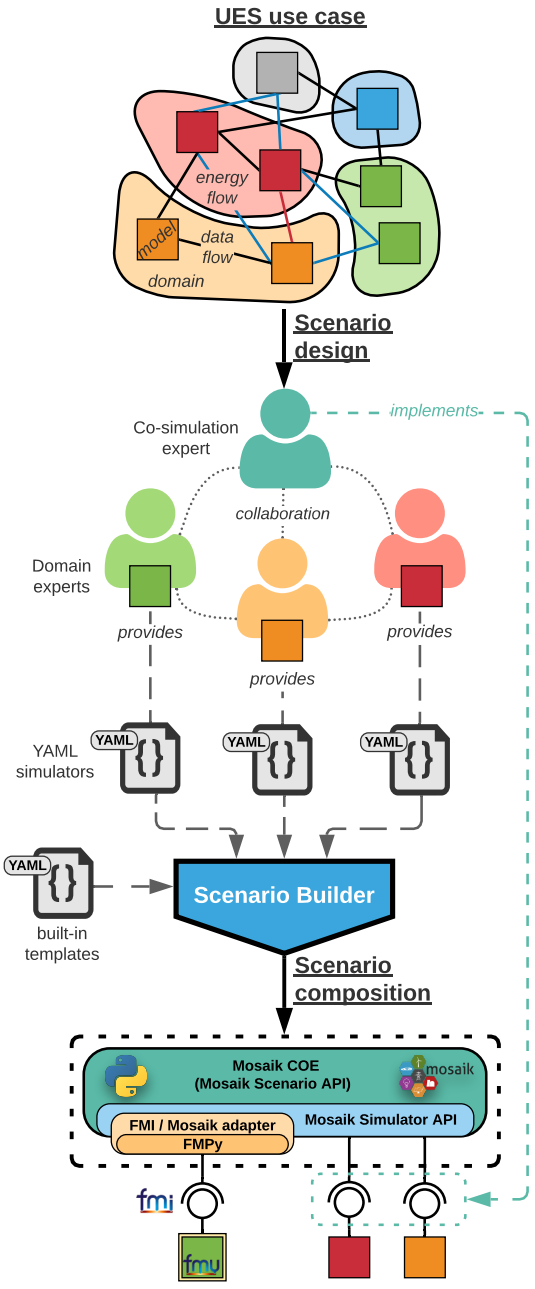

Fig. 2. Overview of the main steps to perform scenario design and composition starting from a UES use case within the co-simulation platform.

establishes a common understanding among the co-simulation expert and model domain experts, as illustrated in Fig. 2. In particular, the wrapper parses a YAML configuration file containing all information to set-up the entire co-simulation environment and distributes such information throughout the underlying classes and methods of *Mosaik* COE. YAML is a human-readable data serialisation standard based on nested objects as "*key: value*" structures, commonly used as a configuration file. Figure 3 depicts the adopted schema of a YAML file describing the scenario template structure and its main contents. In our co-simulation platform, the schema of the YAML configuration file is based on four root objects: *i) scenario configuration* that contains common scenario settings, *ii) simulator configuration* that includes models' instances with their simulation settings and initial conditions, *iii) connections* that contains topology of the connections among models' instances exploiting a list of run-time data exchanged for each connection, and *iv) scenario outputs* that includes setting to display or save the desired variables.

The use of YAML by the platform's users allows simplifying the scenario composition using default settings patterns of the objects and focus only on changing the desired parameters in a plug-and-play fashion. Indeed, it is possible to compose different YAML files containing template objects that share the adopted schema into a single YAML configuration file. For example, a modelist can focus on setting significant model instance's parameters and connections in agreement with the other experts and the scenario requirements, while later can add built-in YAML templates to set up the connection protocol interface and the shared simulation environment interface. Furthermore, if the platform's user explicitly requests the output of desired variables under the YAML root object *scenario outputs*, then the *Scenario Builder* automatically adds and connects back-end components that are necessary to provide the functionalities of the *Data I/O & Dashboard* module. Finally, the *Scenario Builder* module allows the centralisation of the set-up operation in a single network node that distributes the information remotely towards the other network nodes.

Data I/O module instead provides an interface to easily set and retrieve requested data coming from the co-simulation infrastructure. In particular the module is made by: *i)* an output interface for storing data through *Hierarchical Data Format* (HDF5), *ii)* an input interface for setting environment, scenario, and tunable parameters through a publish/subscribe pattern [29] using the open-source asynchronous messaging library *ZeroMQ*, as well as *iii)* an interface to communicate with a web app *Dashboard*. The *Dashboard* was developed to provide a user-friendly lab-view interface that provides a data flow graph among scenario's models, a run-time simulation view, and a canvas to plot the time-series of the desired variables from the output simulation data set.

B. Mosaik Co-Simulation Orchestration Engine

The *Mosaik COE* is the master node of the distributed co-simulation platform and provides time management and synchronisation to orchestrate the co-simulation environment during its run-time execution. Furthermore, it allows control and management of data flow among the models' instances. As shown in Fig. 2, *Mosaik* COE receives as input the scenario information from the *Scenario Builder* and set up the co-simulation environment exploiting:

- *Mosaik Scenario API* that allows to start simulators and instantiate their models to generate the co-simulation environment as described in the scenario configuration file, i.e. how the models' instances should be parameterised and interconnected with each other.
- *Mosaik Simulator API* that has to be implemented to establish an interface between *Mosaik* and a simulator to set up data exchange and orchestrate all stages of the co-simulation framework (i.e. initialisation, step management, data handling by getting/setting them from/to models' instances).

The COE manages the time synchronisation based on a *Discrete Event* synchronisation method. To do so, *Mosaik* sets a schedule based on the time-step description provided by each simulator. According to these descriptions, the schedule contains pre-defined synchronisation points and exploits a *directed acyclic schedule graph* to determine the order of

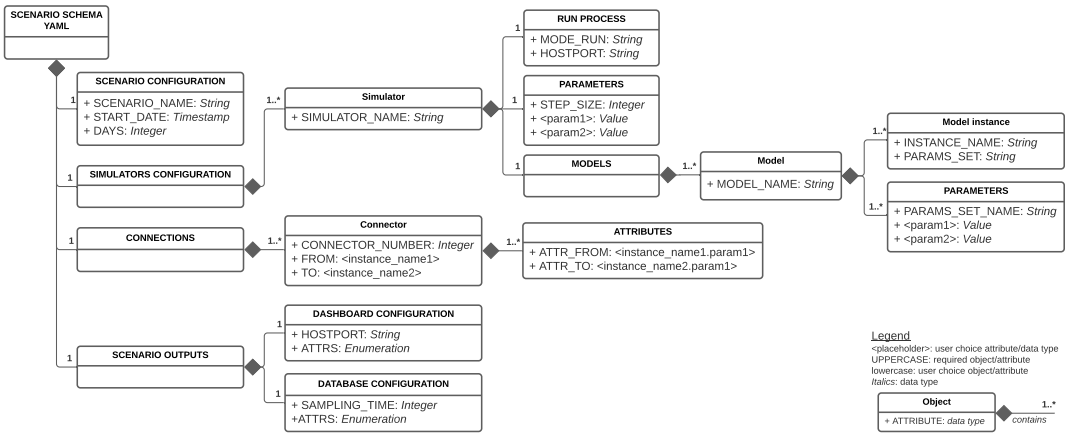

Fig. 3. The tree diagram from the YAML template scenario file showing the general characteristics of the scenario schema root element, with main parent elements and children elements of: *i)* scenario configuration, *ii)* simulators configuration, *iii)* connections topology of data flow, and *iv)* scenario outputs.

step commands, which are sent to each simulator. It is worth noting that this feature allows the integration of simulators with different time step resolution in the same co-simulation scenario, from one second up to ad lib.

C. FMI/Mosaik adapter

The FMI/*Mosaik* adapter was developed to map the *Mosaik Simulator API* with FMI functions by using the *FMPy* library [30], an open-source Python-based library to simulate FMUs, as shown in Fig. 1. Moreover, *FMPy* supports the latest versions of the FMU co-simulation and model-exchange methods, which were validated with cross-check rules defined by the FMI standard steering committee. The adapter allows loading and controlling FMU in the shared co-simulation environment that interacts with the FMU through C-functions to create one or more instances of a model, to manage simulation evolution, and to exchange data enabling interaction among different simulation engines. Therefore, the adapter automatise the integration of new encapsulated FMU models without the domain expert having to implement the *Mosaik Simulator API* specification, as shown in Fig. 2. Indeed, a YAML FMU simulator template is provided to standardise the platform's user data input procedure by establishing a common understanding with the other domain experts. Subsequently, the *Scenario Builder* can process and integrate the FMU simulator into the co-simulation environment automating the setup and execution of the FMU instances.

D. Simulators and Interfaces

Each model has to be integrated into the co-simulation environment by exploiting the *Mosaik Simulator API* specification to handle the COE-simulator communication. This preliminary procedure could be challenging because, normally, the model's domain expert does not know the co-simulation framework, and it would be necessary to collaborate with the co-simulation expert, as illustrated in Fig. 2. To make this procedure effective, the *Scenario Builder* module provides a standard YAML simulator template that helps to set up the simulation and model requirements through a grey-box modelling approach with minimum effort and cost. Therefore, it subsequently simplifies the model integration process into the co-simulation platform.

The co-simulation platform accepts both Python simulators and encapsulated models as FMUs thanks to FMI/*Mosaik* adapter. The simulators expose their input/output variables and parameters to the shared simulation environment, thus the *Scenario Builder* provides to connect them among each other and with the COE. Hence, the platform guarantees a plug-and-play integration of models and simulators, in which one or more models can be easily replaced without affecting the whole simulation engine. Finally, simulators can reside on different network nodes and communicate through TCP/IP sockets allowing distributed and parallelised computing to enhance co-simulation performances.

III. Scenario Design and Simulators

The flexibility provided by the co-simulation platform permits to use it as a virtual testbed for complex multi-energy systems, as well as conducting experimental research on enabling technologies. To demonstrate the capability of the presented platform in integrating and synchronising heterogeneous models written in different programming languages and executed with different software in a plug-and-play fashion, a well-consolidated and feasible building energy system scenario was designed and implemented. Indeed, it is out of the scope of the present research to propose new domain-specific models or scenarios since the focus of the present work is on the co-simulation infrastructure by showing the effective and simplified use of the platform layers to realise complex scenarios. This section provides a full description of designing building energy system scenario and the simulators implemented.

Fig. 4 illustrates the scheme of simulation blocks, focusing on blocks related to the cyber-physical elements of the system, and the energy and data flow among them. A simulation block represents an encapsulated entity of the system under analysis, which contains the stand-alone model that is able to interact with the environment exchanging data, parameters, and state variables dynamically. As illustrated in Fig. 2, the simulation blocks are added, parameterised and connected through the YAML configuration files provided by the domain experts and finally managed by *Scenario Builder* that instances the models and composes the overall scenario automatically. Each of the simulation block mentioned above are deepened in the following paragraphs.

The scenario consists of an EnergyPlus building model plugged into the platform through a FMU, and linked to a Modelica-based Electric Heat Pump model with a control system model, encapsulated in FMUs as well. Another FMU provides a battery system modelled in MATLAB Simulink. Furthermore, a photovoltaic system, the household behaviour, and weather data are provided to the building by linking stand-

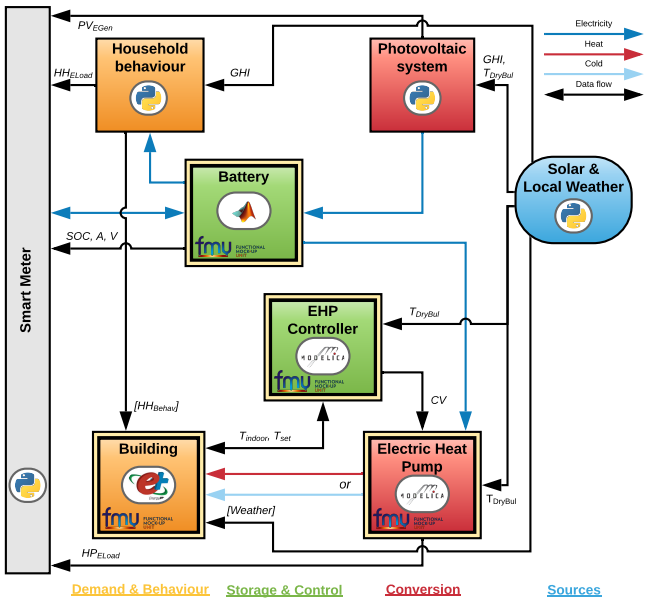

Fig. 4. Block diagram of the scenario designed within the co-simulation platform. The diagram shows both the energy- and data-flow connections between simulation blocks.

alone Python simulators. In addition to the simulation blocks illustrated in Fig. 4, the *Scenario Builder* automatically adds and connects the back-end blocks as described in the previous section.

i) Solar & Local Weather. Weather data are integrated from third-party data sources, such as Weather Underground [31], through an integrated Python application interface. As shown in Fig. 4, the application retrieves the information at the time step required by the models and distributes them in run-time through *Mosaik* that manages the time synchronisation. If the minimum available time-step of the weather data sources is greater than the time step required by a model, a linear interpolation is performed.

ii) Photovoltaic system. The Photovoltaic (PV) system was modelled by integrating the simulation infrastructure presented in [32]. The infrastructure allows to estimate the PV potential and to simulate the solar radiation profiles in real-sky conditions with a high spatio-temporal resolution (depending on the resolutions of the input data). It uses as inputs: *(a)* GIS data to describe building rooftops in terms of slope, orientation, possible obstacles, and shadows; *(b)* weather data provided by *Solar & Local Weather* block such as the Global Horizontal Irradiance in real-sky condition GHI and the outdoor Dry-Bulb Temperature T_{DryBul}. The PV simulation can be performed with the same time step as the resolution of the GHI data. The on-site generated electricity is primarily self-consumed by the household while any surplus is used to charge the battery or sent to the grid. The last case happens if the battery is already fully charged.

iii) Household behaviour. The household behaviour was integrated into the co-simulation platform by using the Python simulator proposed in [33]. The model uses a different kind of input data to create a non-homogeneous semi-Markov model for simulating the household electricity behaviour and thermal gains and retrieve the aggregated electricity and thermal load profiles. The Household behaviour simulator performs activity simulation with a resolution of 10 minutes, and it can provide appliances and an aggregate load profiles with a time step of one second up to ad lib.

The model creates the household by specifying the composition of the family, starting from census data. Then, the set of appliances in the households are distributed according to statistics obtained from *Use of Energy* surveys. Whilst, the statistics obtained from the *Time of Use* surveys were used to create a Semi-Markov model and generate each person behaviour. Finally, the simulator uses the created Semi-Markov model to generate household behaviour in terms of occupancy, type and duration of each activity performed by household's inhabitants, which is associated to specific usage of electric appliances (e.g., washing machine, dish-washer, vacuum cleaner, fridge, etc.). The simulator also uses weather data, provided by *Solar & Local Weather* block, to compute the energy consumption of domestic lighting systems according to solar radiation GHI.

In the end, the Household behaviour parses the number of occupants in a zone and their interactions with lights and appliances, providing aggregated loads, appliances and lights loads and schedules, which are given as input vector (HH_{Behav}) to the *Building* block.

iv) Electric Heat Pump. The air-to-water Electric Heat Pump (EHP) has been developed using Modelica's standard components by using the open-source OpenModelica modelling and simulation environment. The EHP model was exported as a FMU for co-simulation that contains the model and exposes inputs and outputs, as shown in Fig. 4. Moreover, the numerical solver is embedded and supplied by OpenModelica. Finally, the EHP model was interfaced through the FMI/*Mosaik* adapter.

The EHP model computes the sensible heat gain required to maintain the set-point temperature T_{set} in rooms. The EHP needs as input the regulation of the actuator provided by the EHP Controller. The coefficient of performance of EHP is set by a parametric relationship with the outdoor Dry-Bulb Temperature T_{DryBul} and the outlet flow temperature of the water-based underfloor heating system. The most implemented regulation system for the outlet flow temperature is the use of a climatic curve, which sets the temperature based on the outdoor temperature T_{DryBul} through a piece-wises linear function. The output of the EHP FMU is the heat requested by the *Building* block through the heating system. The measured variable T_{indoor} provided by the *Building* block is controlled to maintain the desired set-point T_{set} by implementing a Proportional-Integral-Derivative (PID) controller that acts on the water mass flow rate of the heating system through regulation of the control valve actuator (CV). PID is a negative feedback closed loop control system that calculates the error value as the difference between the desired set-point T_{set} and the measured value T_{indoor} and applies a correction based on proportional, integral and derivative terms on the CV actuator to regulate water mass flow rate and minimise the error over time. It is worth noting that thanks to the platform's flexibility, it is possible to replace the control system in a plug-and-play fashion with more advanced control systems.

v) Building. The building was modelled in EnergyPlus, a well-known open-source detailed building energy modelling engine that performs calculation of energy consumption in buildings, such as heating and cooling loads, disaggregated energy end-uses, and many other building-related features. Furthermore, EnergyPlus allows exporting the IDF file building model as a FMU for co-simulation through the Python package *EnergyPlusToFMU*. It is worth noting that EnergyPlus is a well-established energy simulation engine used as the core engine of many commercial and non-commercial energy simulation software, such as OpenStudio and DesignBuilder. These solutions provide a graphical interface and additional tools to improve the building's design and complexity, such as the import of CAD geometry, 3D model, or the direct import of building models from Building Information Modelling (BIM) tools. Therefore, the potential provided by Energy Plus and its extensions combined with the possibility of exporting the building model as a FMU unlock a perfect integration within the co-simulation platform, allowing flexibility and composability of building models with different level of complexity and design in function of the modelist's choices and the scenario objectives.

Inputs and outputs of the Building FMU are managed by EnergyPlus through three types of *External Interface* objects

being on the IDF file: *i)* the vector of the Household behaviour variables (HH_{Behav}) and heat gain provided by EHP for the water-based underfloor heating system were interfaced as input schedules, *ii)* the indoor temperature T_{indoor} and set-point temperature T_{set} were linked to external interface as output variables, and *iii)* the weather data needed by EnergyPlus (i.e., Dry-Bulb Temperature, Dew-Point Temperature, Relative Humidity, Barometric Pressure, Direct Normal Radiation, Diffuse Horizontal Radiation, Total and Opaque Sky Cover, Wind Direction, Wind Speed) were interfaced as input actuators passing a vector data ($Weather$). EnergyPlus can perform simulations with a minimum time step of one minute up to one hour.

vi) Battery. The electrical energy storage system was modelled in MATLAB Simulink by using the *generic battery model* provided by the Simscape Library, and it was encapsulated into a FMU for co-simulation. The model describes the dynamic behaviour of the most popular types of rechargeable batteries, and it can be fully parameterised using commercial battery data-sheet. The main parameters are the type of battery (e.g., Li-ion and lead-acid), nominal voltage, and rated capacity. The other detailed parameters are derived from the discharge characteristics and they are already implemented in the Simulink model for the common types of batteries.

The battery model was electrically connected with the production unit and the loads. A simplified controller manages the charge and discharge of the battery under the depth of discharge limit, prioritising self-consumption and, eventually, charging the battery only from the surplus of PV production. The main state variables of the battery (SOC, A and V) are sent to the *Smart Meter* to manage the energy flux of the entire system. For simplicity, in this scenario it was neglected the temperature and ageing effects on the battery. However, based on the capability of the battery model, it is possible to simulate these effects by interfacing it with the other blocks of the scenario, e.g., the *Solar & Local Weather* or the *Building* block to provide the ambient temperature to the battery. The present battery model is set to simulate the charge and discharge of the battery with a resolution of 10 minutes.

vii) Smart Meter. The virtual Smart Meter provides the physical and data interface between the building system and the distribution network. It receives the PV electrical generation (PV_{EGen}), the aggregated household electrical load (HH_{ELoad}), the EHP electrical consumption (HP_{ELoad}), and main state variables of the battery (SOC, A and V). It was used as a data collector manager returning the simulation results either in run-time or at the end of the simulation. The smart meter simulator can perform the simulation with whatever time step resolution without any limitation.

IV. Scenario Simulation and Results

In order to test the capability and usability of the presented co-simulation platform, the scenario designed in Section III was simulated for a hypothetical and realistic house located in Turin, Italy. The scenario is a test-case to evaluate the energy performance of a complex building energy system through the analysis of its dynamics and operation. Therefore, the YAML configuration file (see the schema in Fig. 3) was filled with all data required to set-up co-simulation environment, models and connections among them as described in the following.

Our test-case consists of a *Building* of about $150\ m^2$ equipped with a water-based underfloor heating system. It was modelled with the OpenStudio suite that supports whole building energy modelling using EnergyPlus engine. The software includes a SketchUp plug-in to graphically create geometry needed for EnergyPlus, as depicted in Fig. 5. Geometry, materials and construction layers of the building model have been chosen to be representative of a single-family house in Northern Italy. The main geometric and construction data of the building envelope are shown in Table II.

The *Solar & Local Weather* block provides weather data to the sub-models of the scenario giving the location and the reference year.

Before starting the co-simulation, the *Household behaviour* block generates the family, starting from the local socio-demographic and energy-related data. It consists of four members.

The PV system is composed of 16 panels on the roof-top south oriented with a 37° tilt angle. It provides a total of $5\ kW_p$ to the premises.

An Electrical Heat Pump (EHP) is installed in order to satisfy the heating demand of the house with a power input of $3\ kW_e$ and a COP of 4 at nominal conditions. The heat pump is equipped with a PID controller and an inertial flywheel to decouple the heat generator with the underfloor heating system. The performance characteristics were retrieved from data-sheets of common residential heat pumps. In the winter season, the desired indoor temperature was set to $20\ °C$[1] from 6:00 to 22:00, and $16\ °C$ for the remaining day's hours.

A Lithium-Ion battery is installed with a rated capacity of $60\ Ah$ and a nominal voltage of $200\ V$ and located inside the house. The parameters of the discharge characteristics were derived from the built-in Li-Ion battery Simulink model.

The *Scenario Builder* module parses the YAML configuration file to retrieve all the required information to perform the scenario simulation and distributes them to COE through *Mosaik* API. The simulation blocks are instantiated using the *Mosaik Simulator API* (retrieving model settings, parameters,

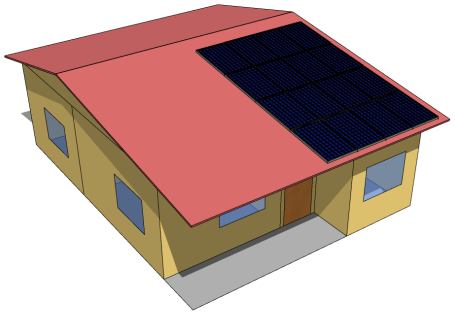

Fig. 5. The 3D layout of the house modelled with the OpenStudio suite and SketchUp plug-in.

[1]The Italian regulation establishes private homes are not supposed to be heated to more than $20\ °C$, although the norms allow a margin of $2\ °C$.

TABLE II
MAIN GEOMETRIC AND CONSTRUCTION DATA OF THE BUILDING ENVELOPE.

Quantity	Unit	Value
Conditioned net floor area	m^2	136.82
Conditioned gross volume	m^3	412
Gross Wall Area	m^2	134.64
Gross Window-Wall Ratio	-	0.26
U-value external wall	$Wm^{-2}K^{-1}$	0.427
U-value floor (roof)	$Wm^{-2}K^{-1}$	1.260
U-value window	$Wm^{-2}K^{-1}$	2.674

constants, and time-step) and connected to each other, as shown in Fig. 4, via *Mosaik Scenario API*. The time-steps Δt were set considering the scenario characteristics, capability of the solvers, and computational effort: EnergyPlus (i.e. the Building) 10 min, Modelica (i.e. the Electric Heat Pump) 5 min, MATLAB Simulink (i.e. the Battery) 10 min, PV simulator 15 min and household behaviour 10 min. The *Solar & Local Weather* block provides data to each simulation engine at the requested time-step. The scenario was simulated for a whole thermal season. The distributed co-simulation platform resides in our campus and involves five computers (or nodes) all connected between them by using a Local Area Network. Each network node is an Intel® Xeon® E3-1245v5 CPU@3.50Ghz with 32GB DDR4@2133MHz RAM. The different simulators and models are distributed across the five computers as follows:

- **Node A** Scenario set-up and Mosaik COE;
- **Node B** Python simulators (i.e. Household behaviour, PV, Smart Meter, and Solar & Local Weather);
- **Node C** EnergyPlus (i.e. Building FMU);
- **Node D** MATLAB Simulink (i.e. Battery FMU);
- **Node E** Modelica (i.e. EHP FMU).

The main results of the simulation are depicted in Fig. 6. The figure shows a general time window of two consecutive days of two months, i.e. a weekend of January in Fig. 6(a)-(e) and two working days of March in Fig. 6(f)-(j). The chosen months represent the colder and hotter month of the thermal season in the given location. The main characterising data for analysing the energy building behaviour are the occupancy, battery profile and State Of Charge (SOC), total and disaggregated load profiles, Coefficient Of Perfomance (COP) of EHP, PV production, net power and net energy im-

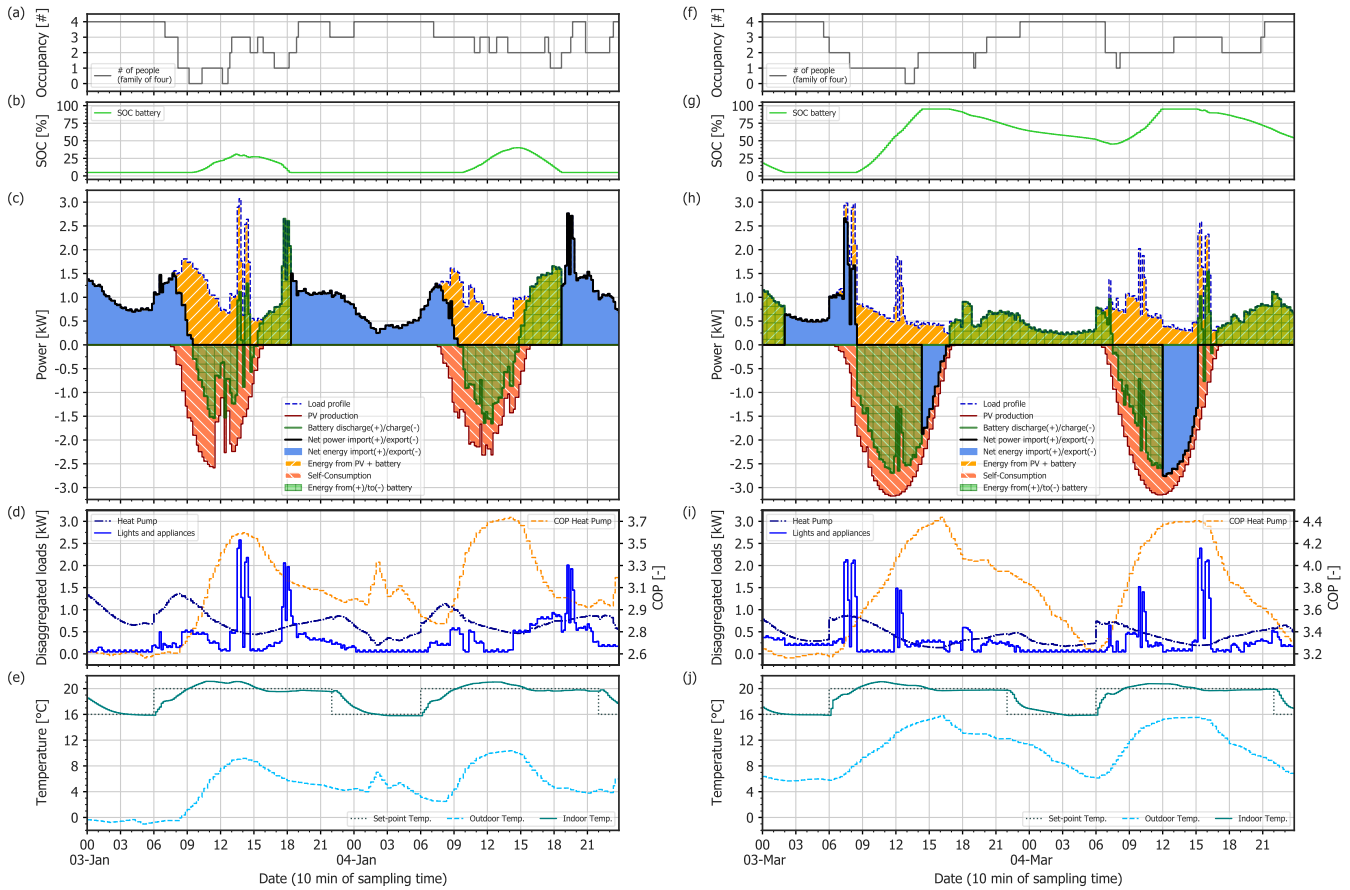

Fig. 6. The figure presents the scenario simulation results showing a general time window of two consecutive days in two months of the winter season, i.e., a weekend of January *(a)-(e)* and two working days of March *(f)-(j)*. The time-series plots depict: *(a)* and *(f)* the number of people in the house; *(b)* and *(g)* the State Of Charge (SOC) of the battery; *(c)* and *(h)* the profiles of the net power, total load, PV production, and battery, highlighting the self-consumption and the energy covered by the combined PV-battery system; *(d)* and *(i)* the view of load profile in terms of its disaggregated elements, i.e., lights, appliances, and heat pump, the latter linked to its Coefficient Of Performance (COP); *(e)* and *(j)* the outdoor, indoor, and scheduled set-point temperatures.

port/export measured on the meter, self-consumption, energy from/to battery, energy covered from combined PV-battery system and, finally, outdoor and indoor temperatures with the schedule of the desired temperature. The data were collected with a sampling time of 10 minutes. Nevertheless, it can be requested from the co-simulation platform information from either each module that exposes inputs/outputs and parameters or simulation environment directly at wherever sampling time.

The household behaviour is resumed by the occupancy in Fig. 6(a) and (f). Occupancy is one of the most important factors affecting the building energy demand for heating/cooling by varying conditioning periods and house settings, as well as the electricity demand by the use of the EHP itself, lights, and appliances as shown in the disaggregated loads plot in Fig. 6(d) and (i). As shown in Fig. 6(e) and (j), the set-point temperature's schedule affects the thermal demand from the EHP and, as a consequence, its electrical consumption. Moreover, the electric load requested by the EHP to satisfy the heating demand is considerably higher than the electrical household consumption due to appliances and lights, especially on colder days. Therefore, the resulting aggregated load profile, shown in Fig. 6(c) and (h), almost follows the EHP load profile. For example, in the weekend of January during the morning from 6:00 to 9:00, the EHP consumption increases until it reaches a maximum power absorption of about $1.4\ kW$ to cover the increased heating demand due to the change of indoor set-point temperature. Meanwhile, people leave the house and, as a consequence, lights and appliances consumption becomes minimum and does not exceed $0.5\ kW$ of absorption. However, on the same morning hours during the working days of March, it takes place different power peaks of about $2\ kW$ cause by more use of appliances and lights.

By comparing the curves of EHP load and its COP in Fig. 6(d) and (i), the EHP consumption is almost higher when the COP drops and vice versa. An example can be seen on the afternoons of January, where the EHP absorbs about $0.5\ kW$ while its COP reaches about 3.6. More noticeable on March, where COP reaches 4.4 while the EHP absorbs only $0.25\ kW$. This typical behaviour is linked to the outlet flow temperature of the EHP that is regulated by a climatic curve. Indeed it is regulated considering the outdoor temperature. As a consequence, the COP qualitatively follows the outdoor temperature.

During winter days, the outdoor temperature can quickly swing, as shown in the temperatures plot in Fig. 6(e) and (j), even reaching temperatures below zero as in January. The requested heating demand is satisfied by the EHP and, thanks to the PID control strategy, the room temperature remains around the set-point temperatures with few little oscillations. The PID controller was tuned during run-time simulation till reaching optimum values and stability for the desired control response. Indeed, the tunable parameters can be changed during execution through *Data I/O* interface, thus directly seeing the feedback results on simulation.

Following the power-related plot in Fig. 6(c) and (h), we can see how the combined PV-battery system affects the load profile of the building measured on the meter (net power). The energy management system rules the energy flows by maximising self-consumption: the in-site electricity production is directly used to cover the load reducing the imported power from the grid, and any surplus is used to charge the battery whenever it is not full. Otherwise, the power surplus is exported to the grid. The load profile and PV production curves show the timing imbalance between peak demand and renewable energy production. The battery helps to reduce the timing imbalance by shifting the load: it is charged during daylight by solar power, then it is discharged, providing power into the premises in the evening, thus reducing imported power and increasing self-consumption. The Fig. 6(b) and (g) show the SOC curve during the analysed days, which are strictly correlated with the charging and discharging phases depicted in Fig. 6(c) and (h). As can be seen in the figures, a change of sign on the first derivative of the SOC curve from positive to negative is equivalent to a change of phase of the battery from power charging to power discharging, and vice versa. When the first derivative is null, corresponding to SOC equal to the upper (95%) or lower limit (5%), the battery power is null.

During the days of March, the combination of good solar radiation and low EHP load results in high levels of SOC up to full charge, thus increasing the opportunity to sell energy to the grid. Moreover, the storage capacity of the battery permits to cover the entire evening and night loads. As an example, on March 3^{rd} following Fig. 6(g)-(h), the daily PV production was $19.91\ kWh$: about $17.68\ kWh$ was self-consumed of which $5.96\ kWh$ directly used to cover the load and $11.72\ kWh$ used to charge the battery, while the remaining $2.23\ kWh$ was sold to the grid. In particular, the battery starts charging at 8:30 until reaching the upper charge limit at 14:00 when the power surplus begins to be sold until the end of PV production. From 18:00 until the next beginning of the daily PV production, the battery succeeds in covering the entire load, even remaining with a SOC of about 50%. On the other hand, during the colder and cloudy days of January resulting in less solar radiation, the power export rarely happens and the battery works at low operating levels. In any case, the exploitation of the accumulated capacity permits to cover a significant part of the evening load. For instance, on January 4^{th} following Fig. 6(b)-(c), the daily PV production is entirely self-consumed. A part of this energy was exploited to charge the empty battery up to about 40% of SOC and later used to cover about $4.54\ kWh$ of the load during the afternoon until 19:00.

The co-simulation platform simulates the complexity of the building energy system's dynamics with great details and realism by integrating each subsystem with its solver's flexibility and efficiency. Notably, as depicted in Fig. 6(c) and (h), the daily PV production profiles of the two months analysed are affected by the real-sky weather conditions resulting in markedly different. During the two days analysed in January, the power peaks of PV do not exceed $2.6\ kW$ and their profiles result jagged due to the cold and cloudy days of the winter, with a total in-site electricity production of $22.15\ kWh$. Going towards spring in March, the PV profiles are more uniform due to warmer and clear days, reaching power peaks of almost $3.5\ kW$ with a total production of $40.38\ kWh$. On the other hand, the energy management system actively responses to the

variability of PV production and load request by charging and discharging the battery.

A. Testing of a different control strategy.

The flexibility and usability of the proposed co-simulation platform permits us to easily implement experiments testing different solutions. Indeed, it is only needed to replace the desired simulator block in a plug-and-play fashion through the *Scenario Builder* module. The module sees the simulator as a grey-box model, thus it will only be necessary to re-link the input/output variables with the other models and set the internal parameters of the new integrated block through the YAML configuration file.

By way of example, the PID controller of the EHP was replaced with a more simpler hysteresis controller that turns ON/OFF the heating system if the T_{indoor} exceeds -2/+2 $°C$ with respect to T_{set}. The indoor temperatures trends and EHP loads obtained from the simulation of the two control strategies were compared and depicted in Fig. 7 during the days of March. As it can be seen, the PID controller regulates the indoor temperature better than the hysteresis one, hence reducing the discomfort by maintaining on average the temperature closer to the chosen set-point. From an energy point of view, the EHP regulated with the PID controller performs better than the hysteresis controller by reducing its energy consumption by about 33%. Indeed, during the two days of March, EHP with PID controller consumes 53.92 kWh, while EHP with hysteresis controller consumes 79.77 kWh.

In the end, there is no limitation on implementing more advanced control strategies as their usage is strictly dependent on the use case analysed only. For example, it is possible to implement more advanced control strategies and optimisation algorithms that, for instance are based on Model Predictive Control or deep Reinforcement Learning. For example, a general control block can be implemented as follow: the emulator could be a detailed EnergyPlus model or even a hardware-in-the-loop model of a real building that could share information through a TCP/IP connection with the possibility to control the heating system within the co-simulation platform remotely; to perform the predictions, a validated dynamic model based on reduced order RC model can be implemented directly in Python or by using dedicated libraries, which can be easily integrated into the co-simulation platform by implementing the *Mosaik Simulator API*; the optimisation algorithms can be implemented by exploiting plenty of different libraries as well. Finally, the connection of the various elements of the control block with the rest of the system can be made through the *Scenario Builder* module, similarly to what was made with the other simulation blocks. In the same way, the flexibility and usability of the proposed co-simulation platform also permits to effectively integrate algorithms that are able to analyse the system more in depth, for example, to find optimal control and operating strategies concerning a pre-defined objective function, e.g., maximising the total self-consumption, which increases the savings by buying less energy from the grid. Regarding the system plants' sizing, in this work, the cost-benefits analysis of the optimal PV and battery sizes was out of the scope. However, the co-simulation platform allows to find the optimal sizes of the subsystems performing optimisation by simulation, or directly using advanced optimisation tools by exploiting the capability of the platform to perform a hybrid multi-modelling simulation. In conclusion, the simplified scenario composition and implementation procedures demonstrate the usability of the co-simulation platform, where different disciplinary fields and tools can be integrated by sharing common knowledge to build even more complex energy scenarios.

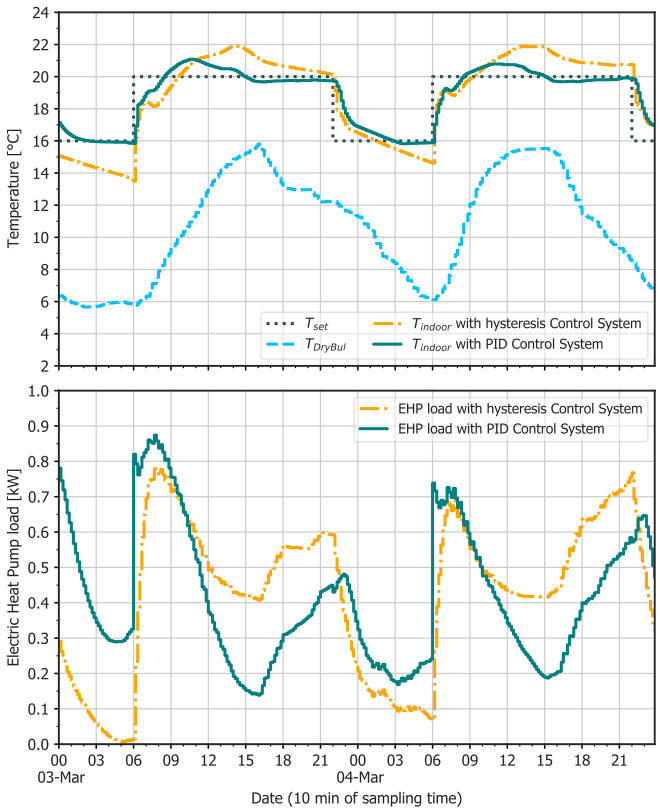

Fig. 7. Comparison of the indoor temperatures trends and EHP loads by implementing different control strategies: PID and hysteresis Control Systems.

V. Conclusion

This paper presents a distributed co-simulation platform able to perform multi-modelling scenario simulation of smart building energy systems, evaluating the energy performance by analysing complex dynamics, operations, and control strategies. The platform permits integrating several and heterogeneous simulators by coupling *Mosaik* framework and the FMI standard in a shared and distributed environment. Indeed, the platform applies distributed computing to parallelise simulators execution of different models. This choice enhances the flexibility and scalability of the overall co-simulation platform, extending feasible interconnections between models and simulators. Moreover, the scenario making process was simplified and centralised by adopting a single standardised configuration file managed by the *Scenario Builder* module. In this way, the platform permits to easily plug-and-play one or more models with predefined settings patterns. A demonstration example

of a building energy system was designed, and a test-case scenario was presented to test the functionalities, capability and usability of the co-simulation platform. From results, it is possible to observe how the platform can co-simulate different aspects of a building with a high level of detail and complexity without losing the flexibility and efficiency of the integrated subsystems solvers.

The proposed platform is meant to smartly optimise the planning and operation of the building energy systems or test new advanced solutions. In future works, the presented framework will be extended to co-simulate an entire urban district with its distribution networks. Indeed, it is possible to simulate plenty of different buildings considering the dynamic behaviour of the energy networks. This kind of scenarios can be simulated *i)* to analyse and optimise UES use cases towards ESI, *ii)* to improve the decision-making process, *iii)* and to provide adequate energy policies and interventions at the urban scale. Furthermore, an agent-based approach can be implemented to model the entities and actors that constitute the complex socio-economic urban energy system like distributors, aggregators, energy communities, which govern their physical systems interacting with each other. Finally, software models of physical components will be replaced with real devices (e.g. smart meters, PV and battery systems) unlocking real-time Hardware-In-the-Loop co-simulations.

REFERENCES

[1] U. DESA, "World urbanization prospects: The 2018 revision," 2019.

[2] United Nations, "Energy, UN-Habitat," https://unhabitat.org/urban-themes/energy/.

[3] European Commission, "Communication from the commission to the European Parliament, the Council, the European Economic and Social Committee and the Committee of the Regions. Powering a climate-neutral economy: An EU Strategy for Energy System Integration, COM/2020/299 final," https://eur-lex.europa.eu/legal-content/EN/ALL/?uri=COM:2020:299:FIN, 2020.

[4] European Commission, "Clean energy for all Europeans," *Luxemburg: Publication Office of the European Union*, 2019.

[5] C. Cambini, R. Congiu, T. Jamasb, M. Llorca, and G. Soroush, "Energy systems integration: implications for public policy," *Energy policy*, vol. 143, p. 111609, 2020.

[6] P. Mancarella, "MES (multi-energy systems): An overview of concepts and evaluation models," *Energy*, vol. 65, p. 1–17, 02 2014.

[7] L. Abrardi, "Behavioral barriers and the energy efficiency gap: a survey of the literature," *Journal of Industrial and Business Economics*, vol. 46, no. 1, pp. 25–43, 2019.

[8] H.-K. Ringkjøb, P. M. Haugan, and I. M. Solbrekke, "A review of modelling tools for energy and electricity systems with large shares of variable renewables," *Renewable and Sustainable Energy Reviews*, vol. 96, pp. 440–459, 2018.

[9] M. Vogt, F. Marten, and M. Braun, "A survey and statistical analysis of smart grid co-simulations," *Applied energy*, vol. 222, pp. 67–78, 2018.

[10] L. Bottaccioli, A. Estebsari, E. Pons, E. Bompard, E. Macii, E. Patti, and A. Acquaviva, "A flexible distributed infrastructure for real-time co-simulations in smart grids," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 6, pp. 3265–3274, 2017.

[11] D. S. Schiera, F. D. Minuto, L. Bottaccioli, R. Borchiellini, and A. Lanzini, "Analysis of rooftop photovoltaics diffusion in energy community buildings by a novel gis-and agent-based modeling co-simulation platform," *IEEE Access*, vol. 7, pp. 93 404–93 432, 2019.

[12] V. Reinbold, C. Protopapadaki, J.-P. Tavella, and D. Saelens, "Assessing scalability of a low-voltage distribution grid co-simulation through functional mock-up interface," *Journal of Building Performance Simulation*, pp. 1–13, 2019.

[13] C. Steinbrink, M. Blank-Babazadeh, A. El-Ama, S. Holly, B. Lüers, M. Nebel-Wenner, R. P. Ramírez Acosta, T. Raub, J. S. Schwarz, S. Stark *et al.*, "Cpes testing with mosaik: Co-simulation planning, execution and analysis," *Applied Sciences*, vol. 9, no. 5, p. 923, 2019.

[14] L. Barbierato, A. Estebsari, L. Bottaccioli, E. Macii, and E. Patti, "A distributed multimodel cosimulation platform to assess general purpose services in smart grids," *IEEE Transactions on Industry Applications*, vol. 56, no. 5, pp. 5613–5624, 2020.

[15] Modelica Association Project, "Functional Mockup Interface," https://fmi-standard.org/.

[16] OFFIS - Institute for Information Technology, "Mosaik," https://mosaik.offis.de/.

[17] D. S. Schiera, L. Barbierato, A. Lanzini, R. Borchiellini, E. Pons, E. F. Bompard, E. Patti, E. Macii, and L. Bottaccioli, "A distributed platform for multi-modelling co-simulations of smart building energy behaviour," in *2020 IEEE International Conference on Environment and Electrical Engineering and 2020 IEEE Industrial and Commercial Power Systems Europe (EEEIC / I CPS Europe)*, 2020, pp. 1–6.

[18] J. Chapman, P.-o. Siebers, and D. Robinson, "Multi-agent stochastic simulation of occupants for building simulation," in *In Proc. of: Building Simualtion 2017 IBPSA*, 2017.

[19] D. Thomas, C. Miller, J. Kämpf, and A. Schlueter, "Multiscale co-simulation of energyplus and citysim models derived from a building information model," in *In Proc. Of: IBPSA Bausim 2014*, 2014, pp. 469–476.

[20] Y. Kwak, J.-H. Huh, and C. Jang, "Development of a model predictive control framework through real-time building energy management system data," *Applied Energy*, vol. 155, pp. 1–13, 2015.

[21] A. Nicolai and A. Paepcke, "Co-simulation between detailed building energy performance simulation and modelica hvac component models," in *In Proc. of: 12th International Modelica Conference, Prague, Czech Republic, May 15-17, 2017*, no. 132. Linköping University Electronic Press, 2017, pp. 63–72.

[22] K. Wang, P.-O. Siebers, and D. Robinson, "Towards generalized co-simulation of urban energy systems," *Procedia engineering*, vol. 198, pp. 366–374, 2017.

[23] M. Jia and R. Srinivasan, "Building performance evaluation using coupled simulation of energyplus™ and an occupant behavior model," *Sustainability*, vol. 12, no. 10, p. 4086, 2020.

[24] A. Raad, V. Reinbold, B. Delinchant, and F. Wurtz, "Energy building co-simulation based on the wrm algorithm for efficient simulation over fmu components of web service," *Proceedings of the BS*, vol. 15, 2015.

[25] T. Hong, H. Sun, Y. Chen, S. C. Taylor-Lange, and D. Yan, "An occupant behavior modeling tool for co-simulation," *Energy and Buildings*, vol. 117, pp. 272–281, 2016.

[26] European Parliament, "Directive 2010/31/EU of the European Parliament and of the Council of 19 May 2010 on the energy performance of buildings," 2010.

[27] M. Wetter, "Co-simulation of building energy and control systems with the building controls virtual test bed," *Journal of Building Performance Simulation*, vol. 4, no. 3, pp. 185–203, 2011.

[28] L. Barbierato, D. S. Schiera, E. Patti, E. Macii, E. Pons, E. F. Bompard, A. Lanzini, R. Borchiellini, and L. Bottaccioli, "Games: A general-purpose architectural model for multi-energy system engineering applications," in *2020 IEEE 44th Annual Computers, Software, and Applications Conference (COMPSAC)*, 2020, pp. 1405–1410.

[29] P. T. Eugster, P. A. Felber, R. Guerraoui, and A.-M. Kermarrec, "The many faces of publish/subscribe," *ACM computing surveys (CSUR)*, vol. 35, no. 2, pp. 114–131, 2003.

[30] Dassault Systèmes, "FMPy Python Library," https://github.com/CATIA-Systems/FMPy.

[31] Weather Underground, "Weather Forecast & Reports, Long Range & Local," https://www.wunderground.com.

[32] L. Bottaccioli, E. Patti, E. Macii, and A. Acquaviva, "Gis-based software infrastructure to model pv generation in fine-grained spatio-temporal domain," *IEEE Systems Journal*, vol. 12, no. 3, pp. 2832–2841, 2017.

[33] L. Bottaccioli, S. Di Cataldo, A. Acquaviva, and E. Patti, "Realistic multi-scale modeling of household electricity behaviors," *IEEE Access*, vol. 7, pp. 2467–2489, 2018.

Daniele Salvatore Schiera received the B.Sc. degree in Energy Engineering from the Università degli Studi di Palermo, Italy, in 2016, and the M.S. degree in Energy and Nuclear Engineering from the Politecnico di Torino, Italy, in 2018. He has been with Politecnico di Torino, Italy, since 2018, where he is currently a Ph.D. Candidate with the Department of Energy and an active member of the Energy Center Lab. His main research interests focus on the development of a holistic methodology and tools for the deployment of sustainable integrated energy systems, with particular attention on the interoperability of sub-models at different functional layers, dimensions and domains, allowing to set up and solve different urban energy transition scenarios through effective urban energy planning and operation.

Luca Barbierato graduated in Telecommunication Engineering in 2013. During his studies, he visited the University of Seville and the Department of Image Processing and 3D Innovations of the HHI Fraunhofer in Berlin. He was a collaborator of SWARM Joint Open Lab of Telecom Italia. He was involved in the research activities of FP7 INTrEPID Project, focusing on the development of novel technologies for the energy optimization in residential buildings. In February 2016 he joined the Department of Computer and Control Engineering at Politecnico di Torino as a Research Assistant. In November 2018 he joined as Ph.D. Candidate the Energy Center Lab. His main research interests are Energy Efficiency and Smart Cities with particular attention to the design of co-simulation platform for Multi-Energy System. He is a Student Member of the IEEE.

Andrea Lanzini received the M.S. in Energy and Nuclear Engineering in 2007 and his Ph.D. in Energetics in 2011. He is presently a tenure-track assistant professor at the Energy Department of Politecnico di Torino (Italy) where is a member of the Thermochemical and Electrochemical Systems Group. He is a member of the Energy Center of Politecnico di Torino. He conducts research in the field of high temperature fuel cells, carbon capture and re-use technologies and integrated energy systems analysis. He teaches classes in Sustainable Use of Energy, Polygeneration and Advanced Energy Systems, and Thermal Design and Optimization.

Romano Borchiellini graduated in Mechanical Engineering in 1983 at Politecnico di Torino, Italy. Since February 2001, he is Full Professor at the Department of Energy of Politecnico di Torino of thermodynamics, heat and mass transfer, and HVAC system. Since 2012, he is vice-president of the Italian Committee for Tunnel Management of the World Road Association (PIARC). Between July 2014 and December 2018, he was the President of SiTI (Higher Institute on Territorial Systems for Innovation). Since December 2014, he has been chairing the Advisory Board of the Energy Center and since 2018 he is the Coordinator of the Energy Center Initiative of Politecnico di Torino.

Enrico Pons (M'15) received the master degree in electrical engineering in 2004 and the Ph.D. in industrial safety and risk analysis in 2008 from Politecnico di Torino, Torino, Italy. He is currently associate professor at Politecnico di Torino, where he teaches electrical installations. His research activities include complexity in energy systems, power systems security, complex networks methodologies for the analysis of power systems vulnerability, real-time simulation of power systems, integration of renewable generation in distribution networks, smart metering, traction electrification systems and electrical safety.

Ettore Bompard (M'99) received the Ph.D. in electrical engineering from Politecnico di Torino, Turin (Italy). He has been visiting assistant professors at the University of Illinois, Urbana-Champaign (US) (Apr.-Jul. 1999, Jan.-Oct. 2000) and Senior scientist at the Energy Security Unit of the JRC for Smart Electricity Systems and Interoperability, Institute for Energy and Transport, Power System and Critical Infrastructure in Petten (NL) (Oct. 2012, Sept. 2014). He is Professor of Power Systems at the Department of energy of Politecnico di Torino and scientific director of the Energy Security Transition Lab @Energy Center, Torino. His research interests include electricity markets analysis and simulation, smart grids design and modelling, power system vulnerability assessment and security management, energy security, "science-based" support to policy decision making and data analytics applications to power systems.

Edoardo Patti (M'16) is Assistant Professor at Politecnico di Torino (Italy). He received both M.Sc. and Ph.D. degrees in Computer Engineering from the same university in 2010 and 2014, respectively. His main research interests concern Ubiquitous Computing, Internet of Things and Smart Systems applications. He is a Member of the IEEE.

Enrico Macii (SM'02-F'05) is a Full Professor of Computer Engineering with the Politecnico di Torino, Torino, Italy. He holds a Laurea degree in electrical engineering from the Politecnico di Torino, a Laurea degree in computer science from the Università di Torino, Turin, and a PhD degree in computer engineering from the Politecnico di Torino. His research interests are in the design of electronic digital circuits and systems, with a particular emphasis on low-power consumption aspects. He is a Fellow of the IEEE.

Lorenzo Bottaccioli is Assistant Professor at the Energy Center Lab of Politecnico di Torino (Italy). He received the B.Sc. in Mechanical Engineering in 2011 at University of Perugia; the M.Sc. in Energetic and Nuclear Engineering in 2014 at Politecnico di Torino; and the PhD, cum laude, in Computer Engineering at Politecnico di Torino. His main research interests concern Smart Energy, City and Communities with focus on software solutions for planning, analysing and optimising smart energy system, and for spatial representation of energy information.