Cluster Computing (2023) 26:719–743
https://doi.org/10.1007/s10586-022-03797-8

Modeling and simulation of smart grid-aware edge computing federations

Román Cárdenas[1,2] · Patricia Arroba[1] · José L. Risco-Martín[3] · José M. Moya[1]

Received: 21 December 2021 / Revised: 18 October 2022 / Accepted: 25 October 2022 / Published online: 11 November 2022

Abstract
Compute-intensive Internet of Things (IoTs) applications have led to the edge computing paradigm. Edge computing decentralizes the IT infrastructure in multiple edge data centers (EDCs) across the access networks to reduce latency and network congestion. Edge computing can benefit significantly from different aspects of smart grids to achieve lower energy consumption and greater resilience to electricity price fluctuations. This paper presents a modeling, simulation, and optimization (M&S&O) framework for analyzing and dimensioning smart grid-aware edge computing federations. This tool integrates aspects of a consumer-centric smart grid model to the resource management policies of the EDCs. To illustrate the benefits of this tool, we show a realistic case study for optimizing the energy consumption and operational expenses of an edge computing federation that provides service to a driver assistance IoT application. Results show that this approach can reduce the daily energy consumption by 20.3% and the electricity budget by 30.3%.

Keywords Edge computing · IoT · MBSE · Simulation · Smart grid

1 Introduction

The Internet of Things (IoTs) is leading to a disruption in multiple business sectors. IoT technologies capture and process the actions of users and infrastructures to improve quality of life, safety, and resource management (e.g., autonomous driving or smart cities). These technologies are growing, and 47% of organizations plan to increase their investments in this field within the next few years [1].

✉ Román Cárdenas
r.cardenas@upm.es

Patricia Arroba
p.arroba@upm.es

José L. Risco-Martín
jlrisco@ucm.es

José M. Moya
m.moya@upm.es

[1] Universidad Politécnica de Madrid, Avenida Complutense 30, 28040 Madrid, Spain

[2] Carleton University, 1125 Colonel By Drive, Ottawa, ON K1S 5B6, Canada

[3] Universidad Complutense de Madrid, C/ Prof. José García Santesmases 9, 28040 Madrid, Spain

IoT applications progressively increased the complexity of their ecosystems, sharing and self-managing their resources autonomously to achieve a common goal. However, IoT devices typically present limited storage and processing capabilities due to intrinsic limitations (e.g., battery lifetime and production cost). IoT services that require intensive computation and mass data warehousing may need additional infrastructure for effective real-time processing of a large volume of information from geographically distributed sources. Integrating cloud technologies with IoT has succeeded in overcoming these limitations [2]. IoT applications use these resources to improve their services (e.g., advanced visualization tools or over-the-air firmware updates). However, some IoT applications present strict Quality of Service (QoS) requirements (e.g., low latency and rapid mobility). For those applications where response time is critical, the centralized approach of cloud data centers presents some limitations. For example, IoT applications using cloud services may experience significant communication delays. Furthermore, if IoT applications send heavy data streams to clouds from multiple devices, the core network of the Internet service provider (ISP) would be congested, degrading the overall QoS and

bandwidth usage of any service that requires a connection to the Internet [3].

The edge computing paradigm arises as a solution for these limitations by extending the concept of cloud computing to the network edge. Edge computing decentralizes the computing resources and distributes them geographically closer to IoT devices to significantly reduce latency and network costs [4]. There is no consensus on where edge computing is. Usually, this depends on the vendor's view. For instance, some IoT manufacturers claim that edge computing starts with the device itself. We follow the ISP perspective, which devises edge computing as a continuum of edge data centers (EDCs) (also considered micro data centers, MDCs) distributed from the ISP access networks to the public cloud to provide a drastic latency and network congestion reduction [5]. The assets of each EDC are sized to meet the demand in a given small area. This characteristic makes it possible to define more efficient resource utilization techniques because the system load corresponding to each service area is more predictable than in a centralized solution such as cloud computing [6]. Regarding energy consumption, edge computing infrastructures can significantly benefit from different aspects of smart grids. A smart grid is an electricity network that employs information and communications technologies (ICTs) to monitor and manage electricity usage to optimize production, transmission, and distribution [7]. The EDCs can work as a federation to reduce energy costs and improve resource management strategies [8]. Furthermore, EDCs can adopt more efficient self-consumption policies to transition to low-carbon systems [9]. Additionally, the edge infrastructure would benefit from other advantages of smart grids, such as greater resilience against energy price fluctuations [10].

Complex systems such as smart grid-aware edge computing federations require multi-domain solutions. In these scenarios, one principal source of failure is communication between development teams. Model-based systems engineering (MBSE) methodologies overcome this vulnerability by establishing modeling as the central entity for information exchange [11]. MBSE eases the integration of modeling and simulation (M&S) tools into the system development process to explore and validate the complex system under study, providing information about potential technical risks and functionality of the solution while saving expenses [12].

Here we present a modeling, simulation, and optimization (M&S&O) framework for the analysis and dimensioning of edge computing infrastructures connected to smart grids. This tool integrates smart grids into the resource management process of the EDCs that comprise the edge computing federation. This integration can help to reduce the overall energy consumption and operational expenses of edge computing infrastructures, enabling the deployment of IoT applications with a smaller carbon footprint and increasing the viability of investments in future edge computing infrastructures. We have developed the proposed framework relying on the Discrete EVent System specification (DEVS) formalism [13] and the principles of MBSE that ensure a logical, robust, and reliable incremental design. In particular, the contributions of our research are the following:

- We extend our edge computing model presented in previous publications [14, 15] to integrate cooling infrastructures and support dynamic and more advanced resource management policies.
- We also expand the model to support a decentralized load balancing protocol to map new loads to one of the EDCs comprising the edge computing federation. This new approach includes cloud and edge computing infrastructures working together to improve their overall QoS while responding to unusually high demand.
- We introduce a consumer-centric smart grid model and combine it with the proposed edge computing model. EDCs incorporate current electricity pricing, on-site renewable energy generation, and energy storage to define more efficient federated resource management strategies.
- We develop an M&S&O framework that follows the presented models to assist in dimensioning edge computing federations connected to smart grids.
- We provide a realistic case study to illustrate how this M&S&O framework can assess the efficiency of smart grid-aware edge computing federations for an advanced driver assistance system (ADAS) application. We show that it is possible to reduce the daily energy consumption and cost by 20.3% and 30.3%, respectively.

The rest of the paper is organized as follows. First, we discuss related work in Sect. 2. Section 3 describes the extended edge computing model proposed in this research. In Sect. 4. We also illustrate how we integrate it into the edge computing model. Section 5 presents a use case scenario to demonstrate how the proposed M&S&O framework can assist in dimensioning different aspects of smart grid-aware edge computing federations. Finally, we conclude in Sect. 6.

2 Related work

We provide an overview of works focused on modeling and simulating edge computing infrastructure for computation offloading tasks and smart grids. We also discuss approaches for integrating both fields.

2.1 Edge computing infrastructures

Edge computing for IoT applications is a research topic with several publications focused on improving the performance and energy efficiency of the infrastructure [16]. Some of these publications propose new resource management techniques to reduce energy consumption while still meeting the required QoS using different techniques (e.g., mathematical models [17], deep learning-based approaches [18], or M&S methods [19]). For example, Zhang et al. present ApproxECIoT [20], a novel edge computing architecture to process real-time data streams of IoT applications. The IoT nodes comprising a wireless sensor network (WSN) send data streams to EDCs. As the computing and memory resources of the EDCs are limited, they sample the incoming data streams and apply approximate numeric methods to compute partial results. Then, they forward these partial results to a cloud data center for further processing. The cloud data center aggregates all the partial outcomes and checks if the accuracy of the final result complies with the expected QoS. If not, the cloud service readjusts the sampling rate of the EDCs. The proposal of Dong et al. [21] is of particular interest to this paper. The authors use Digital Twins (DTs) to optimize user association and EDC resource allocation depending on the QoS requirements. DTs are virtual replicas of a physical entity that capture the entities and dynamics of the system under study [22]. DTs are extensively used together with simulation and data analytics tools to integrate physical and virtual data. They enable the exploration of multiple plausible scenarios for optimizing the system's performance [23].

Other works consider cooling energy to study the energy efficiency of edge computing infrastructures [24]. The energy efficiency of data centers is typically measured with the power usage effectiveness (PUE) metric [25]:

$$\mathrm{PUE} = \frac{\text{Total facility power}}{\text{IT power}}. \quad (1)$$

Energy consumed by the information technology (IT) components of the EDCs eventually becomes heat. The cooling infrastructure dissipates this heat to avoid potential damage to the IT elements. Depending on the cooling system, the cooling power may suppose a significant portion of the total facility power, degrading the EDC's PUE. Hyperscale operators can place their data centers in cool climates to use outside air with advanced cooling technologies to obtain PUEs near 1.1 [26]. However, cooling supposes up to 40% of the total power consumption in average air-cooled data centers, leading to a mean PUE of 1.67 [27]. As EDCs are in access networks, they may present additional limitations compared to cloud systems (e.g., limited space and warmer climates). Thus, using efficient cooling systems is a significant concern. Two-phase immersion cooling technologies can mitigate these limitations of edge computing scenarios [28]. Two-phase immersion cooling systems immerse the IT equipment in a close bath full of a coolant fluid with very high heat flux. The heat produced by the IT infrastructure evaporates the coolant. Then, a condenser sets the coolant's vapor to liquid phase again. A secondary working fluid (usually water) captures the heat from the condenser. Usually, the setpoint temperature of the coolant fluid is around 60 °C, enabling a cooling power reduction of up to 95% regardless of the climate. This allows us to increase the power density and reduce the physical footprint up to 10 times [29].

M&S tools are necessary to define effective deployments of edge computing architectures, as they provide in-depth virtual analysis of such complex systems without incurring high costs. Multiple software tools focus on modeling and simulating IoT and edge computing solutions. Thus, they allow us to explore numerous scenarios to design more efficient architectures. We describe next some of the most popular simulators for studying edge computing infrastructures. FogNetSim++ [30] studies connectivity features of edge computing infrastructures. YAFS focuses on the analysis of dependent applications and their relationships [31]. Alternatively, FogTorchPi [32] is a simulator based on the Monte Carlo method to optimize the QoS of IoT applications. The iFogSim simulator [33] considers sensors and actuators to model IoT devices. EdgeCloudSim [34] integrates a simple mobility model for end-users. Finally, IOTSim [35] optimizes IoT services using the MapReduce algorithm. Table 1 shows a brief comparison of the edge computing simulators previously mentioned. Mercury [14] corresponds to the M&S&O framework presented in this paper. As our research focuses on edge computing infrastructures, Mercury had not previously considered cloud computing facilities. However, we integrate a simplified cloud computing model to Mercury for this work. Section 3 describes the proposed simplified cloud model.

2.2 Smart grids

The smart grid paradigm embraces multiple research fields. The literature divides smart grid conceptual models into seven domains: customer, markets, service providers, operations, generation, transmission, and distribution [36]. While we must evaluate each field separately, they have shared requirements (e.g., communication protocols and security) and often interact to enable smart grid functionalities [37]. The customer domain embraces the end-users of electricity. In smart grids, customers may also generate and store energy for better energy management (e.g., demand peak and overall cost reduction [38]). In this

Table 1 Edge computing simulators comparison

Research	Cloud	Mobility	Delay	IT power	Cooling
FogNetSim++ [30]	✔	✔	✔	✗	✗
YAFS [31]	✔	✔	✔	✗	✗
FogTorchPi [32]	✔	✗	✔	✗	✗
iFogSim [33]	✔	✗	✔	✔	✗
EdgeCloudSim [34]	✔	✔	✔	✗	✗
IOTSim [35]	✔	✗	✔	✗	✗
Mercury [14]	✔	✔	✔	✔	✔

context, state-of-the-art works propose different methods to describe and optimize smart grid customers' energy management (e.g., mathematical models for optimal load shifting [39] or machine learning behavioral models [40]). These models integrate pricing data to enhance the consumers' energy consumption. Electricity price changes depend significantly on customer base, country, or locality within a given country [41]. Therefore, dynamic pricing schemes present higher elasticity and better resource utilization than static approaches. However, they are exposed to market volatility, which may affect the end-user cost negatively.

M&S tools are extensively used for designing and validating the components that comprise smart grids. Vaubourg et al. [42] present a co-simulation approach to model the operations, generation, transmission, and distribution domains of smart grids. GECO [43] is an event-driven simulator with configurable time precision that provides good scalability for large systems. Alternatively, Mosaik [44] is an M&S framework that offers a flexible, composable interface for adapting custom consumer/producer models. However, the discrete-time nature of Mosaik leads to a substantial performance penalty when time accuracy is required. Table 2 compares these simulators with the new version of Mercury [14] presented in this paper. Note that the smart grid model implemented by Mercury is focused on the customer side and does not implement aspects that belong to other domains.

2.3 Edge computing and smart grids

Some related works study edge computing infrastructures and smart grids together. However, these studies focus on smart grid scenarios with edge computing as a service for the communication, monitoring, and analysis of the infrastructure of these smart grids [45]. To the best of our knowledge, no previous works analyze generic edge computing scenarios that integrate smart grids as a service for optimizing the energy consumption of the edge computing infrastructure while providing service to different IoT applications. Huang et al. [46] propose a framework for real-time monitoring of smart grids using edge computing facilities, increasing the monitoring rate by 10 times and achieving 85% less communication delay than centralized cloud-based systems. Liu et al. [47] present an IoT-based solution with edge computing that improves the resource management of smart grids in smart cities with deep reinforcement learning. Also, Gai et al. [48] propose an edge computing system for securing smart grids using blockchain technology.

Here we present a formal model for smart grid-aware edge computing federations. The model integrates cooling systems and DTs of the edge computing facilities. It also considers aspects of the smart grid consumer domain (e.g., energy generation and storage). The presented model describes edge computing resource management-related features in detail. In contrast, other parts of the scenario (e.g., physical interfaces, cloud facilities, or energy distribution infrastructures) are defined with less detail. In this way, the proposed model presents a balance between model complexity and simulation performance. The hierarchical

Table 2 Smart grid simulators comparison

Research	Communication	Distribution	Price	Generation	Storage
Vaubourg et al. [42]	✔	✔	✗	✔[a]	✗
GECO [43]	✔	✔	✗	✔[a]	✗
Mosaik [44]	✗	✔	✗	✔	✔
Mercury [14]	✔[b]	✗	✔	✔[b]	✔[b]

[a]Only for energy providers

[b]Only for smart grid customers

and modular approach of the DEVS formalism eases the integration of models with various degrees of detail to capture the interrelationships between different complex systems. Mercury, our M&S&O framework [14], implements the presented model as an extension. This new version enables the optimization of edge computing facilities connected to smart grids, reducing the carbon footprint of IoT services that benefit from this infrastructure. Mercury incorporates a multi-faceted modeling approach that allows users to select different degrees of complexity in the computational model [49]. Multi-faceted models are particularly useful for optimizing parts of the system under study. When optimizing a given feature, we run several simulations using a model with a fine-grained level of detail in the elements of the system that affect this feature the most but a high-level model of the rest of the system. Once optimized, the scenario is extensively validated using the detailed model of the whole system with less effort since the design space has already been explored and exploited. This tool is publicly available in Mercury's GitHub repository [50].

3 Edge computing model

The proposed model is an extension of our previous work [15]. Specifically, it includes cooling systems for EDCs and DTs of EDCs' components to implement more sophisticated resource management policies. It also adds support for decentralized dynamic resource management policies depending on the current state of the scenario. Furthermore, it captures the behavior of cloud services to support demand peaks in case the edge computing infrastructure runs out of resources. This model, depicted in Fig. 1, has been implemented in the Mercury M&S&O framework [14].

Most of the elements of the model are located in a 5G radio access network (RAN) of an ISP. IoT devices correspond to 5G user equipment (UE, e.g., smartphones or connected vehicles). These devices run one or more services that monitor the users and their environment. We refer to the set of all the UE in the scenario as **UE**. UE nodes offload the computation of the data to nearby EDCs comprising specialized hardware resources for providing computing to the users. Alternatively, access points (APs) establish radio communication links between UE devices and the rest of the scenario. All the APs of the model comprise the set **AP**. At time t, a given UE $\in$ **UE** is connected to the AP with the best signal quality in the location of the UE, $\text{ap}_{\text{UE}}(t) \in \textbf{AP}$. This AP allows the UE to use less energy for transmitting data to the network. Therefore, handovers from one AP to another may occur as UE nodes move across the scenario. APs re-direct service-related messages from UE to the EDCs via the crosshaul network. This network comprises optical fiber communication links that interconnect EDCs, APs, and the core network of the ISP for providing Internet connectivity to all the elements connected to the RAN.

Computation offloading follows a Function-as-a-Service (FaaS) fashion in this model. When UE nodes request to open a new service session, EDCs inspect their available resources and reserve those required by the service for granting computation offloading while the session is active. Once UE nodes close the session, held resources are freed and available for new eventual UE service session requests. All the EDCs in a RAN work in a federated manner and share their current state. Usually, UE requests are processed by the closest EDC in the scenario. However, if one of the EDCs runs out of resources due to an abnormal demand peak, it will forward incoming requests to the second-best EDC of the federation. Furthermore, EDCs can forward requests to the public cloud in the event of congestion in all the EDCs. While clients whose requests are sent to the cloud experience higher delays than usual, they will not have to wait until one or more EDCs become available.

3.1 Edge data centers

This section presents the proposed model for describing the behavior of EDCs. Figure 2 represents a schematic of the proposed model.First, the EDC interface hides the complexity of the EDC and shares with the rest of the scenario an alternative version of the EDC's current state with less detail. It also receives new computation offloading requests and decides which EDC of the federation is in charge of processing it. If all the EDCs are busy, it forwards the request to the cloud. Alternatively, processing units (PUs) are the IT resources of the EDC, and the cooler dissipates the heat produced by the PUs to avoid any potential damage. The resource manager is in charge of matching new session requests to one of the PUs in the EDC. The resource manager incorporates a DT of the IT and cooling resources to make more accurate predictions of the EDC's next state and optimize its performance. The demand estimator monitors incoming requests and profiles the current service demand. It may incorporate additional information to estimate the future need for EDC computing resources. Finally, the policy manager considers the current state of the EDC, the current demand, and the future demand estimation to change dynamically the policies used by the resource manager to dispatch new sessions to PUs. Next, we describe the behavior of each of the subcomponents comprising an EDC.

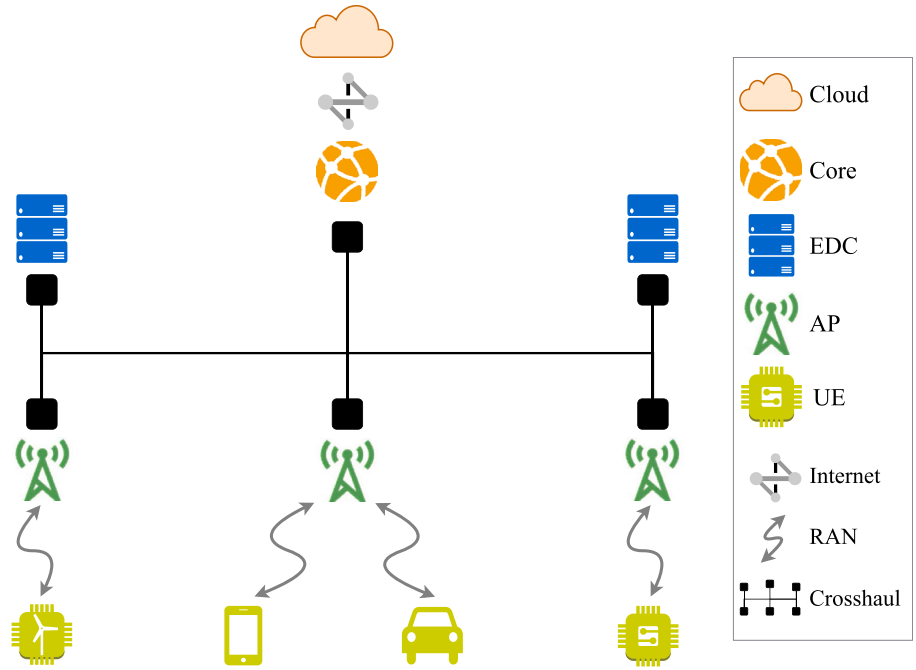

Fig. 1 Proposed edge/cloud model

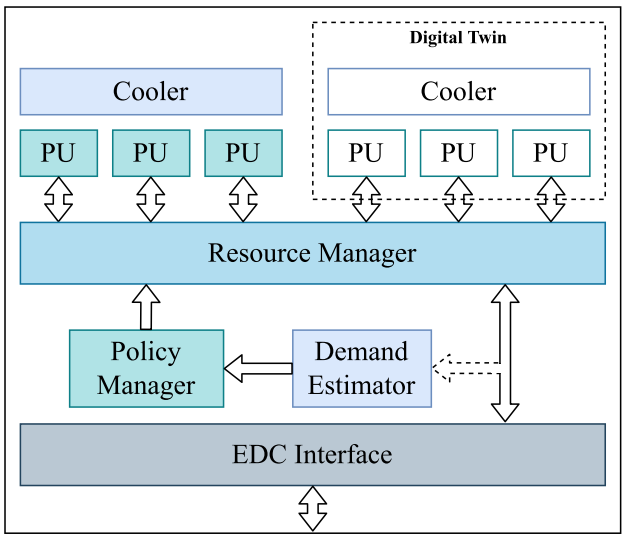

Fig. 2 Edge data center model

3.1.1 Processing units

PUs are the IT resources of the EDC. A PU can start a new IoT service session, process requests of active sessions, or close already active sessions. When an IoT service SRV requests to open a new session, PUs reserve a fraction of their computing resources, $U_{\text{PU}}^{\text{SRV}}$, for this session. Thus, IoT services with active sessions are guaranteed to have the needed resources as long as their session is still active. When a session is closed, these resources become available again for any eventual new session. PUs can only open new sessions if they have enough free resources, as the utilization factor of a PU at time t, $u_{\text{PU}}(t)$, cannot be greater than 1:

$$u_{\text{PU}}(t) = \sum_{\text{SRV} \in \mathbf{srv}_{\text{PU}}(t)} U_{\text{PU}}^{\text{SRV}} \leq 1, \tag{2}$$

where $\mathbf{srv}_{\text{PU}}(t)$ is the set of active sessions on the PU at time t.

The power consumption (in W) of a PU at time t, $p_{\text{PU}}(t)$, depends on its current utilization factor, architecture, and hardware specifications. When a PU is not hosting any service session [i.e., $\mathbf{srv}_{\text{PU}}(t) = \emptyset$], it can be turned off to save energy. As soon as a service requests to start a new session, the PU switches on and creates the requested session. However, switching on and off a PU introduces additional delays ($T_{\text{PU}}^{\text{on}}$ and $T_{\text{PU}}^{\text{off}}$, respectively). These delays may impact negatively on the latency. To avoid poor QoS, a PU can remain on hot standby. Hot standby PUs persist idle even when they do not host any session. While their power consumption is higher, these PUs are ready to open new sessions, reducing the response time significantly. The model implemented in Mercury allows you to define custom power consumption functions for each PU. This feature enables the heterogeneous IT equipment in each EDC. The IT power consumption of an EDC at time t, $p_{\text{EDC}}^{\text{IT}}(t)$, corresponds to the sum of the power consumption of all its PUs:

$$p_{\text{EDC}}^{\text{IT}}(t) = \sum_{\text{PU} \in \mathbf{PU}_{\text{EDC}}} p_{\text{PU}}(t), \tag{3}$$

where $\mathbf{PU}_{\text{EDC}}$ is the set of all the PUs within an EDC.

3.1.2 Cooler

The cooler dissipates the heat produced by the PUs to avoid any potential damage. The cooling power at time t, $p^{\text{cool}}_{\text{EDC}}(t)$, depends on $p^{\text{IT}}_{\text{EDC}}(t)$ and the cooling technology used. Even though Mercury supports different cooling technologies, we model $p^{\text{cool}}_{\text{EDC}}(t)$ as a pump-driven two-phase immersion cooler in this research. Figure 3 shows a schematic of the proposed cooling system.

The pump that drives the secondary working fluid is the only element of this cooling system that requires electric power to function. The pump power consumption depends on the flow rate of the secondary working fluid. Equation (4) shows how to obtain the flow rate:

$$\Phi_{\text{EDC}}(t) = \frac{1}{277.78} \cdot \frac{p^{\text{IT}}_{\text{EDC}}(t)}{\rho \cdot C_p \cdot \Delta T} \ \left(\text{m}^3\,\text{h}^{-1}\right), \tag{4}$$

where ρ is the density of the secondary working liquid (in $\text{g}\,\text{cm}^{-3}$), C_p is its specific heat capacity (in $\text{J}\ \text{g}^{-1}\ \text{K}^{-1}$), and ΔT corresponds to the difference between the outlet and inlet temperature of the fluid (in K). We divide by 277.78 to transform the flow rate units from $\text{cm}^3\,\text{s}^{-1}$ to $\text{m}^3\,\text{h}^{-1}$. The relation between $p^{\text{cool}}_{\text{EDC}}(t)$ and $\Phi_{\text{EDC}}(t)$ depends on the characteristics of the pump used.

3.1.3 EDC Digital Twin

Every EDC has a DT of its IT and cooling infrastructure. Resource management tasks can use this DT to explore the search space and evaluate the effect on the EDC state for optimizing its performance. For example, we could forecast the impact of starting a new service session in a given PU on its power consumption. First, the PU's DT would predict the resulting PU's utilization factor:

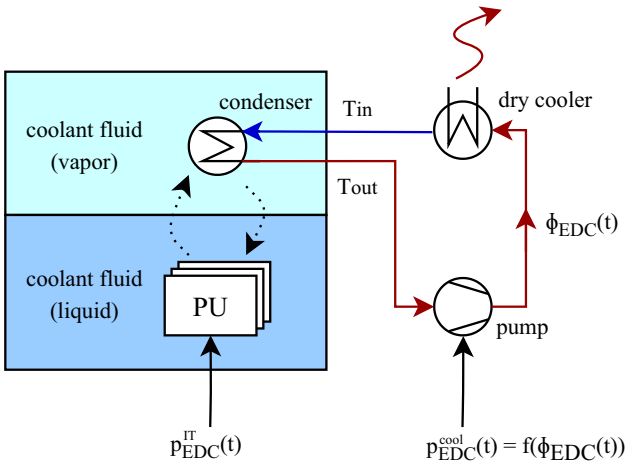

Fig. 3 Schematic of pump-driven two phase cooling system

$$u'_{\text{PU}}(\text{SRV}, t) = u_{\text{PU}}(t) + U^{\text{SRV}}_{\text{PU}}. \tag{5}$$

Then, the DT of the PU would also evaluate the resulting power consumption if the PU started this new service session:

$$p'_{\text{PU}}(\text{SRV}, t) = p_{\text{PU}}(t)|_{u_{\text{PU}}(t)=u'_{\text{PU}}(\text{SRV,t})}. \tag{6}$$

3.1.4 Resource manager

The main functionality of this module is matching new session requests to one of the PUs in the EDC. When a new service SRV requests to open a session in the EDC, the resource manager forwards this request to the PU selected by the EDC dispatching function, $\text{disp}_{\text{EDC}}(\text{SRV}, t)$. Mercury allows defining multiple dispatching functions depending on the use case (e.g., minimizing the response time or dividing the PUs resource utilization evenly). In this research, we use the current state of the EDC and its DT for mapping new sessions to the PU that would experience the minimum IT power consumption increase:

$$\begin{aligned} \text{disp}_{\text{EDC}}(\text{SRV}, t) = & \underset{\text{PU} \in \mathbf{PU}_{\text{EDC}}}{\arg\min}\ p'_{\text{PU}}(\text{SRV}, t) - p_{\text{PU}}(t) \\ & \text{s.t. } u'_{\text{PU}}(\text{SRV}, t) \leq 1, \end{aligned} \tag{7}$$

where $p_{\text{PU}}(t)$ is the power of the PU, and $p'_{\text{PU}}(\text{SRV}, t)$ and $u'_{\text{PU}}(\text{SRV}, t)$ are the power consumption and utilization factor predicted by the DT. Additionally, the resource manager determines which PUs shall remain on hot standby depending on the hot standby function, $\text{stdby}_{\text{EDC}}(t) \subseteq \mathbf{PU}_{\text{EDC}}$. Note that the policy manager can change $\text{disp}_{\text{EDC}}(\text{SRV}, t)$ and $\text{stdby}_{\text{EDC}}(t)$ depending on the state of the scenario.

3.1.5 Demand estimator

The demand estimator module continuously monitors the incoming service requests to profile the current service demand, $\text{prof}_{\text{EDC,SRV}}(t)$. It also implements a demand estimation model for every service, $\text{estim}_{\text{EDC,SRV}}(t)$, to forecast the future need for the resources of the EDC. The demand profiling and estimation are forwarded to the EDC policy manager for further evaluation.

3.1.6 Policy manager

The edge computing infrastructure can adapt its resource management policies depending on the current status of the scenario. Service demand on edge computing varies significantly over time in response to the user activity in the region. For instance, service demand may be lower at 3:00 AM, whereas a demand peak may be reported at 7:00 AM.

EDCs resource management policies must change over time to provide an efficient solution to these variations. The policy manager keeps track of the current state of the EDC and the future demand estimation. From this information, the policy manager can apply reactive, proactive, or hybrid techniques to automatically change the dispatching and hot standby policies and adjust them according to the scenario state.

3.1.7 EDC interface

The EDC interface hides the complexity of the EDC and shares with the rest of the EDCs an alternative version of the EDC's current state with less detail. It represents all the active service sessions on an EDC at time t, $\mathbf{srv}_{\mathrm{EDC}}(t)$, as the union of all the active service sessions of all the EDC's PUs:

$$\mathbf{srv}_{\mathrm{EDC}}(t) = \bigcup_{\mathrm{PU} \in \mathbf{PU}_{\mathrm{EDC}}} \mathbf{srv}_{\mathrm{PU}}(t). \tag{8}$$

The overall resource utilization of the EDC corresponds to the mean resource utilization of its PUs:

$$u_{\mathrm{EDC}}(t) = \frac{\sum_{\mathrm{PU} \in \mathbf{PU}_{\mathrm{EDC}}} u_{\mathrm{PU}}(t)}{|\mathbf{PU}_{\mathrm{EDC}}|} \leq 1. \tag{9}$$

It also computes the hot standby utilization of the EDC, $u^{\mathrm{stdby}}_{\mathrm{EDC}}(t)$. The hot standby utilization estimates how many resources are currently available for computation offloading services:

$$u^{\mathrm{stdby}}_{\mathrm{EDC}}(t) = \frac{\sum_{\mathrm{PU} \in \mathrm{stdby}_{\mathrm{EDC}}(\mathrm{t})} u_{\mathrm{PU}}(t)}{|\mathrm{stdby}_{\mathrm{EDC}}(t)|} \leq 1. \tag{10}$$

The EDC interface shares the IT and cooling power consumption with the scenario. It also defines the EDC overall power demand, $p^{\mathrm{demand}}_{\mathrm{EDC}}(t)$, as the sum of the IT and cooling power. Finally, the EDC interface computes the PUE of the EDC, $\mathrm{pue}_{\mathrm{EDC}}(t)$:

$$\mathrm{pue}_{\mathrm{EDC}}(t) = \frac{p^{\mathrm{demand}}_{\mathrm{EDC}}(t)}{p^{\mathrm{IT}}_{\mathrm{EDC}}(t)} = 1 + \frac{p^{\mathrm{cool}}_{\mathrm{EDC}}(t)}{p^{\mathrm{IT}}_{\mathrm{EDC}}(t)}. \tag{11}$$

When a UE requests to open a new session, the AP that provides connectivity to the UE forwards the request to the nearest EDC. The interface of this EDC is in charge of determining which EDC should create this session. First, the EDC interface specifies the candidate EDCs of the RAN with enough resources to host the new session:

$$\mathbf{edc}_{\mathrm{candidate}}(t) = \left\{ \mathrm{EDC} \in \mathbf{EDC} | u^{\mathrm{stdby}}_{\mathrm{EDC}}(t) < 1 \right\}. \tag{12}$$

Note that we only consider the hot standby utilization of the EDCs. In this way, even if an EDC has numerous computing resources, but none of them is on hot standby, its perceived utilization would be 1. Therefore, this EDC would not be considered a valid candidate. If there are no candidate EDCs [i.e., $\mathbf{edc}_{\mathrm{candidate}}(t) = \emptyset$], it forwards the request to the public cloud. Otherwise, it dispatches the new session request to one of the candidate EDCs. The selected EDC depends on the EDC mapping policy, $\mathrm{edc}_{\mathrm{EDC}}(t)$. The proposed model allows multiple EDC mapping policies depending on the use case. In this research, we favor the nearest candidate EDC:

$$\mathrm{edc}_{\mathrm{EDC}}(t) = \underset{\mathrm{edc} \in \mathbf{edc}_{\mathrm{candidate}}(t)}{\arg\min} \; d(\mathrm{edc}, \mathrm{EDC}). \tag{13}$$

This EDC mapping policy enhances computation, reduces the overall latency, and avoids communication bottlenecks and general system failures. If the selected EDC corresponds to the one computing the mapping, the open session request is forwarded to the EDC's resource manager. Otherwise, the request is sent to the selected EDC. In the proposed model, requests forwarded from other EDCs are directly forwarded to the resource manager. In this way, we avoid requests starvation (i.e., requests forwarded from one EDC to another indefinitely). Figure 4 depicts a flowchart of the EDC mapping process enforced by the EDC interface.

3.2 Cloud

The presented edge computing infrastructure shows multiple advantages (e.g., low latency and less overall network congestion). However, it also introduces new challenges. For instance, computing resources are not unlimited, as EDCs are dimensioned to satisfy the demand of a limited area. Thus, an abnormal demand peak may lead to resource shortages in the edge infrastructure. Furthermore, due to unexpected features, the proposed solution must be

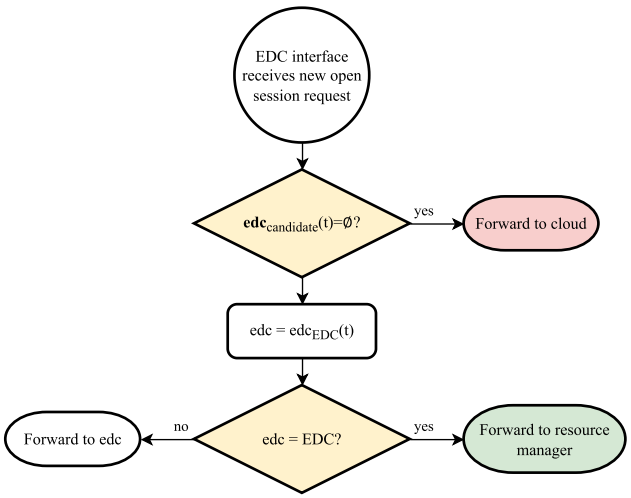

Fig. 4 Workflow diagram of the EDC mapping process

resilient to potential EDC shutdowns. A collaboration between the edge computing infrastructure and cloud facilities can alleviate the negative effect of these problems.

Our research focuses on resource management techniques for edge computing facilities, and we had not considered the cloud infrastructure in previous works. Here we introduce a simple cloud model that serves as a backup for situations where the available edge computing resources are scarce. This cloud model represents a simple collaboration between EDCs and clouds. In future work, we will explore alternative collaboration patterns to define more complex applications that use both edge and cloud resources.

We divide the proposed cloud model into the Internet connection and the cloud facility. The Internet connection is modeled as a delay buffer that retains messages before forwarding them to their corresponding receiver. The Internet delay for a message *MSG* is computed as follows:

$$\text{delay}(\text{MSG}) = t_{\text{prop}} + \frac{\text{size}(\text{MSG})}{v_{\text{trans}}} \text{ (s)}, \tag{14}$$

where t_{prop} is the propagation time from the RAN to the public cloud (in s), size(MSG) is the size of the message (in b), and v_{trans} is the mean transmission speed between the cloud and the RAN (in b s^{-1}).

Alternatively, the cloud facility has virtually unlimited computing resources. In contrast with the EDCs, the cloud can simultaneously process any number of service sessions. Once a service session is opened, the cloud will take $T^{\text{proc}}_{\text{cloud,SRV}}$ seconds before sending a response to the client.

Table 3 resumes the most relevant edge computing model-related parameters previously discussed.

4 Consumer-centric smart grid model

Figure 5 shows the proposed consumer-centric smart grid model.This model comprises one energy provider PROVR and a set of smart grid consumers **CONSR**. Each consumer CONSR $\in$ **CONSR** may use energy from the energy provider to satisfy its energy needs. In this paper, there is one smart grid consumer for every EDC of the edge computing federation.

4.1 Energy provider

The energy provider supplies electricity to smart grid consumers. Consumers pay the energy provider according to the electricity they consume from the grid. At time t, the energy provider offers electricity at price(t) \$/Wh. The provider revises this price with an update cycle of T_{PROVR} seconds.

4.2 Smart grid consumers

Consumers include four submodules: power demand, energy source(s), storage unit, and storage controller. The power demand, $p^{\text{demand}}_{\text{CONSR}}(t)$, represents the energy rate (in W) required by a consumer at time t. We describe the rest of the submodules below.

4.2.1 Energy sources

Energy sources provide electricity without buying it from the energy provider [e.g., photovoltaic (PV) systems or generator sets]. At time t, the energy sources of a consumer provide $p^{\text{gen}}_{\text{CONSR}}(t)$ W. The power generated by the consumer reduces its overall electricity costs. The consumer power surplus, $p^{\text{surplus}}_{\text{CONSR}}(t)$, is the difference between the consumer's power generation and the power demand:

$$p^{\text{surplus}}_{\text{CONSR}}(t) = p^{\text{gen}}_{\text{CONSR}}(t) - p^{\text{demand}}_{\text{CONSR}}(t) \text{ (W)}. \tag{15}$$

Ideally, it would be 0 W always (i.e., consumers generate the same power they demand). In this scenario, the electricity consumption from the energy provider would be 0 W. A negative power surplus would imply that the consumer requires more power than it can generate. In this case, the consumer would need to buy the remaining energy from the provider. On the other hand, a positive surplus indicates that power generation is greater than the demand, and the energy surplus is returned to the grid. This scenario is also undesirable since consumers usually sell the energy surplus at a lower price than the energy providers, even to the point of not receiving any economic benefit [51].

4.2.2 Storage unit

Consumers use energy storage units (e.g., batteries) to store their energy surplus and use it later when consumers generate less power than their demand. Each consumer can store up to $\text{CAP}^{\text{max}}_{\text{CONSR}}$ Wh in its energy storage unit. The capacity of the consumer's storage unit at time t, $\text{cap}_{\text{CONSR}}(t)$, must be between 0 and $\text{CAP}^{\text{max}}_{\text{CONSR}}$ Wh:

$$0 \leq \text{cap}_{\text{CONSR}}(t) \leq \text{CAP}^{\text{max}}_{\text{CONSR}}. \tag{16}$$

The consumer charging power, $p^{\text{charge}}_{\text{CONSR}}(t)$, corresponds to the electric power (in W) used to charge/discharge the consumer's energy storage unit. If $p^{\text{charge}}_{\text{CONSR}}(t)$ is greater than 0, the energy storage unit is charging. On the other hand, when the charging power is less than 0, the consumer

Table 3 Summary of the edge computing model

Parameter	Description
$\text{ap}_{\text{UE}}(t)$	AP to which the UE is connected at time t
$\text{edc}_{\text{EDC}}(t)$	EDC mapping policy applied by the EDC to determine which EDC should host computation offloading sessions of UE at time t
$T_{\text{PU}}^{\text{on}}$	Time required to switch on a PU before it can perform any other operation
$T_{\text{PU}}^{\text{off}}$	Time required to switch off a PU before it can perform any other operation
$U_{\text{PU}}^{\text{SRV}}$	Resource utilization required by service SRV to open a session on a PU while meeting a given QoS
$T_{\text{PU,SRV}}^{\text{proc}}$	Time required to process new requests of an active service session SRV on a PU before it can perform any other operation
$\mathbf{srv}_{\text{EDC}}(t)$	Set of active service sessions on an EDC at time t
$\text{prof}_{\text{EDC,SRV}}(t)$	Current demand profile of an EDC for service SRV at time t
$\text{estim}_{\text{EDC,SRV}}(t)$	Demand estimation of an EDC for service SRV at time t
$u_{\text{EDC}}(t)$	Mean resource utilization factor of an EDC at time t
$u_{\text{EDC}}^{\text{stdby}}(t)$	Mean resource utilization factor of an EDC at time t considering only PUs on hot standby
$p_{\text{EDC}}^{\text{IT}}(t)$	IT power consumption of an EDC at time t
$p_{\text{EDC}}^{\text{cool}}(t)$	Cooling power consumption of an EDC at time t
$p_{\text{EDC}}^{\text{demand}}(t)$	Overall power demand of an EDC at time t
$\text{pue}_{\text{EDC}}(t)$	PUE of an EDC at time t
$\text{disp}_{\text{EDC}}(\text{SRV}, t)$	Session dispatching policy of an EDC at time t
$\text{stdby}_{\text{EDC}}(t)$	Hot standby PUs policy of an EDC at time t
t_{prop}	Propagation time (in s) between the cloud and the RAN
v_{trans}	Mean transmission speed (in b s^{-1}) between the cloud and the RAN
$T_{\text{cloud,SRV}}^{\text{proc}}$	Time required to process new requests of an active service session SRV on the cloud facilities

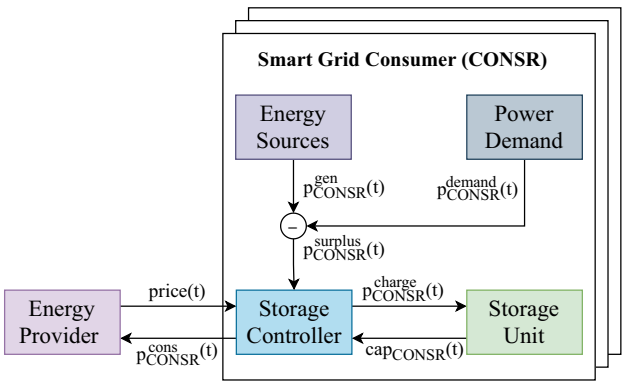

Fig. 5 Consumer-centric smart grid model

subtracts energy from the storage unit. The capacity of the storage unit at time t is obtained according to Eq. (17):

$$\text{cap}_{\text{CONSR}}(t) = \text{CAP}_{\text{CONSR}}^{\text{init}} + \frac{1}{3600}\int_0^t p_{\text{CONSR}}^{\text{charge}}(\tau)d\tau \text{ (Wh)}, \tag{17}$$

where $\text{CAP}_{\text{CONSR}}^{\text{init}}$ is the initial capacity (in Wh) of the storage unit. Note that, when integrating the charging power, the energy is expressed in J. We must divide the integration by 3600 to obtain the energy expressed in Wh.

The charging power is limited to the maximum charge/discharge power the storage unit allows, $\text{PMAX}_{\text{CONSR}}^{\text{charge}}$:

$$-\text{PMAX}_{\text{CONSR}}^{\text{charge}} \leq p_{\text{CONSR}}^{\text{charge}}(t) \leq \text{PMAX}_{\text{CONSR}}^{\text{charge}}. \tag{18}$$

4.2.3 Storage controller

This module is responsible for setting $p_{\text{CONSR}}^{\text{charge}}(t)$ depending on the current power surplus of the consumer and the electricity price. By default, it sets $p_{\text{CONSR}}^{\text{charge}}(t)$ to 0 W (i.e., electricity is not stored nor subtracted from the storage unit). When the electricity price is low enough, providing that the storage unit is not entirely charged, the energy controller will charge its storage unit at a rate of $\text{PMAX}_{\text{CONSR}}^{\text{charge}}$ W. $\text{PRICE}_{\text{CONSR}}^{\text{charge}}$ refers to the maximum price (in \$/Wh) a consumer is willing to pay for charging its energy storage unit using energy from the grid.

If the electricity price is higher than $\text{PRICE}_{\text{CONSR}}^{\text{charge}}$, the storage controller explores different configurations depending on the consumer's power surplus. When $p_{\text{CONSR}}^{\text{surplus}}(t) > 0$ (i.e., the consumer is generating more electric power than demanded), the storage controller sets

$p^{\text{charge}}_{\text{CONSR}}(t)$ to the minimum between the surplus and the storage unit's maximum charging power until the capacity reaches its maximum value. On the other hand, when $p^{\text{surplus}}_{\text{CONSR}}(t) < 0$, the storage controller will only subtract energy from the storage unit if the electricity price is higher than $\text{PRICE}^{\text{discharge}}_{\text{CONSR}}$ \$/Wh. The reason for this condition is that storing energy is a costly process. Thus, the storage controller must ensure that consuming the stored energy will provide a high enough decrease in the energy cost. Figure 6 shows a workflow diagram of the storage controller decision-making algorithm.

The power consumption of a consumer, $p^{\text{cons}}_{\text{CONSR}}(t)$, represents the electric power (in W) that the smart grid consumer gets from the grid. It is obtained from the charging power, demand, and power generation:

$$\begin{aligned} p^{\text{cons}}_{\text{CONSR}}(t) &= p^{\text{charge}}_{\text{CONSR}}(t) + p^{\text{demand}}_{\text{CONSR}}(t) - p^{\text{gen}}_{\text{CONSR}}(t) \\ &= p^{\text{charge}}_{\text{CONSR}}(t) - p^{\text{surplus}}_{\text{CONSR}}(t). \end{aligned} \tag{19}$$

For each consumer, the electricity cost at time t, $cost_{\text{CONSR}}(t)$, is computed by summing the electricity price multiplied by the energy consumption during every pricing cycle, as shown in Eq. (20). Note that, in this model, smart grid consumers do not benefit from returning energy to the grid. Thus, we do not consider those periods in which power consumption is less than 0 W.

$$cost_{\text{CONSR}}(t) = \sum_{k=0}^{\left\lfloor \frac{t}{T_{\text{PROVR}}} \right\rfloor} \left(\text{price}(kT_{\text{PROVR}}) \cdot \frac{1}{3600} \cdot \int_{kT_{\text{PROVR}}}^{\min(t,(k+1)\cdot T_{\text{PROVR}})} \max\left(0, p^{\text{cons}}_{\text{CONSR}}(\tau)\right) d\tau \right) (\$). \tag{20}$$

Table 4 describes of the most relevant parameters of the smart grid model.

4.3 Smart grid-aware edge computing model

The edge computing model presented in Sect. 3 can be combined with the smart grid model proposed in this section to reduce the operational expenses of edge computing facilities. Figure 7 shows a schematic of the proposed hybrid model.

EDCs of the edge computing federation are also smart grid consumers. The power demand of an EDC corresponds to the power demand of its matching smart grid consumer. Additionally, the EDC interface adds smart grid-related fields to the EDC's current status. Finally, EDCs' interface and policy manager are also aware of the current electricity price offered by the energy provider. By doing so, they can incorporate these new fields into the search space and optimize the EDC configuration considering both edge computing and smart grid-related parameters.

5 Use case

We first propose an edge computing scenario to show how edge computing can improve the QoS while reducing the network traffic to the Internet. After selecting a convenient configuration, we include smart grid elements and compare their simulation outcome to illustrate the benefits of connecting edge computing infrastructures to the smart grid. We executed the simulations sequentially on a MacBook Pro Retina, 15-in., Mid 2015 with a 2.5 GHz Intel Core i7 processor, 16 GB 1600 MHz DDR3 memory, and macOS 12.5.

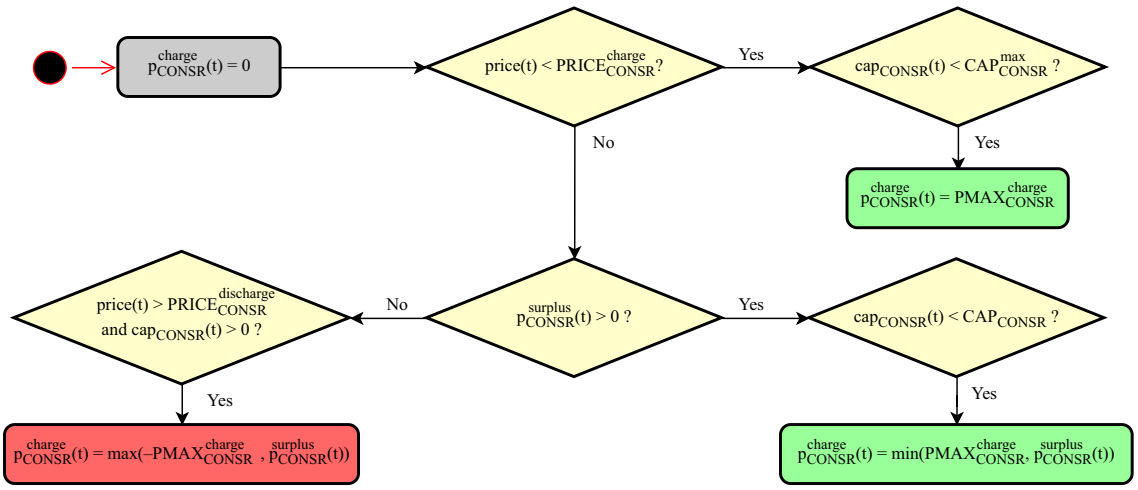

Fig. 6 Workflow diagram for computing the charging power of the storage unit

Table 4 Summary of the smart grid model

Parameter	Description
$\text{price}(t)$	Price (in \$/Wh) offered by energy provider at time t
$\text{PRICE}^{\text{charge}}_{\text{CONSR}}$	Maximum price (in \$/Wh) that consumer CONSR is willing to pay for charging its energy storage unit
$\text{PRICE}^{\text{discharge}}_{\text{CONSR}}$	Minimum price (in \$/Wh) for consumer CONSR to consider discharging its energy storage unit
$p^{\text{gen}}_{\text{CONSR}}(t)$	Electric power (in W) generated by the energy sources of the smart grid consumer CONSR at time t
$p^{\text{charge}}_{\text{CONSR}}(t)$	Electric power (in W) used for charging the storage unit of smart grid consumer CONSR at time t. Negative values imply that the unit is being discharged
$\text{PMAX}^{\text{charge}}_{\text{CONSR}}$	Maximum charge/discharge power (in W) of the energy storage unit of smart grid consumer CONSR
$p^{\text{cons}}_{\text{CONSR}}(t)$	Electric power (in W) consumed from the grid by smart grid consumer CONSR at time t. Negative values indicate that CONSR returns energy to the grid
$\text{cost}_{\text{CONSR}}(t)$	Cost (in \$) of the energy consumed from the grid by smart grid consumer CONSR
$\text{cap}_{\text{CONSR}}(t)$	Energy capacity (in Wh) of the energy storage unit of smart grid consumer CONSR at time t
$\text{CAP}^{\text{max}}_{\text{CONSR}}$	Maximum energy capacity (in Wh) of the energy storage unit of smart grid consumer CONSR
$\text{CAP}^{\text{init}}_{\text{CONSR}}$	Initial energy capacity (in Wh) of the energy storage unit of smart grid consumer CONSR

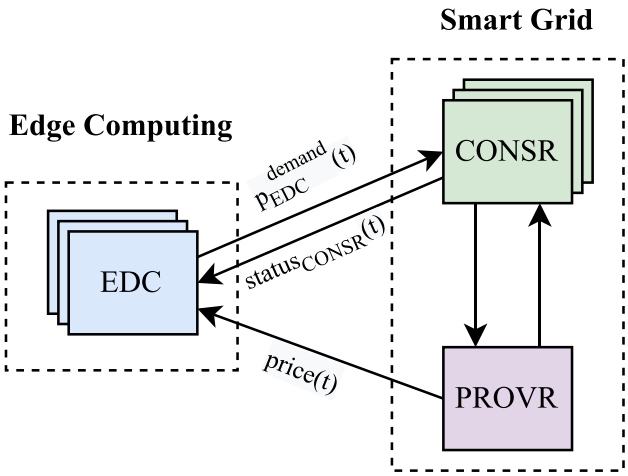

Fig. 7 Smart grid-aware edge computing model

5.1 Scenario description

The scenario is an ADAS application. Vehicles periodically capture images of their driver's face to detect potentially dangerous situations (e.g., distractions or eye fatigue) using an onboard predictive model. Cars use the edge computing federation to train their predictive models remotely. Each vehicle requests to open a new session every 20 min. As soon as this session starts, the automobile sends a batch containing new driver images with additional information gathered by the vehicle (e.g., GPS position, brake press, or steering angle). The session remains open for 15 min. During this time, the element hosting the session (i.e., an EDC or the cloud) trains the model with new data gathered by the vehicle and other nearby vehicles. When the automobile closes the session, the computing element sends the re-trained model to the corresponding vehicle if this new model differs significantly from the one in the car. Computation offloading service demand corresponds to real mobility traces of taxis in San Francisco, USA [52]. The data set contains GPS coordinates of 535 taxis collected over May and June 2008 in the San Francisco Bay Area. Figure 8 shows the number of taxis in the scenario from May 17th to June 6th, both included. All the simulations in this research use the mobility traces of June 6th, 2008. Figure 8 depicts this day in orange.

The PUs comprising the EDCs' computation resources are AMD Sapphire Pulse Radeon RX 580 graphics processing units (GPUs). Table 5 contains all the configuration parameters of the PU model used in the simulations. This configuration is based on the work of Pérez et al. [53].

We set the propagation time between the cloud and the RAN, t_{prop}, to 80 ms. Alternatively, the mean transmission speed between the cloud and the RAN is 600 Mb s^{-1}. We assume that cloud data centers use the same hardware to process service requests. Therefore, $T^{\text{proc}}_{\text{cloud,SRV}}$ is 0.1 s. As

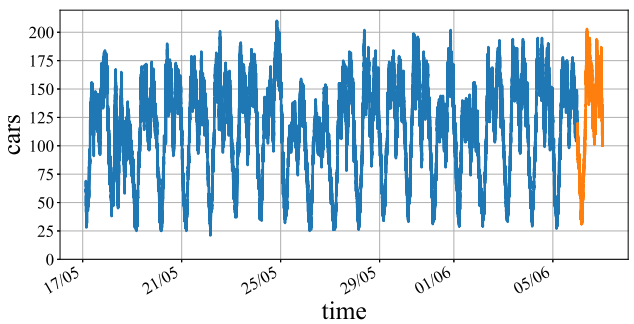

Fig. 8 Traffic flow in the scenario from May 17th to June 6th

Table 5 Configuration parameters of processing units

Parameter	Value
$U_{\mathrm{PU}}^{\mathrm{SRV}}$	20% of PU's computing resources
$T_{\mathrm{PU}}^{\mathrm{on}}$	60 s
$T_{\mathrm{PU}}^{\mathrm{off}}$	10 s
$T_{\mathrm{PU,SRV}}^{\mathrm{proc}}$	0.1 s
$p_{\mathrm{PU}}(t)$	Custom power consumption model

shown in Fig. 8, the demand peak in the scenario is slightly greater than 200 vehicles. Every PU can host up to $\frac{100}{U_{\mathrm{PU}}^{\mathrm{SRV}}} = 5$ simultaneous sessions. Thus, the edge federation must contain at least $\frac{200}{5} = 40$ PUs. We decided to add redundant PUs to avoid system congestion in case an unusual demand peak occurs. The edge federation comprises three EDCs, each containing 20 PUs. In this way, the sessions are divided into three different geographic areas. Each EDC can host up to 100 sessions.

The pump-driven two-phase immersion cooling system of the EDCs uses the 3M Novec 7100 fluid as refrigerant and water as condenser secondary working fluid ($\rho \approx 1\,\mathrm{g\,ml^{-1}}$, $C_p = 4.1813\,\mathrm{J\,g^{-1}\,K^{-1}}$). The difference between the outlet and inlet temperature of the water is 20 K. Each PU can consume up to 100 W, approximately. Therefore, the maximum IT power of each EDC is $100 \cdot 20 = 2\,\mathrm{kW}$. Thus, according to Eq. (4), the pump used for driving the condenser fluid must be able to provide $0.086\,\mathrm{m^3\,h^{-1}}$ (i.e., $1.434\,\mathrm{l\,min^{-1}}$). We selected a 0142YA-12-15 micro diaphragm pump, which provides a flow of $1.5\,\mathrm{l\,min^{-1}}$ consuming $p_{\mathrm{EDC}}^{\mathrm{cool}}(t) = 15\,\mathrm{W}$. On the other hand, we assume that the mean PUE of the cloud is 1.5.

We used the allocation manager tool included with the Mercury M&S&O framework to determine the location of APs and EDCs [14]. This tool divides the scene into a grid of 40 m by 40 m cells. Next, it analyzes the user mobility traces to determine the maximum number of users inside every spatial cell during a time window of 1 min. Figure 9 shows the spatial density of users in the scenario computed by the allocation manager.After calculating the scenario density, the allocation manager applies the Same-Size K-Means algorithm to place APs and EDCs in suitable locations for distributing the service demand evenly. Figure 10 shows the scenario setup used in the experiments. Small dots correspond to user location traces used by the allocation manager. The 19 stars scattered in the scenario represent the location of APs. Additionally, three EDCs (represented as big squares) provide computation offloading to all the users. The color of every element corresponds to the preferred EDC.

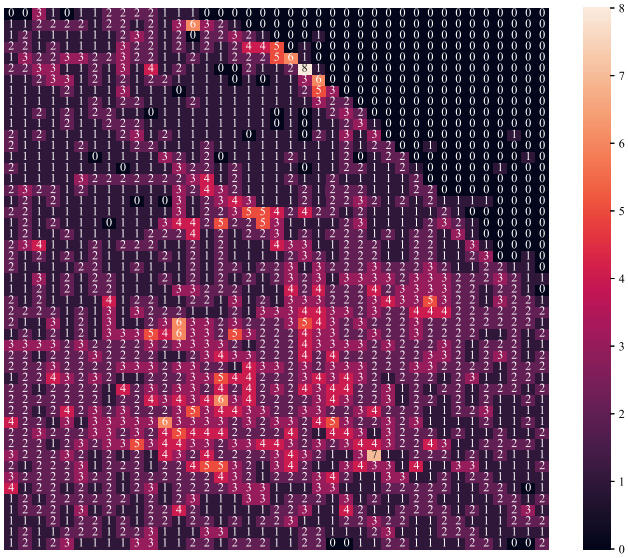

Fig. 9 Spatial density of users in the scenario

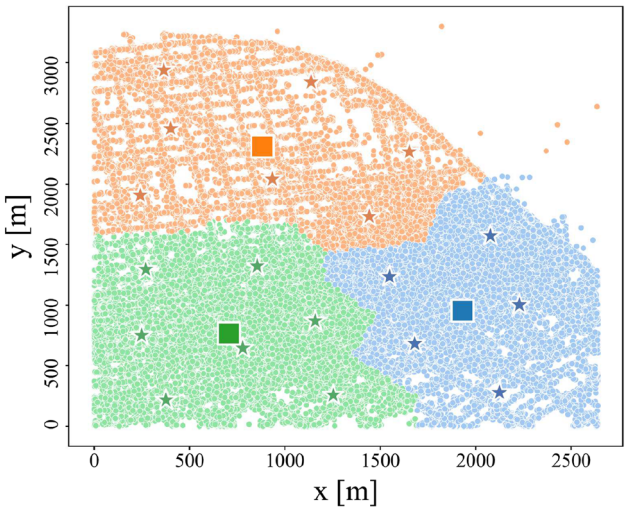

Fig. 10 Scenario setup

5.2 Scenarios without smart grid integration

First, we simulate several edge computing scenarios that do not integrate any aspect of the presented smart grid model. In particular, we simulated edge scenarios with different hot standby policies.

5.2.1 Static hot standby policies

The first two policies are static (i.e., they do not vary throughout the simulation):

- *None* PUs do not work in hot standby mode and switch on only if they host one or more sessions.
- *All* all the PUs are on hot standby and are switched on even if they are idling.

These static policies serve as yardsticks for assessing the efficacy of the remaining hot standby approaches. For instance, in the *None* policy, the hot standby utilization of the EDCs is always 1. Therefore, the edge computing federation forwards all the requests to the cloud. On the other hand, with the *All* policy, the EDCs accept all incoming requests. Consequently, this policy presents the lowest delay perceived by the users.

The edge computing federation does not process requests with the *None* policy. Accordingly, the overall energy consumption of the EDCs is 0 Wh. Thus, the cloud is responsible for processing all the requests. On average, clients experienced delays of 160.000 ms for opening sessions, 273.334 ms for sending information before starting the online training, and 160.267 ms when closing the session and receiving the newly trained model. Figure 11 shows the demand for cloud resources in this scenario.

The cloud would need at least 41 PUs to support a peak demand of 203 clients. Figure 12 shows a power consumption estimation of the cloud facilities to support the scenario demand. The mean power consumption of the IT infrastructure is 4.20 kW, with peaks reaching 4.52 kW. Considering a PUE of 1.5, the total power consumption associated with the service is, on average, 6.30 kW, with peaks approximating 6.79 kW. In this scenario, the total energy consumption of the cloud is 150.93 kWh.

In contrast, the cloud is unnecessary with the *All* policy, as the EDCs can process all the requests. Figure 13 shows the demand for edge computing resources under this policy.The EDC edc_2 is the most crowded one, as it is closer to the rest of the City of San Francisco. On the other hand, while the preferred region of the EDC edc_1 is the largest, it is located in a less crowded zone, and its average demand is the lowest. The delay experienced by clients for opening and closing sessions is negligible. On the other hand, the delay in sending information before starting the online training is very close to the processing time of the PUs (i.e., 100 ms).

Figure 14 shows the power consumption of the edge computing federation.With the *All* policy, the PUs of the EDCs are always on. The mean power consumption is 5.78 kW (i.e., 1.58 kW more than the power consumed by the cloud when using the *None* policy). However, the characteristics of the edge infrastructure lead to significantly lower PUEs. Figure 15 shows the PUE of the EDCs in the scenario. Note that the PUE is inversely proportional to the power consumption, as the cooling power remains constant regardless of the demand.

Overall, the total energy consumption of the edge federation is 139.00 kWh, which is 11.93 kWh less than the cloud with the *None* policy (i.e., 7.9% less). Thus, edge computing with no cloud cooperation allows us to provide a better QoS while reducing energy consumption of the infrastructure.

5.2.2 Proactive hot standby policies

Next, we explore how proactive hot standby policies can improve the overall performance of the edge computing infrastructure while working with cloud. These proactive strategies rely on service demand predictions to modify the hot standby resources on each EDC. In this work, the service demand estimation corresponds to the daily average traffic flow in the scenario according to the 20 previous days before June 6th. However, modelers can implement their demand estimation model with the Mercury M&S&O framework. Figure 16 depicts the daily average traffic flow in the San Francisco Bay Area according to the mobility traces from May 17th to June 5th.The hot standby policy will dynamically change to ensure that each EDC maintains as many PUs as necessary in hot standby to accept the

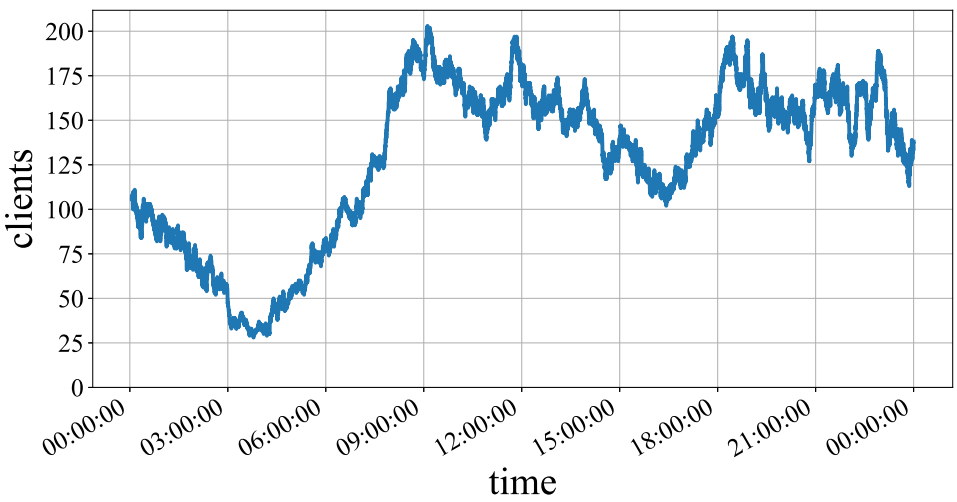

Fig. 11 Demand for cloud resources with *None* policy

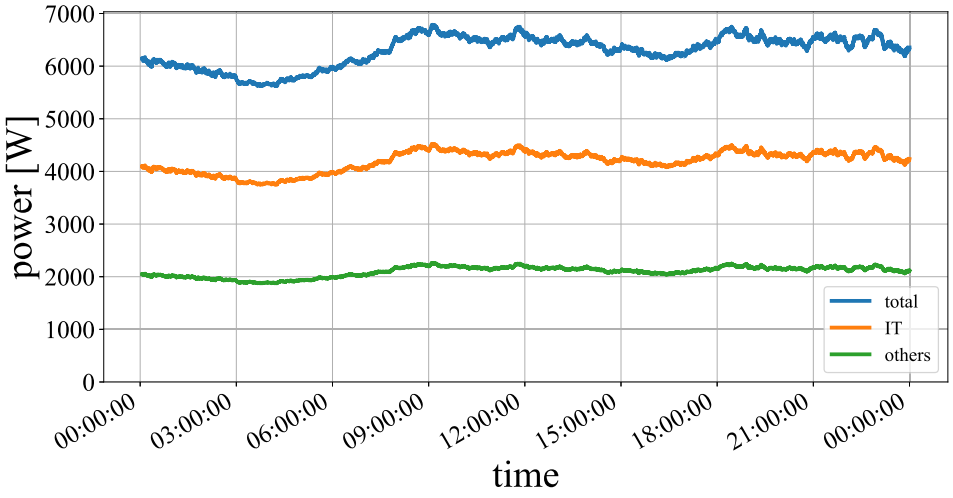

Fig. 12 Cloud power consumption estimation with *None* policy

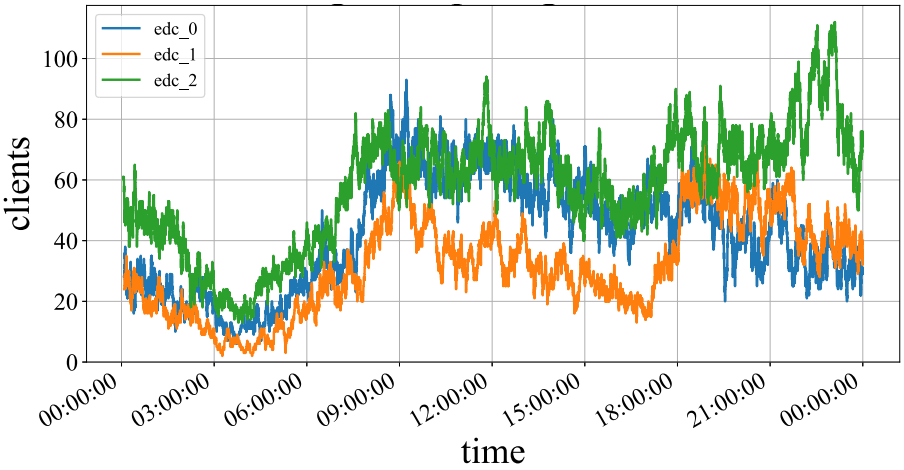

Fig. 13 Demand for edge computing resources with *All* policy

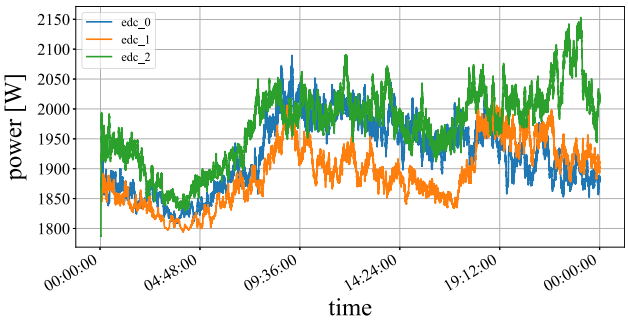

Fig. 14 Power consumption of edge computing resources with *All* policy

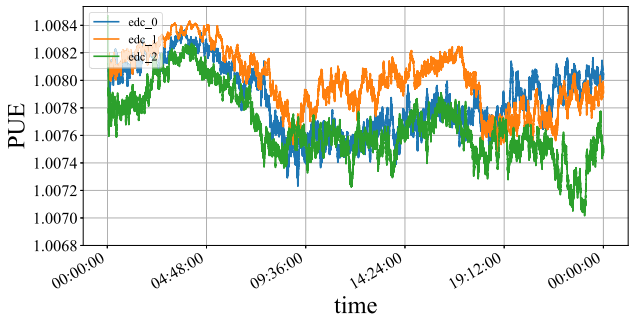

Fig. 15 PUE of EDCs with *All* policy

expected service requests without switching on any additional PU. At time t, the number of PUs on hot standby of a given EDC is computed as follows:

$$n^{\text{stdby}}_{\text{EDC,SRV}}(t) = \left\lceil \text{estim}_{\text{EDC,SRV}}(t) \cdot (1+\alpha) \cdot U^{\text{SRV}}_{\text{PU}} \right\rceil, \qquad (21)$$

where α is a correction factor that assumes that the demand will be a fixed percentage higher than the prediction. While this estimation increment leads to higher power consumption, it is more robust against anomalous demand peaks, achieving a better QoS. We explored six proactive policies with different values for the correction factor. Namely, α is set to 0.0, 0.1, 0.2, 0.3, 0.4, and 0.5. Table 6 summarizes the results of the static and proactive hot standby policies.

The mean service delay perceived by clients decreases as we increase the value of α. Compared to the *All* policy, the proactive policy with $\alpha = 0.0$ exhibits a degradation of

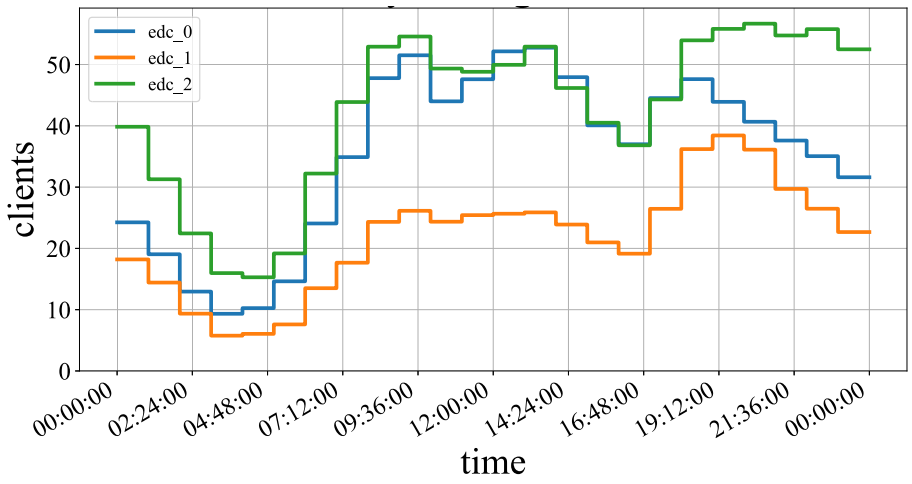

Fig. 16 Hourly average traffic flow obtained from the previous 20 days

Table 6 Simulation results for static proactive hot standby policies

Scenario	Energy (kWh)	Mean service delay (ms)	Mean session start delay (ms)
None	150.930	273.334	160.000
All	138.998	100.000	0.000
Proactive ($\alpha = 0.0$)	106.187	109.044	8.349
Proactive ($\alpha = 0.1$)	98.172	103.433	3.169
Proactive ($\alpha = 0.2$)	85.280	101.806	1.667
Proactive ($\alpha = 0.3$)	**83.816**	**100.768**	**0.709**
Proactive ($\alpha = 0.4$)	84.816	100.403	0.376
Proactive ($\alpha = 0.5$)	86.000	100.210	0.197

9 ms. This degradation diminishes as α increases. For $\alpha = 0.5$, the performance degradation is near to 200 ms. In contrast, the total energy consumption of the scenarios does not follow a trend proportional to α. Figure 17 divides the infrastructure energy consumption into the EDCs and the cloud.

The energy consumption of all the EDCs increases with α. For instance, edc_0 needs 19.81 kWh for $\alpha = 0.0$, while it consumes 27.68 kWh for $\alpha = 0.5$ (i.e., 39.73% more). However, as the edge infrastructure has more available resources, we can reduce the number of backup cloud resources. For example, for $\alpha = 0.0$, the peak demand in the cloud is 73 clients. Thus, the cloud needs 15 PUs to fulfill the demand instead of the 41 PUs required with the *None* policy. In contrast, the peak cloud demand when $\alpha = 0.5$ is 9 clients, and we only need 2 PUs. Thus, even if the edge federation consumes more energy, it is more efficient than the cloud, and the overall energy decreases.

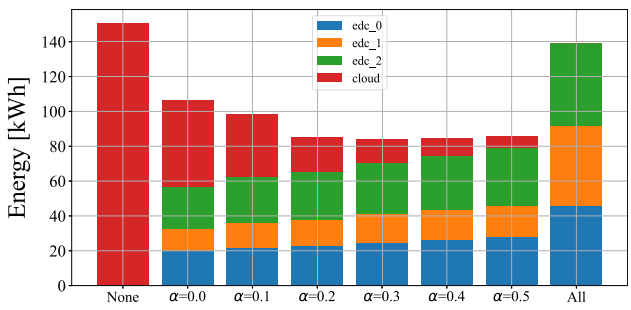

Fig. 17 Energy consumption of proactive hot standby policies

Choosing a convenient hot standby policy depends on the requirements of the service. If the application needs low response times, we should select a policy that puts the service delay before power consumption (e.g., *All*). We can use the Mercury M&S&O tool to explore different scenarios and find the configuration that better suits our requirements. For this use case, we selected the proactive hot standby policy with $\alpha = 0.3$ (the one outlined in bold in Table 6) as the reference proactive hot standby policy because this scenario shows the lowest energy consumption and its average QoS degradation is below 1 ms. Table 7 shows more simulation results of this scenario.

The EDC edc_2 is the most loaded EDC, with a mean utilization of 58.08%. On the other hand, the average utilization of EDCs edc_0 and edc_1 is 43.36% and 31.24%, respectively. Figure 18 shows the resource utilization of EDCs throughout the simulation.As depicted in Fig. 16, the hourly demand estimation from the previous 20 days

Table 7 Simulation results for proactive hot standby policy ($\alpha = 0.3$)

	edc_0	edc_1	edc_2	Cloud
Mean utilization (%)	43.36	31.24	58.08	9.41
Mean hot standby utilization (%)	80.46	90.12	81.89	9.41
Mean power consumption (W)	869.49	813.41	1528.39	550.344
Mean PUE	1.02	1.03	1.01	1.50
Energy consumption (kWh)	24.618	16.264	29.726	13.208

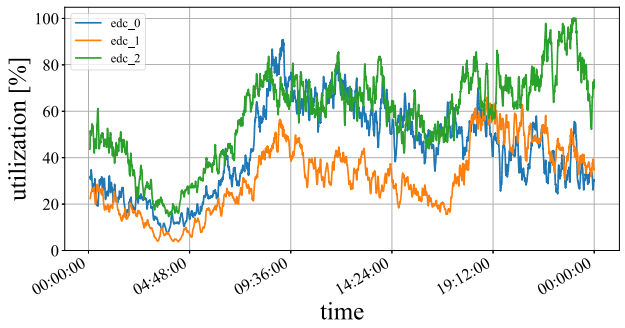

Fig. 18 EDC utilization with proactive hot standby policy ($\alpha = 0.3$)

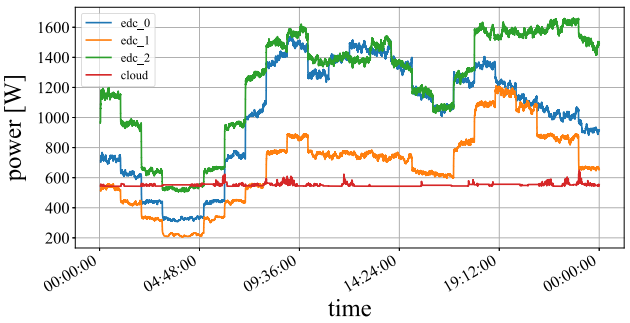

Fig. 20 Power consumption with proactive hot standby policy ($\alpha = 0.3$)

presents similar results. While the resource utilization of the EDCs resembles the scenario demand, the EDCs aim to maximize their hot standby utilization to reduce their energy consumption. Figure 19 shows the hot standby utilization of the EDCs. For every EDC, the mean hot standby utilization is between 80.46 and 90.12%.

Figure 20 shows the power demand of the EDCs of the edge federation and the backup resources in the cloud.The shape of the power demand curves of the EDCs is similar to their utilization. However, the priors are more staggered. This is because the static power of PUs (i.e., the power consumption of PUs when there is no activity) is significantly higher than their dynamic power (i.e., the power consumption increment when a PU is busy compared to its static power). As the number of PUs switched on changes depending on the service demand estimation, the power demand changes significantly every hour. In contrast, the power consumption of the cloud backup resources is stable at around 550 W, as all the PUs remain on hot standby throughout the simulation. The EDC edc_2 presents the best mean PUE (1.01), while the EDC edc_1 obtains the worst average PUE (1.03).

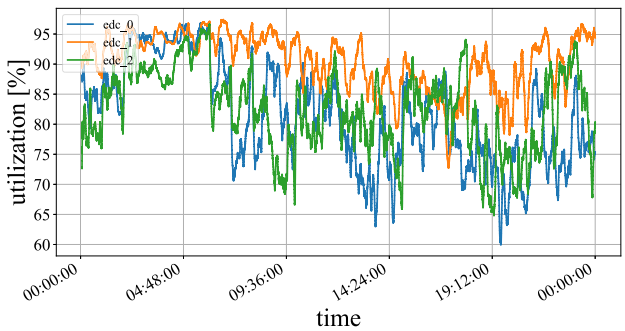

Fig. 19 EDC hot standby utilization with proactive hot standby policy ($\alpha = 0.3$)

5.2.3 Hybrid hot standby policies

Next, we explore the effect of using a hybrid hot standby policy to improve the QoS of the infrastructure. These hybrid strategies combine service demand predictions with the current service demand of the scenario. At time t, the number of PUs on hot standby of a given EDC is computed as follows:

$$n^{\text{stdby}}_{\text{EDC,SRV}}(t) = \lceil \max(\text{demand}_{\text{EDC,SRV}}(t), \text{estim}_{\text{EDC,SRV}}(t)) \cdot (1+\alpha) \cdot U^{\text{SRV}}_{\text{PU}} \rceil. \quad (22)$$

The hybrid policy reacts faster to demand peaks, achieving a better QoS. However, the power consumption of the EDCs tends to be greater than with the proactive policy. We explored six hybrid policies with different values for the correction factor. Namely, α is set to 0.0, 0.1, 0.2, 0.3, 0.4, and 0.5. Table 8 summarizes the results of the hybrid hot standby policies.

The mean service delay perceived by clients is considerably lower than the achieved by the proactive policy. The hybrid policy with $\alpha = 0.0$ shows a degradation of 8 ms and diminishes as α increases. Figure 21 divides the infrastructure energy consumption into the EDCs and the cloud.

Table 8 Simulation results for hybrid hot standby policies

Scenario	Energy (kWh)	Mean service delay (ms)	Mean session start delay (µs)
Hybrid ($\alpha = 0.0$)	91.629	100.058	97.58
Hybrid ($\alpha = 0.1$)	91.426	100.015	56.02
Hybrid ($\alpha = 0.2$)	**90.407**	**100.008**	**21.62**
Hybrid ($\alpha = 0.3$)	92.550	100.002	1.75
Hybrid ($\alpha = 0.4$)	97.296	100.000	0.29
Hybrid ($\alpha = 0.5$)	97.812	100.000	0.00

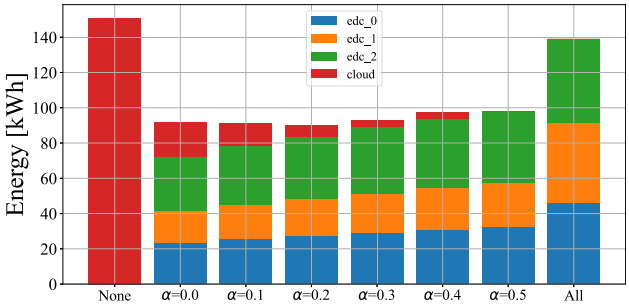

Fig. 21 Energy consumption of hybrid hot standby policies

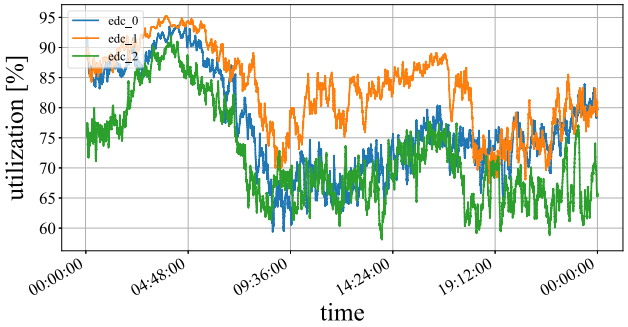

Fig. 22 EDC hot standby utilization with hybrid hot standby policy ($\alpha = 0.2$)

Compared to proactive policies, the scenarios with hybrid policies use fewer cloud resources. For example, for $\alpha = 0.0$, the peak demand in the cloud is 26 clients, which only demands 6 PUs (instead of the 15 PUs required with the proactive policy with $\alpha = 0.0$). When $\alpha = 0.5$, the cloud is no longer needed, as the edge computing federation can process all the demand. Thus, the QoS matches the *All* policy using 29.63% less energy. Choosing between proactive and hybrid hot standby policies depends on the requirements of the service. In this paper, we selected the hybrid hot standby policy with $\alpha = 0.2$ (the one outlined in bold in Table 8) because this scenario shows the lowest energy consumption. Furthermore, the average QoS degradation is below 50 ms. Table 9 shows more simulation results of this scenario.

The utilization of the EDCs is similar to the utilization of the hybrid policy with $\alpha = 0.3$. However, the hot standby utilization of the hybrid policy is slightly smaller than in the proactive, as the EDCs tend to keep more PUs in hot standby through the simulation. Figure 22 shows the hot standby utilization of EDCs throughout the simulation.

Figure 23 shows the power consumption of the edge federation and the cloud.Compared to the proactive policy, the effect of the demand prediction is less obvious. For instance, the power demand curves of the EDCs are not staggered and resemble more to their utilization. Additionally, the power consumption of the backup cloud resources is 49.09% less than with the proactive policy with $\alpha = 0.3$. The cloud resources remain idle most of the time and only receive requests at dawn. As the power consumption of this scenario is slightly greater than with the proactive policy, the PUE of this scenario is better. Figure 24 shows the PUE of each EDC.

As the cooling system only requires 15 W per EDC, their PUE is close to 1 most of the time. Note that the PUE is inversely proportional to the service demand. Thus, EDCs edc_0 and edc_2 present the best mean PUE (1.01), while the EDC edc_1 obtains the worst average PUE (1.02). At dawn, the resource utilization of all the EDCs is lower than in the rest of the day, and their PUE is considerably greater comparing the rest of the day.

Table 9 Simulation results for hybrid hot standby policy ($\alpha = 0.2$)

	edc_0	edc_1	edc_2	Cloud
Mean utilization (%)	42.51	31.85	57.52	12.29
Mean hot standby utilization (%)	75.42	81.30	71.00	12.29
Mean power consumption (W)	1140.66	865.35	1480.78	280.154
Mean PUE	1.01	1.02	1.01	1.50
Energy consumption (kWh)	27.376	20.768	35.539	6.724

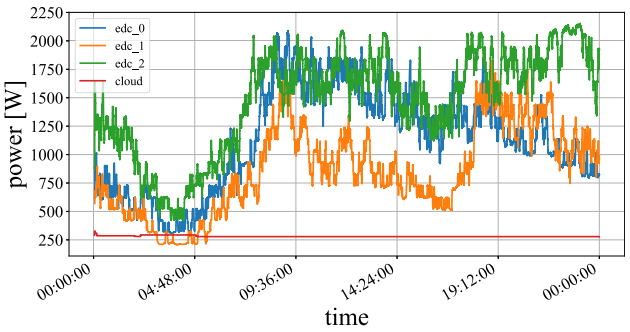

Fig. 23 Power consumption with hybrid hot standby policy ($\alpha = 0.2$)

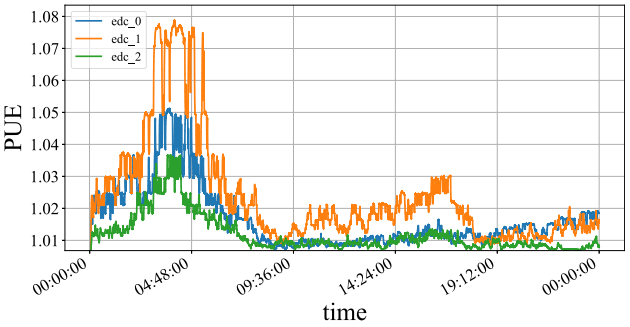

Fig. 24 PUE of EDCs with hybrid hot standby policy ($\alpha = 0.2$)

5.3 Integrating smart grid to the scenario

The next set of experiments integrates the smart grid model described in Sect. 4 to reduce the impact of the energy price fluctuations on the overall service cost. The electricity price in the simulations corresponds to the average hourly day-ahead wholesale energy price in California in 2017 (see Fig. 25).It is important to note that the wholesale price typically corresponds to 35% of the final retail price. However, it is possible to compare the results of different simulations.

With this pricing scheme and the hybrid policy with $\alpha = 0.2$, the edge federation infrastructure would spend 2.44 $ to satisfy its energy demand for this day. Figure 26 represents the accumulated energy cost per EDC when

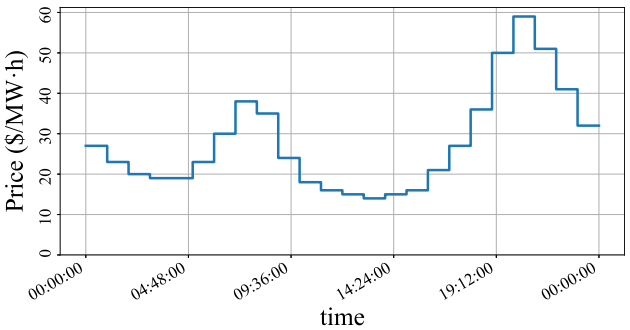

Fig. 25 Average hourly day-ahead energy price in scenario

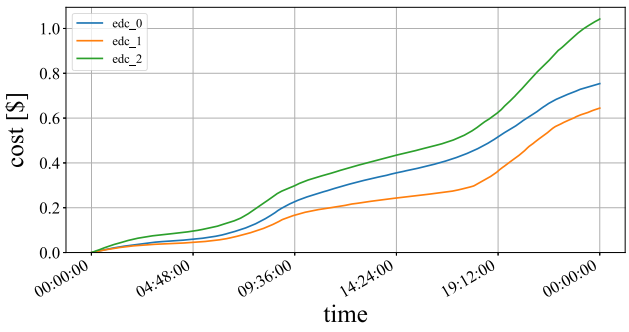

Fig. 26 Accumulated energy cost per EDC with hybrid policy ($\alpha = 0.2$)

using the hybrid hot standby policy with $\alpha = 0.2$.The cost curves present significant increments from 06:00 to 09:00 and from 18:00 to 23:00, coinciding with the energy price peaks of Fig. 25. Smart grid consumers can alleviate the overall cost by integrating energy generation sources and storage systems.

Next, we present four scenarios that integrate the smart grid model to reduce the energy consumption and cost of the edge computing federation. The EDCs use the hybrid ($\alpha = 0.2$) hot standby policy in all the scenarios. Additionally, EDCs incorporate two BISTAR TP6L72M(H) 9BB crystalline silicon (c-Si) PV panels to reduce their power consumption. Each PV system provides a peak power of 910 W and presents 14% of system losses. We obtained the data traces of the PV panels' power generation from the PVGIS-NSRDB database according to the approximate location of the EDCs in Fig. 10. The time precision provided by this database is 1 h. Due to the discrete-event nature of the simulator, the simulated power generation looks staggered, as shown in Fig. 27.

The EDCs have a low-voltage energy storage system. Depending on the scenario, the storage unit of the EDCs may correspond to a PylonTech US2000C battery ($\text{CAP}^{\text{max}}_{\text{CONSR}} = 2.28\,\text{kWh}$, $\text{PMAX}^{\text{charge}}_{\text{CONSR}} = 1.20\,\text{kW}$) or a US3000C battery ($\text{CAP}^{\text{max}}_{\text{CONSR}} = 3.37\,\text{kWh}$, $\text{PMAX}^{\text{charge}}_{\text{CONSR}} = 1.78\,\text{kW}$). The maximum charge/discharge power corresponds to the values recommended by the

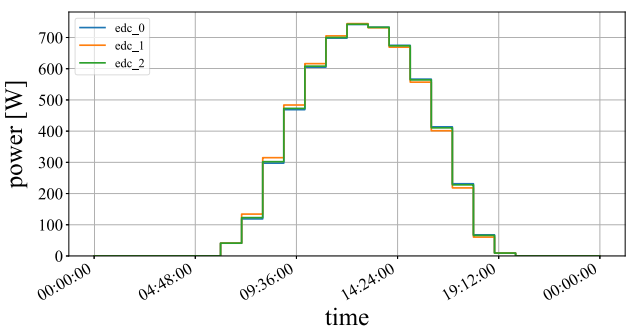

Fig. 27 Power generation of PV panels of the EDCs in June 6th

manufacturer. Even though the battery can charge/discharge faster, the storage controller will limit it to extend the battery life. The batteries of all the EDCs are fully discharged at the beginning of the simulation. We also explored two storage controller policies for the EDCs. The first one, called *Loose*, sets $\text{PRICE}_{\text{CONSR}}^{\text{charge}}$ to 20 \$/MWh and $\text{PRICE}_{\text{CONSR}}^{\text{discharge}}$ to 35 \$/MWh. The second policy, called *Strict*, sets more demanding requirements for charging or discharging the battery. Namely, it configures $\text{PRICE}_{\text{CONSR}}^{\text{charge}}$ to 19 \$/MWh and $\text{PRICE}_{\text{CONSR}}^{\text{discharge}}$ to 37 \$/MWh. While we limit the set of experiments to one hot standby policy, two battery models, and two storage controller strategies, Mercury allows us to explore other scenario parameters to optimize the edge computing infrastructure's overall performance (e.g., EDC mapping strategy or power generation modules). Table 10 shows the simulation results obtained by the scenarios that integrate the smart grid model. The first row corresponds to the base case scenario with no smart grid integration. Note that the shown results do not include the energy consumption of the backup cloud resources, as these are not affected by the integration of smart grid technologies on the edge infrastructure.

The EDCs' energy demand and the mean service delay of all the smart grid scenarios are the same as the base case. However, the smart grid scenarios can reduce the overall energy consumption by 20.3% due to the PV system installed in the EDCs. As the PV system and resource management policies are the same in all the smart grid experiments, there is no significant difference in the overall energy consumption. However, the energy cost is different for each configuration.

Simulations with US2000C batteries show a 24.6/25.4% reduction in service cost. On the other hand, simulations with US3000C batteries achieved a 29.1/30.3% reduction. This is because the maximum capacity of these batteries is 47.8% greater than the US2000C model, allowing the EDCs to store more energy to reduce their energy consumption for a longer time when the electricity price is high. Alternatively, the *Strict* storage controller policy reduces the overall service cost compared to the *Loose* policy. We achieve this reduction by limiting battery usage to higher electricity prices. Figures 28 and 29 show the energy stored in the EDCs' batteries for the scenarios with the US3000C battery and the *Loose* and *Strict* power storage policies, respectively. Green areas correspond to periods when the electricity price is equal to or less than $\text{PRICE}_{\text{CONSR}}^{\text{charge}}$. On the other hand, red areas represent periods when the electricity price is equal to or greater than $\text{PRICE}_{\text{CONSR}}^{\text{discharge}}$.

The *Loose* policy starts to charge the battery at 2:00 AM when the electricity price reaches 20 \$/MWh. EDCs can charge their batteries until 5:00 AM when the electricity price is 23 \$/MWh. However, their batteries are fully charged one hour before, approximately. On the other hand, the *Strict* policy allows EDCs to charge their batteries at 3:00 AM when the electricity price is 19 \$/MWh. Even though the *Strict* policy starts charging the batteries later, the charging time window is enough to charge the batteries while saving energy costs.

From 7:00 AM to 8:00 AM (i.e., the first red area), the electricity price is above $\text{PRICE}_{\text{CONSR}}^{\text{discharge}}$ in both scenarios. Thus, the storage controllers subtract energy from their batteries to reduce the EDCs' power consumption. With the *Loose* policy, EDCs get electricity from the batteries until 9:00 AM. The capacity of the batteries decreases at different rates, depending on their corresponding power demand. None of the batteries fully discharges. However, their capacity is significantly less with the *Loose* policy, as EDCs use them for a prolonged period. From 10:00 AM to 4:00 PM, the storage controllers can consume power to charge the batteries. However, batteries reach their maximum capacity in less than an hour. Note that the scenario with the *Loose* policy requires more energy to charge the batteries, as they are initially less charged.

With the *Loose* policy, EDCs start discharging their batteries at 6:00 PM to reduce their power consumption. Batteries are fully discharged around 8:00 PM, coinciding when the electricity price reaches its maximum value (59 \$/MWh). In contrast, with the *Strict* policy, batteries provide energy from 7:00 PM to 9:00 PM, reducing the EDCs' power consumption when electricity is more expensive.

Table 10 Simulation results for smart grid scenarios

Configuration	Energy demand (kWh)	Energy consumption (kWh)	Energy return (Wh)	Cost (\$)
Base	83.684	83.684	0.000	2.44
US2000C (*Loose*)	83.684	66.672	16.11	1.84
US2000C (*Strict*)	83.684	66.672	16.11	1.82
US3000C (*Loose*)	83.684	66.672	16.11	1.73
US3000C (*Strict*)	**83.684**	**66.672**	**16.11**	**1.70**

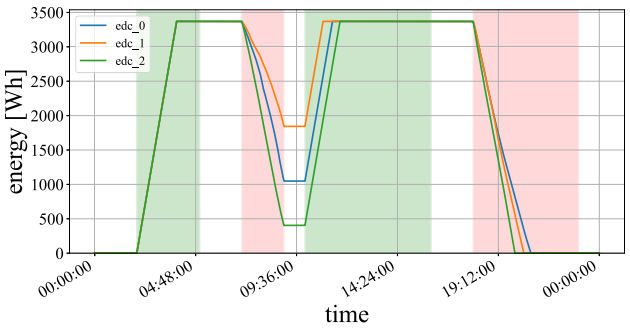

Fig. 28 Energy stored by EDCs with US3000C battery and *Loose* power storage policy

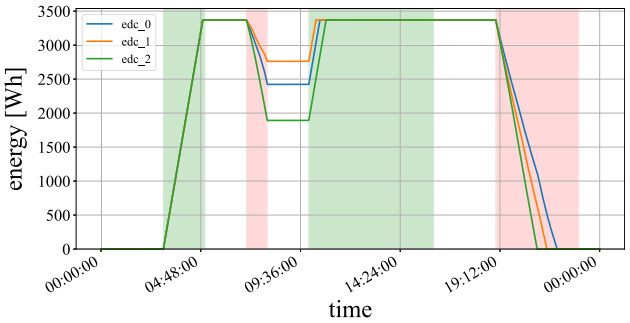

Fig. 29 Energy stored by EDCs with US3000C battery and *Strict* power storage policy

Again, choosing the best configuration depends on the specific use case. For instance, even though the US3000C batteries show better performance, we need to consider the trade-off between a higher capital expense for buying batteries with better performance and overall operational costs. We may also consider heterogeneous scenarios in which EDCs edc_0 and edc_1 have a US2000C battery while EDC edc_2 contains a US3000C battery, as its overall utilization is higher. Here we select the scenario with US3000C batteries and the *Strict* storage controller policy. Table 10 highlights this scenario in bold. Table 11 contains additional simulation results of the selected scenario.

Table 11 Simulation results in smart grid scenario

	edc_0	edc_1	edc_2
Mean utilization (%)	42.51	31.85	57.52
Mean hot standby utilization (%)	75.42	81.30	71.00
Mean PUE	1.01	1.02	1.01
Mean power demand (W)	1140.69	865.42	1480.75
Mean power consumption (W)	888.86	587.82	1224.61
Energy return (Wh)	0.00	16.11	0.00
Energy consumption (kWh)	21.709	15.099	29.864
Cost ($)	0.51	0.41	0.78

While the average utilization, PUE, and power demand are similar to the base case scenario (see Table 9), power consumption differs significantly. Figure 30 depicts the power consumption curves of all the EDCs of the edge computing federation. The behavior of the *Strict* storage controller policy is also displayed in this figure. Additionally, blue areas highlight periods when EDCs return energy to the grid.

The power consumption curves show several abrupt changes throughout the simulation. These changes are mainly caused by the *Strict* storage controller policy of the scenario and coincide with changes in the EDCs' battery capacity. For example, from 3:00 AM to 5:00 AM, EDCs consume 1.78 kW more to charge their batteries. Then, at 7:00 AM, EDCs are allowed to discharge their battery. Thus, their power consumption drops to 0 W. Note that, at 7:30 AM, the power demand of EDC edc_2 is slightly higher than 1.78 kW (i.e., the maximum discharge power of the battery). Thus, its power consumption presents a small peak of 200 kW. From 8:00 AM to 5:00 PM, the PV systems of the EDCs generate a considerable amount of power (see Fig. 27), resulting in a reduction in power consumption. At 10:00 AM, the EDCs charge their batteries again, leading to a power consumption peak. From 11:00 AM to 4:00 PM, all the batteries are fully charged. However, the power demand of EDC edc_1 is sometimes lower than the power generated by its PV system, leading to short periods of energy returned to the grid. At the end of the simulation, EDC edc_1 returned 16.11 Wh to the grid. Overall, EDCs edc_0, edc_1, and edc_2 reduced their energy consumption by 20.7%, 27.3%, and 16.0% compared to the base case scenario, respectively.

Figure 31 compares the accumulated energy cost of the EDCs in the selected scenario. The figure depicts the behavior of the *Strict* storage controller policy as in the previous figures.When entering the green areas, the slope of the accumulated increments because the storage controllers consume additional energy to charge their

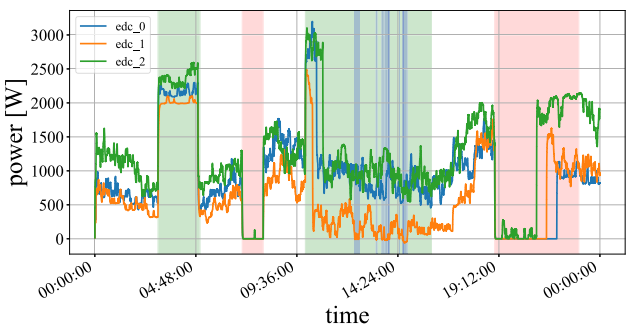

Fig. 30 Power consumption per EDC in smart grid scenario

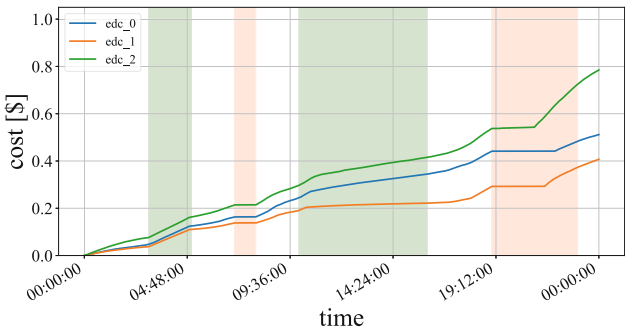

Fig. 31 Accumulated energy cost per EDC in smart grid scenario

corresponding storage unit. Once the storage units are completely charged, the slope of the curves decreases again. In contrast, when the curves enter a red area (i.e., storage controllers allow the use of energy stored in the storage units), the accumulated cost curves stagnate while batteries discharge. Once the storage units are discharged, the slope of the curves increments again. From 8:00 AM to 7:00 PM, the cost curves standstill due to the electricity generated by the PV systems. Overall, EDCs edc_0, edc_1, and edc_2 reduced their energy costs by 32.3%, 36.4%, and 25.2%, respectively. These results show that, with the proposed smart grid-aware architecture, EDCs can achieve greater cost savings than the power consumption decrease.

5.4 Discussion

While edge computing aims to significantly improve the QoS of applications that require computation offloading services while reducing the carbon footprint of state-of-the-art cloud facilities, there is still room for improvement. The architecture proposed in this research integrates aspects of smart grids to provide edge computing infrastructure with new mechanisms to reduce both its energy consumption and associated costs. EDCs include energy sources and storage units to reduce their electricity consumption. Furthermore, they behave as smart grid consumers and change their power consumption patterns depending on the current electricity price. We extended the Mercury M&S&O framework to include this new proposed architecture. Now, Mercury allows us to explore the effect of different resource management techniques on the power consumption of the solution. Additionally, we can use Mercury to determine which electricity source and storage infrastructures our scenario may need. We presented several scenarios to illustrate our proposal and simulated them using Mercury. The simulation results show that cooperation between edge computing resources and smart grids can significantly reduce the energy consumption of the next-generation computing infrastructures.

Note that the integration of smart grid infrastructures presented in this work is compatible with other studies with different approaches regarding the architecture and usage of edge computing infrastructures. These works can extend the behavior of edge nodes to include smart grid-related features and reduce their associated energy consumption and costs. With this regard, the hierarchical and modular approach of the DEVS formalism eased the integration of the presented model in Mercury. In the context of the M&S of complex systems, it is vital to follow a robust and modular approach that helps us when extending its functionality or reiterating its specification.

6 Conclusion

We proposed an M&S&O approach for studying federated edge computing infrastructures connected to smart grids. The model presented in this paper allows us to define federated edge computing infrastructures that dynamically adapt to the current system load to optimize their energy consumption while meeting the required QoS. The EDCs of the federation integrate DTs of their resources to enforce high-level resource management policies. Furthermore, the EDCs can dynamically change these policies depending on their current state and the predicted service demand. If the edge computing infrastructure runs gets congested, it can redirect new requests to a cloud facility with backup computation resources. Additionally, we integrate smart grid-related parameters in the resource management policies of the EDCs to reduce the electricity consumption and overall cost of the federation. By doing so, we enable more sustainable edge computing infrastructures that rely on smart grid advantages (e.g., electricity demand curve flattening or self-generation) to reduce their carbon footprint and increase their economic feasibility.

We implemented this model in Mercury, an M&S&O framework based on the DEVS mathematical formalism. With Mercury, it is possible to run multiple simulations of scenarios with different configurations (e.g., IT resources, energy storage units, or energy consumption policies) and compare their performance to find an optimal system design. To illustrate how Mercury works, we presented a realistic use case scenario of an edge computing federation that provides computation offloading services to taxis in the San Francisco Bay Area. We showed how the proposed approach helps edge computing federation operators to increase their resilience to electricity price fluctuations, reducing electricity consumption by 20.3% and operational expenses by 30.3%.

In future work, we plan to include regional clouds in the edge computing model with more complex interrelationships. We also want to add temperature models for PUs to

integrate the temperature into the resource management policies. Doing so can avoid hot spots within an EDC that may damage the IT equipment. Regarding the smart grid model, we want to expand the capabilities of the electricity providers to enable more sophisticated pricing schemes. We are currently working on adding a decision support system to Mercury. This system will automatically optimize multiple parameters of the scenario using different metaheuristic search methods (e.g., simulated annealing or Tabu search).

Author Contributions All authors contributed to the study conception and design. Material preparation, data collection and analysis were performed by RC. The first draft of the manuscript was written by RC and all authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

Funding Open Access funding provided thanks to the CRUE-CSIC agreement with Springer Nature. This paper has been supported by the Spanish Ministry of Economic Affairs and Digital Transformation (MINECO) under Grant PID2019-110866RB-I00. We also would like to thank Google, Inc. for giving the Google Cloud Platform (GCP) Education Grant for this investigation.

Data availability The datasets generated and/or analyzed during the current study are publicly available in Mercury's GitHub repository: https://github.com/greenlsi/mercury_mso_framework.

Declarations

Conflict of interest The authors have no relevant financial or non-financial interests to disclose.

References

1. Gartner Survey Reveals 47% of Organizations Will Increase Investments in IoT Despite the Impact of COVID-19. Gartner, Inc. (2020). https://www.gartner.com/en/newsroom/press-releases/2020-10-29-gartner-survey-reveals-47-percent-of-organizations-will-increase-investments-in-iot-despite-the-impact-of-covid-19-
2. Stergiou, C., Psannis, K.E., Kim, B.-G., Gupta, B.: Secure integration of IoT and cloud computing. Future Gener. Comput. Syst. **78**, 964–975 (2018). https://doi.org/10.1016/j.future.2016.11.031
3. Chang, H., Hari, A., Mukherjee, S., Lakshman, T.V.: Bringing the cloud to the edge. In: 2014 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), 2014, pp. 346–351 (2014). https://doi.org/10.1109/INFCOMW.2014.6849256
4. Shi, W., Cao, J., Zhang, Q., Li, Y., Xu, L.: Edge computing: vision and challenges. IEEE Internet Things J. **3**(5), 637–646 (2016). https://doi.org/10.1109/JIOT.2016.2579198
5. Pan, J., McElhannon, J.: Future edge cloud and edge computing for Internet of Things applications. IEEE IoT J. **5**(1), 439–449 (2018). https://doi.org/10.1109/JIOT.2017.2767608
6. Shi, W., Cao, J., Zhang, Q., Li, Y., Xu, L.: Edge computing: vision and challenges. IEEE IoT J. **3**(5), 637–646 (2016). https://doi.org/10.1109/JIOT.2016.2579198
7. Dileep, G.: A survey on smart grid technologies and applications. Renew. Energy **146**, 2589–2625 (2020). https://doi.org/10.1016/j.renene.2019.08.092
8. Feng, C., Wang, Y., Chen, Q., Strbac, G., Kang, C.: Smart grid encounters edge computing: opportunities and applications. Adv. Appl. Energy (2020). https://doi.org/10.1016/j.adapen.2020.100006
9. Jimenez-Castillo, G., Tina, G., Munoz-Rodriguez, F., Rus-Casas, C.: Smart meters for the evaluation of self-consumption in zero energy buildings. In: 2019 10th International Renewable Energy Congress (IREC), 2019, pp. 1–6. IEEE (2019). https://doi.org/10.1109/IREC.2019.8754609
10. Oprea, S.V., Bâra, A., Ifrim, G.: Flattening the electricity consumption peak and reducing the electricity payment for residential consumers in the context of smart grid by means of shifting optimization algorithm. Comput. Ind. Eng. **122**, 125–139 (2018). https://doi.org/10.1016/j.cie.2018.05.053
11. Friedenthal, S., Moore, A., Steiner, R.: A Practical Guide to SysML: The Systems Modeling Language, 3rd edn. Elsevier, Amsterdam (2015). ISBN 978-0-12-800202-5
12. Mittal, S., Tolk, A.: Complexity Challenges in Cyber Physical Systems: Using Modeling and Simulation (M&S) to Support Intelligence, Adaptation and Autonomy. Stevens Institute Series on Complex Systems and Enterprises. Wiley, New York (2019). ISBN 9781119552468
13. Zeigler, B.P., Muzy, A., Kofman, E.: Theory of Modeling and Simulation: Discrete Event and Iterative System Computational Foundations, 3rd edn. Academic, San Diego (2019). ISBN 978-0-12-813370-5
14. Cárdenas, R., Arroba, P., Blanco, R., Malagón, P., Risco-Martín, J.L., Moya, J.M.: Mercury: a modeling, simulation, and optimization framework for data stream-oriented IoT applications. Simul. Model. Pract. Theory **101**, 102037 (2020). https://doi.org/10.1016/j.simpat.2019.102037. (**Modeling and Simulation of Fog Computing**)
15. Cárdenas, R., Arroba, P., Martín, J.L.R.: Bringing AI to the edge: a formal M&S specification to deploy effective IoT architectures. J. Simul. (2021). https://doi.org/10.1080/17477778.2020.1863755
16. Khan, W.Z., Ahmed, E., Hakak, S., Yaqoob, I., Ahmed, A.: Edge computing: a survey. Future Gener. Comput. Syst. **97**, 219–235 (2019). https://doi.org/10.1016/j.future.2019.02.050
17. Dong, Y., Guo, S., Liu, J., Yang, Y.: Energy-efficient fair cooperation fog computing in mobile edge networks for smart city. IEEE IoT J. **6**(5), 7543–7554 (2019). https://doi.org/10.1109/JIOT.2019.2901532
18. Etemadi, M., Ghobaei-Arani, M., Shahidinejad, A.: A cost-efficient auto-scaling mechanism for IoT applications in fog computing environment: a deep learning-based approach. Clust. Comput. (2021). https://doi.org/10.1007/s10586-021-03307-2
19. Al-Zoubi, K., Wainer, G.: Fog and cloud collaboration to perform virtual simulation experiments. Simul. Model. Pract. Theory **101**, 102032 (2020). https://doi.org/10.1016/j.simpat.2019.102032

20. gan Zhang, D., hao Ni, C., Zhang, J., Zhang, T., Yang, P., xuWang, J., ran Yan, H.: A novel edge computing architecture based on adaptive stratified sampling. Comput. Commun. **183**, 121–135 (2022). https://doi.org/10.1016/j.comcom.2021.11.012
21. Dong, R., She, C., Hardjawana, W., Li, Y., Vucetic, B.: Deep learning for hybrid 5G services in mobile edge computing systems: learn from a Digital Twin. IEEE Trans. Wirel. Commun. **18**(10), 4692–4707 (2019). https://doi.org/10.1109/TWC.2019.2927312
22. Tao, F., Zhang, H., Liu, A., Nee, A.Y.C.: Digital twin in industry: state-of-the-art. IEEE Trans. Ind. Inform. **15**(4), 2405–2415 (2019). https://doi.org/10.1109/TII.2018.2873186
23. Qi, Q., Tao, F.: Digital twin and big data towards smart manufacturing and Industry 4.0: 360 degree comparison. IEEE Access **6**, 3585–3593 (2018). https://doi.org/10.1109/ACCESS.2018.2793265
24. Chen, X., Lu, Z., Ni, W., Wang, X., Wang, F., Zhang, S., Xu, S.: Cooling-aware optimization of edge server configuration and edge computation offloading for wirelessly powered devices. IEEE Trans. Veh. Technology **70**(5), 5043–5056 (2021). https://doi.org/10.1109/TVT.2021.3076057
25. Zoie, R.C., DeliaMihaela, R., Alexandru, S.: An analysis of the power usage effectiveness metric in data centers. In: 2017 5th International Symposium on Electrical and Electronics Engineering (ISEEE), 2017, pp. 1–6. https://doi.org/10.1109/ISEEE.2017.8170650
26. Masanet, E., Shehabi, A., Lei, N., Smith, S., Koomey, J.: Recalibrating global data center energy-use estimates. Science **367**(6481), 984–986 (2020). https://doi.org/10.1126/science.aba375
27. Jones, N.: How to stop data centres from gobbling up the world's electricity. Nature **561**(7722), 163–167 (2018). https://doi.org/10.1038/d41586-018-06610-y
28. Ebrahimi, K., Jones, G.F., Fleischer, A.S.: A review of data center cooling technology, operating conditions and the corresponding low-grade waste heat recovery opportunities. Renew. Sustain. Energy Rev. **31**, 622–638 (2014). https://doi.org/10.1016/j.rser.2013.12.007
29. Li, J., Zhou, G., Tian, T., Li, X.: A new cooling strategy for edge computing servers using compact looped heat pipe. Appl. Therm. Eng. **187**, 116599 (2021). https://doi.org/10.1016/j.applthermaleng.2021.116599
30. Qayyum, T., Malik, A.W., Khattak, M.A.K., Khalid, O., Khan, S.U.: FogNetSim++: a toolkit for modeling and simulation of distributed fog environment. IEEE Access **6**, 63570–63583 (2018). https://doi.org/10.1109/ACCESS.2018.2877696
31. Lera, I., Guerrero, C., Juiz, C.: YAFS: a simulator for IoT scenarios in fog computing. IEEE Access **7**, 91745–91758 (2019). https://doi.org/10.1109/ACCESS.2019.2927895
32. Brogi, A., Forti, S.: QoS-aware deployment of IoT applications through the fog. IEEE IoT J. **4**(5), 1185–1192 (2017). https://doi.org/10.1109/JIOT.2017.2701408
33. Gupta, H., VahidDastjerdi, A., Ghosh, S. K., Buyya, R.: iFogSim: a toolkit for modeling and simulation of resource management techniques in the Internet of Things, edge and fog computing environments. Softw. Pract. Exp. **47**(9), 1275–1296 (2017). https://doi.org/10.1002/spe.2509
34. Sonmez, C., Ozgovde, A., Ersoy, C.: EdgeCloudSim: an environment for performance evaluation of edge computing systems. Trans. Emerg. Telecommun. Technol. (2018). https://doi.org/10.1002/ett.3493
35. Zeng, X., Garg, S.K., Strazdins, P., Jayaraman, P.P., Georgakopoulos, D., Ranjan, R.: IOTSim: a simulator for analysing IoT applications. J. Syst. Archit. **72**, 93–107 (2017). https://doi.org/10.1016/j.sysarc.2016.06.008
36. Greer, C., Wollman, D., Prochaska, D., Boynton, P., Mazer, J., Nguyen, C., FitzPatrick, G., Nelson, T., Koepke, G., Hefner, A., Pillitteri, V., Brewer, T., Golmie, N., Su, D., Eustis, A., Holmberg, D., Bushby, S.: NIST Framework and Roadmap for Smart Grid Interoperability Standards, Release 4.0 (Draft). NIST, Gaithersburg (2021)
37. Cintuglu, M.H., Mohammed, O.A., Akkaya, K., Uluagac, A.S.: A survey on smart grid cyber–physical system testbeds. IEEE Commun. Surv. Tutor. **19**(1), 446–464 (2017). https://doi.org/10.1109/COMST.2016.2627399
38. Ahmed, N., Levorato, M., Li, G.P.: Residential consumer-centric demand side management. IEEE Trans. Smart Grid **9**(5), 4513–4524 (2018). https://doi.org/10.1109/TSG.2017.2661991
39. Hu, R.L., Skorupski, R., Entriken, R., Ye, Y.: A mathematical programming formulation for optimal load shifting of electricity demand for the smart grid. IEEE Trans. Big Data **6**(4), 638–651 (2020). https://doi.org/10.1109/TBDATA.2016.2639528
40. Varghese, A.C., Padmini, V., Kumar, G., Khaparde, S.A.: Smart grid consumer behavioral model using machine learning. In: International Conference on Innovative Smart Grid Technologies, ISGT Asia 2018, 2018, pp. 734–739. IEEE (2018). ISBN 9781538642917. https://doi.org/10.1109/ISGT-Asia.2018.8467824
41. Yang, J., Zhao, J., Luo, F., Wen, F., Dong, Z.Y.: Decision-making for electricity retailers: a brief survey. IEEE Trans. Smart Grid **9**(5), 4140–4153 (2018). https://doi.org/10.1109/TSG.2017.2651499
42. Vaubourg, J., Presse, Y., Camus, B., Bourjot, C., Ciarletta, L., Chevrier, V., Tavella, J.-P., Morais, H.: Multi-agent multi-model simulation of smart grids in the MS4SG project. In: Demazeau, Y., Decker, K.S., Bajo Pérez, J., de la Prieta, F. (eds) Advances in Practical Applications of Agents, Multi-agent Systems, and Sustainability: The PAAMS Collection, pp. 240–251. Springer (2015). ISBN 978-3-319-18944-4. https://doi.org/10.1007/978-3-319-18944-4_20
43. Lin, H., Veda, S.S., Shukla, S.S., Mili, L., Thorp, J.: GECO: global event-driven co-simulation framework for interconnected power system and communication network. IEEE Trans. Smart Grid **3**(3), 1444–1456 (2012). https://doi.org/10.1109/TSG.2012.2191805
44. Rohjans, S., Lehnhoff, S., Schütte, S., Scherfke, S., Hussain, S.: Mosaik—a modular platform for the evaluation of agent-based Smart Grid control. In: 2013 4th IEEE/PES Innovative Smart Grid Technologies Europe, ISGT Europe 2013, pp. 1–5 (2013). ISBN 9781479929849. https://doi.org/10.1109/ISGTEurope.2013.6695486
45. Samie, F., Bauer, L., Henkel, J.: Edge Computing for Smart Grid: An Overview on Architectures and Solutions, pp. 21–42. Springer, Cham (2019). ISBN 978-3-030-03640-9
46. Huang, Y., Lu, Y., Wang, F., Fan, X., Liu, J., Leung, V.C.: An edge computing framework for real-time monitoring in smart grid. In: Proceedings—2018 IEEE International Conference on Industrial Internet, ICII 2018, 2018, no. icii, pp. 99–108. IEEE (2018). ISBN 9781538677711. https://doi.org/10.1109/ICII.2018.00019
47. Liu, Y., Yang, C., Jiang, L., Xie, S., Zhang, Y.: Intelligent edge computing for IoT-based energy management in smart cities. IEEE Netw. **33**(2), 111–117 (2019). https://doi.org/10.1109/MNET.2019.1800254
48. Gai, K., Wu, Y., Zhu, L., Xu, L., Zhang, Y.: Permissioned Blockchain and edge computing empowered privacy-preserving smart grid networks. IEEE IoT J. **6**(5), 7992–8004 (2019). https://doi.org/10.1109/JIOT.2019.2904303
49. Cárdenas, R., Arroba, P., Moya, J.M., Risco-Martín, J.L.: Multi-faceted modeling in the analysis and optimization of IoT complex systems. In: Proceedings of the 2020 Summer Simulation

Conference. Virtual Event, July 2020, pp. 1–12. Society for Computer Simulation International (2020)
50. Cárdenas, R.: Mercury M&S&O Framework for Fog Computing. https://github.com/greenlsi/mercury_mso_framework
51. Jäger-Waldau, A., Bucher, C., Frederiksen, K.H.B., Guerro-Lemus, R., Mason, G., Mather, B., Mayr, C., Moneta, D., Nikoletatos, J., Roberts, M.B.: Self-consumption of electricity produced from PV systems in apartment buildings—comparison of the situation in Australia, Austria, Denmark, Germany, Greece, Italy, Spain, Switzerland and the USA. In: 2018 IEEE 7th World Conference on Photovoltaic Energy Conversion (WCPEC) (A Joint Conference of 45th IEEE PVSC, 28th PVSEC, 34th EU PVSEC), 2018, pp. 1424–1430 (2018). https://doi.org/10.1109/PVSC.2018.8547583
52. Piorkowski, M., Sarafijanovoc-Djukic, N., Grossglauser, M.: A parsimonious model of mobile partitioned networks with clustering. In: The First International Conference on COMmunication Systems and NETworkS (COMSNETS), January 2009 (2009). https://doi.org/10.1109/COMSNETS.2009.4808865
53. Pérez, S., Pérez, J., Arroba, P., Blanco, R., Ayala, J.L., Moya, J.M.: Predictive GPU-based ADAS management in energy-conscious smart cities. In: 2019 IEEE International Smart Cities Conference (ISC2), 2019, pp. 349–354. IEEE (2019). https://doi.org/10.1109/ISC246665.2019.9071685

Román Cárdenas received the M.Sc. Degree in Telecommunication Engineering in 2019 from Technical University of Madrid (UPM), Spain, where he holds an FPU Fellowship to pursue the Ph.D. in Electronic Systems Engineering in Cotutelle with Carleton University (CU). His research interests include modeling and simulation with applications in the IoT domain.

Patricia Arroba is an Assistant Professor at the Technical University of Madrid (UPM), Spain. She received her Ph.D. Degree in Telecommunication Engineering from UPM in 2017. Her research interests include energy and thermal-aware modeling and optimization of data centers.

José L. Risco-Martín received his Ph.D. from Complutense University of Madrid (UCM), Spain, where he currently is Full Professor in the Department of Computer Architecture and Automation. His research interests include computer-aided design and modeling, simulation, and optimization of complex systems.

José M. Moya is an Associate Professor in the Technical University of Madrid (UPM). He received his Ph.D. Degree in Telecommunication Engineering from UPM in 2003. His research interests include proactive and reactive thermal-aware optimization of data centers.