



TECHNICAL UNIVERSITY
OF CLUJ-NAPOCA, ROMANIA

Computer Science

PhD THESIS

- ABSTRACT -

Contributions to the Development of Orchestration Solutions for Computational Continuum over the Smart Grid

PhD Student:
Gabriel Ioan Arcaş

PhD Supervisor:
Prof. Eng. Tudor Cioara

- Cluj-Napoca -
2025

1. Introduction and Motivation

The integration of digital technologies in the energy sector is redefining how infrastructure is understood, designed, and managed. Shifting from centralized energy management systems demands not only hardware innovation but a new approach to embedding intelligence across distributed environments. Academically, this opens space for research: How can we develop models and decision mechanisms that are computationally and communicational efficient to respond closer to real-time to fluctuating conditions?

As more data will be generated at the edge of the energy grid the traditional cloud-based management architectures will become inefficient for smart grid needs. Latency, bandwidth constraints, and the centralized nature of cloud processing often affect the speed decisions and actions execution. This is especially relevant in energy systems where near real-time responsiveness is important to ensure the continuity of energy delivery for customers. Important management tasks or services such as power flow control, load balancing, or even predictive maintenance of assets require fast, context-aware decisions that can't afford to "wait for the cloud."

From a theoretical perspective, this work dives into computational implications of managing smart grid energy assets using state of the art models such as the Digital Twins (DTs) while the computational continuum distribution of resources and data. To reflect the energy and computational systems real-time states, we create digital models using Directed Acyclic Graphs (DAGs) that clarify energy and computational resources as well as ask dependencies. To ease the optimization and decision-making process for workload offloading we have considered Knowledge Graphs (KGs) that add semantic reasoning into operational workflows executed at the intersection of the energy and computing continuum networks. These models aren't just passive representations; they actively support workflow orchestration across the entire computational continuum, as supported by recent work on task-aware DAGs for real-time edge computing [25], semantic models in smart grids [26], and hybrid orchestration platforms based on Digital Twins [30].

Therefore, in this research, we explore the computing continuum, a coordinated architecture across edge, fog, and cloud layers, as a formal paradigm that reflects this need for distributed intelligence in smart grid energy systems. The challenge is not only technical, but conceptual: how to ensure responsiveness and contextual awareness across the computational continuum while considering the energy grid topology and services requirements.

In industry, the transformation is operational and urgent. Companies working on large-scale smart infrastructure, mobility, and energy must deal with ever-increasing data volumes, geographically distributed assets, and stringent real-time constraints. Systems must now autonomously decide where to execute tasks, how to route information, and when to act, all within milliseconds. My own experience in industry projects, including with Bosch, revealed a significant mismatch between what current orchestration systems offer and what real-time applications demand. For instance, decentralized energy resources or environmental sensors generate continuous data streams, yet central systems struggle to interpret and act on this data in time. Even minor delays in reacting to air quality changes or energy spikes can lead to inefficiencies or regulatory penalties.

In response, this thesis explores energy and data-aware orchestration across the computing continuum by combining the digital models that reflect the energy and computational state of the system with optimization processes to support decision making. In this thesis, we have explored two types of optimization processes based on heuristics and learning. Heuristics offer a more adaptable way to find near-optimal solutions efficiently, making them a better choice

for managing DERs, optimizing energy dispatch, and balancing computational loads across the smart grid infrastructure. Therefore, we have provided in developed a Whale Optimization based heuristic on top of the DAG model for offloading the computation workload closer to the edge.

Alternatively, we have developed reinforcement learning solution for edge offloading integrated with the KG model for learning based on simulations from past interactions, actions and outcomes, to enhance efficiency, minimize latency, and optimize the utilization of available computational resources.

The offloading and orchestration platform was built with scalability, efficient integration, and practical deployment in mind, leveraging and integrating the good practices of Lambda Architecture. Furthermore, the inclusion of real use cases such as fuel-mixing based on air quality and energy forecasting reflects a direct response to industrial challenges. These use cases were not chosen arbitrarily; they are driven by actual industry needs, where system responsiveness and transparency have economic and environmental impact. Therefore, this thesis integrates academic innovations in modelling, AI, and reasoning with industrial constraints such as deployment complexity, sensor integration, and system stability under pressure. Each model, algorithm, and architectural choice was tested using real-world datasets and practical implementations. This ensures the research contributes not only conceptually but also by delivering functional tools for building scalable, and adaptive systems.

The main objective of this thesis is to contribute to the development of computational orchestration platform for supporting the distribution in efficient manner of the smart grid energy management tasks across the overlapping across edge, fog, and cloud computational layers. The platform considers computational continuum modelling over the smart grid, offloading optimization decision making through heuristics and reinforcement learning to manage resource allocation dynamically. Validated through real-world use cases, it offers both conceptual and applied contributions that bridge academic innovation with industrial relevance.

Along with our research we considered a layered modeling strategy that uses Digital Twins with energy assets in the smart grid and DAGs, and Knowledge Graphs to capture computational distribution states at the intersection of computational continuum with the smart energy grid. The models are enriched with semantic rules to support intelligent decision-making. To optimize computational task allocation, we explored two complementary strategies: a model-based offloading method using the Whale Optimization Algorithm for low-latency environments, and a learning-based strategy using Deep Q-Learning and Advantage Actor-Critic for dynamic, evolving topologies. These techniques are implemented within a modular orchestration platform built on the Lambda Architecture. The platform is validated with two real-world use cases, one for predicting energy consumption and another for adjusting fuel mixes based on air quality data. Stress testing confirms the platform's adaptability, scalability, and industrial applicability. By unifying these elements, the thesis delivers both novel models and a practical orchestration system, a balanced integration of academic practice and industrial deployment.

2. Current state of knowledge

In Chapter 2, we analysed the existing state of the art in three main directions identified at the intersection of the computational continuum with smart grid decentralized energy management.

We first explored modeling strategies used to represent energy and computation systems in a coordinated way. Most of the reviewed work relies on partial models: Digital Twins [30], [32], DAGs for structuring task flows in latency-sensitive environments [25], and Knowledge Graphs for semantic contextualization [26], [27]. However, very few studies bring all three together into a unified orchestration framework [29]. We identified a gap in the ability of existing models to represent structure, behavior, and meaning at the same time, something necessary when dealing with real-time orchestration across edge, fog, and cloud.

The second part of the chapter investigates how tasks are moved, or offloaded, across these layers. Traditional offloading approaches have matured, but they still struggle with scenarios that change quickly or require adaptability across layers. However, Deep Q-Learning and A2C started to gain more popularity. These models improve task allocation by learning from system feedback. They adjust decisions based on real-time system conditions such as latency, resource usage, or bandwidth. Still, most of these techniques don't connect well with the structural or semantic models mentioned earlier, leaving a disconnect that our work aims to fix.

The last direction of this chapter was on orchestration platforms. These platforms tie everything together and are expected to operate reliably in dynamic environments. We observed that many current orchestration engines, though powerful in cloud contexts, struggle when applied to edge deployments. Edge resources are constrained, networks can be unpredictable, and real-time decision-making becomes harder. That's why orchestration systems must go beyond traditional centralized deployment models. We analyzed recent trends in container orchestration, modular service design, and event-triggered workflows. We also highlighted how Lambda Architecture provides a solid base for combining batch and real-time processing, which is important for the smart grid scenarios.

Beyond the three main focuses, this chapter also addresses specific challenges that arise in the deployment of these models. For instance, modeling smart grid environments is not just about describing nodes and links, but also about ensuring that computational decisions reflect the operational constraints of physical systems. Smart grids operate under policies that govern not only resource use but also legal, safety, and contractual rules. These often go unnoticed in purely algorithmic approaches. Hence, semantic modeling becomes essential.

We noticed a growing trend in using Knowledge Graphs to formalize these contextual rules. By encoding relationships and policies as graph-based ontologies, systems can reason not just about performance, but also about compliance and priorities. For example, if a node belongs to an important infrastructure zone, this may alter its eligibility for task offloading. Or if certain services must remain co-located due to privacy restrictions, the graph structure can help enforce this.

We also studied the use of DAGs in the orchestration context. While most orchestration engines use DAGs for describing tasks and workflows, few of them use annotated DAGs with rich metadata that reflects network conditions or resource constraints. In our work, DAGs are enriched with attributes such as CPU, memory, distance, latency, and bandwidth. This allows the DAG not only to define dependencies but to guide decisions about where and when those dependencies should be executed.

In parallel, we examined Digital Twins as models that reflect the dynamic state of physical components. DTs have mostly been explored in the context of Industry 4.0. In the smart grid domain, their use is less popular, but we found some interesting early deployments that use DTs to track battery performance, energy flow, and fault detection. Still, most of these use cases are detached from orchestration pipelines. We aim to integrate DTs as live feeds into decision-making.

Our analysis of orchestration platforms showed that edge orchestration is still a relatively new field. Cloud-native technologies like Kubernetes and Argo Workflows have matured, but extensions like KubeEdge or KubeStellar are still evolving. We identified the need for platforms that can operate in a hybrid mode, managing both stable, centralized resources and volatile, constrained edge environments. We also emphasized the role of lightweight containers, fast communication protocols like MQTT, and distributed event handling for real-time orchestration.

As part of the modeling literature, we explored graph-based extensions such as RDF and SPARQL for querying semantic models. This combination of structured querying with reasoning allows orchestration systems to evaluate real-time constraints without relying solely on pre-coded logic. These tools enable adaptive querying, such as retrieving all nodes that match a particular condition (e.g., “all fog nodes with CPU > 2GHz and latency < 50ms”) without hardcoding those parameters into the workflow logic.

In the area of offloading strategies, we found a large volume of research on reinforcement learning, which aims to adapt decision-making based on feedback. However, we also noticed limitations: RL models typically assume a well-structured environment and need time to converge. In dynamic systems like smart grids, that time may not be available. Hybrid models, where reinforcement learning is combined with heuristics, appear to offer a better trade-off.

Our analysis also considered the performance trade-offs in different offloading strategies. Metaheuristics like WOA and PSO provide faster decisions but are less flexible in the face of changing system conditions. RL-based methods are more adaptable but computationally heavier. In Chapter 4, we proposed using both, depending on the scenario. That conclusion is based on this literature review.

Additionally, the literature pointed to emerging trends such as task migration, cooperative offloading, and federated orchestration. These are advanced forms of orchestration where tasks can shift between nodes during execution, or where multiple nodes share parts of the same task. We took inspiration from these works to consider how orchestration can be made more resilient in the case of partial failures or changing availability.

The review of semantic orchestration also pointed to the benefits of modularity. Systems that decouple logic from infrastructure can adapt more easily when new devices are added or when topology changes. Our use of YAML-based descriptors for nodes and links follows this idea. We want our orchestration platform to adjust configurations without having to rewrite logic.

Another relevant discovery was how different orchestration platforms approach scheduling under uncertainty. Some rely on static rules, while others use runtime analytics to adjust execution. We found that combining event-driven triggers with prediction models offers a good balance between responsiveness and foresight. This principle is used in our Lambda-inspired architecture.

In the context of high availability, we found that most cloud orchestration solutions still assume stable, always-on connectivity. In edge environments, that is not always true. This means that orchestration logic must include retry strategies, data buffering, and fallback mechanisms. Our platform integrates these features by allowing components to store decisions temporarily and synchronize later.

Chapter 2 clarified where existing approaches are limited and pointed to the need for a more integrated, flexible, and semantically aware orchestration framework. This motivated the design choices introduced in the following chapters. The research gaps identified here directly informed the hybrid modeling and offloading methods we developed, as well as the architecture of the orchestration platform itself.

Finally, this chapter sets the current state of knowledge for the practical work that follows. We not only reviewed models and techniques, but also analyzed how they relate to real-world deployment constraints. Our synthesis of these findings shaped a solution that is both grounded in current research and oriented toward future scalability, drawing on prior orchestration models [25], knowledge-driven reasoning [28], and smart grid simulation environments like CloudIEPS [32].

3. Distributed Computational Resources Models for Smart Grid

In Chapter 3, we introduced the modeling techniques that support the orchestration logic proposed in this thesis. The goal was to provide a structured way to describe both the computational resources and the behavior of the smart grid system using formal, adaptable models. This chapter represents the theoretical backbone of the orchestration platform and explains how different representations, Digital Twins, Directed Acyclic Graphs (DAGs), and Knowledge Graphs, are brought together into a hybrid model that captures system state, task structure, and semantic context.

The motivation for developing this model came from the limitations observed in current orchestration solutions. Many platforms can schedule tasks or react to events, but they lack a way to reason about why a task should be assigned to a certain node, what that node represents in the physical world, or how node relationships change over time. Existing approaches typically model computational nodes or physical assets in isolation. Our framework bridges this gap by defining a set of annotated models that unify the physical layout of smart grids with their digital and computational components.

DTs are useful for improving the management, operation, and coordination of various distributed energy assets. For example, energy storage systems are important in renewable energy grids, storing excess energy and releasing it when demand increases. DTs make it possible for operators to track important metrics like state of charge (SOC) or temperature changes. This data helps predict maintenance needs and prevent unexpected failures. Additionally, DTs simulate charging and discharging cycles to identify efficient strategies for seasonal demand fluctuations or sudden energy peaks, improving both efficiency and asset longevity. Finally, advanced coordination and optimization strategies can be implemented, allowing the coordination of multiple small-scale storage systems to address grid stability and balancing problems.

DTs can address the challenges that come with the growing use of EVs. The integration of EVs into the grid creates unpredictable demand patterns, particularly during peak charging hours. DTs provide intelligent tools to simulate charging patterns across various locations, optimizing schedules and reducing grid stress during peak times. They also model shared charging infrastructure, such as that in residential or corporate settings, to optimize placement and capacity while forecasting the long-term impact of increased EV usage on local grids. These capabilities assist planners in developing infrastructure that aligns with future demand without over-investing. By simulating how combined resources behave under different conditions, DTs help ensure that renewable energy is used efficiently while cutting back on the need for traditional power sources. Moreover, DTs support demand response programs by dynamically redistributing energy based on live grid conditions. These simulations reduce grid strain during peak periods while leveraging advanced analytics and machine learning to continuously improve efficiency and performance.

The integration of DTs into energy systems also enhances cybersecurity and resilience. By analyzing data for early signs of equipment degradation or potential cyber threats, DTs enable proactive interventions, minimizing disruptions and protecting grid infrastructure. This

capability is important as the connectivity between IoT devices and digital systems grows, exposing the grid to more sophisticated security risks. DTs extend their functionality through machine learning, blockchain, and edge computing. Machine learning enhances the predictive capabilities of DTs by identifying patterns in historical and real-time data, while blockchain ensures secure data exchange and transparency. Combined with the computational continuum, these technologies enable DTs to scale and adapt to the evolving demands of modern energy systems.

Moreover, we introduced the DAG-based resource representation to tackle these challenges, as show in Figure 1. In our case, the DAG is not just a technical graph for managing workflows; it's an annotated structure where each node represents a computing unit (edge, fog, or cloud), and each link encodes a communication path, enriched with metadata. This metadata includes latency, bandwidth, physical distance, and resource specifications like CPU, RAM, and storage. With this setup, the orchestration system understands the network as a living, measurable structure, allowing decisions to consider both performance constraints and topological awareness.

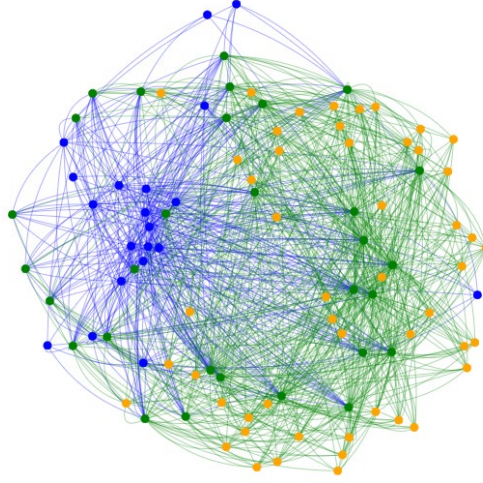


Figure 1. Resulted DAG for computing continuum.

Each node in the DAG is further connected to a Digital Twin that mirrors the behavior or operational state of a physical entity (such as a sensor, a battery, or a charging station). This connection is not static. The Digital Twin can reflect real-time changes in the system and push updates into the orchestration layer. For example, if a node's energy capacity drops below a threshold, that status is fed into the DAG model, which updates the node's weight and influences where tasks will be scheduled. This approach ensures that orchestration remains context-aware and not detached from the physical infrastructure it manages.

Complementing the DAG and Digital Twin components, Knowledge Graphs are used to store semantic information about entities and their roles in the system. This semantic layer supports rule-based reasoning, enabling the system to evaluate constraints like "offload only to nodes in the same region" or "prioritize nodes connected to stable grid segments." Instead of encoding these rules in hard logic, they are expressed semantically and processed through dynamic queries. This adds flexibility and makes the system easier to scale or adapt to new use cases.

In the context of smart grids, KGs provide good features for mapping the computational continuum. They integrate physical energy assets with edge, fog, and cloud computational layers, capturing the hierarchical and functional relationships between these components, as shown in Figure 2. This layered structure not only reflects the distinct roles of each part of the

computational ecosystem but also aligns with the DTs' need for real-time adaptability and resource optimization. Entities in the KG are represented as nodes, while their interactions and dependencies are modeled as edges, offering a comprehensive view of system operations. An important feature of KGs is the ability to incorporate rich metadata, such as latency, bandwidth, energy consumption, and geographic proximity, into the relationships between nodes. This enables KGs to support complex queries and derive actionable insights that go beyond the capabilities of DAGs. For example, a KG can infer indirect dependencies between energy assets and computational nodes or optimize task allocation by analyzing geographic and resource-based constraints. These capabilities position KGs as a natural extension of the digital infrastructure in DTs.

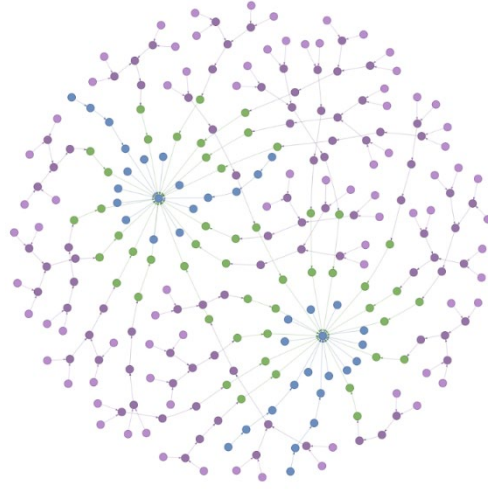


Figure 2. KG model visualization: Smart energy assets are light purple, edge nodes are dark purple, fog nodes are green, and cloud nodes are blue..

However, implementing KGs introduces distinct challenges compared to DAGs. One notable challenge is the increased computational complexity associated with processing and maintaining the semantic relationships embedded in the graph. Unlike DAGs, which mainly deal with direct dependencies, KGs need extra resources to handle ontologies, query data effectively, and stay compatible with standards like RDF and SPARQL. Furthermore, the dynamic nature of smart grids necessitates real-time updates to the KG, which can stress both computational and storage capacities. Another significant challenge involves balancing scalability and semantic richness. While KGs provide detailed contextualization, the added complexity can reduce performance in large-scale systems without proper optimization. Ensuring that the KG remains both scalable and responsive requires advanced techniques for query optimization, data indexing, and distributed graph processing.

We use YAML-based annotations and modular configurations so that new nodes, assets, or rules can be added without redesigning the system. The integration between DAGs, Digital Twins, and Knowledge Graphs is done through a layered reference model, where each layer feeds into the next, and all share a synchronized view of the system state.

This modeling framework not only supports orchestration but also provides a mechanism for system observability and fault detection. Because each component in the model can be independently monitored and annotated with live status data, the system can detect inconsistencies or performance degradation in real time. This is particularly important in infrastructures like smart grids, where early detection of hardware faults or network

congestion can prevent service interruptions or even physical damage. The DAG and DT integration allows such conditions to be propagated through the graph, updating node and link weights to influence future orchestration decisions.

Another benefit of the hybrid model is that it enables policy enforcement through semantic rules without embedding those rules directly into the orchestration logic. For example, if a new energy regulation requires that certain nodes prioritize green energy sources, this can be expressed as a rule in the Knowledge Graph and applied at runtime without changing the orchestration algorithms. This separation between logic and policy allows the platform to adapt to regulatory or organizational changes more easily, supporting long-term sustainability and interoperability.

The human-readability of the modeling framework is also a major strength. By relying on standard formats such as YAML for resource annotations and RDF for semantic relations, the model remains transparent and maintainable. Operators and developers can inspect, modify, or extend the model with minimal training. Moreover, these formats are widely supported by tools in the orchestration and IoT ecosystems, allowing for easy integration and automation across platforms. This maintainability ensures that the model can scale with deployments and be updated as hardware or system goals evolve.

During the implementation, we also focused on modularity and reuse. The modeling components are not bound to any specific orchestration engine. They can be exported, extended, and shared across deployments or between teams. For example, a Digital Twin definition for a solar panel or a battery system can be reused in different projects, as long as the underlying data format and interfaces are respected. This approach supports faster development and encourages the creation of domain-specific libraries of resources, which can accelerate adoption across sectors.

This chapter proposes a design for how to bridge the gap between physical systems, computational layers, and decision logic. By fusing structured graph representations with real-time telemetry and semantic policies, the model supports orchestration that is not only efficient but also meaningful. The orchestration engine, guided by this model, interprets the system, understands context, and reacts accordingly. This capability becomes relevant as systems grow more complex and expectations for adaptability and resilience increase.

4. Offloading decision-making techniques

In Chapter 4, we introduce background concepts related to decision-making techniques in smart grid management, emphasizing their role in overcoming energy and computational infrastructure decentralization challenges. Real-time optimization techniques in the computational continuum for smart grids leverage AI/ML to address the complexities of integrating DERs and managing computational loads. These modern computational approaches are important to achieving scalable and efficient energy management systems, yet they also bring notable computational demands. The transition to smart grids has amplified the need for advanced energy management techniques. The integration of 6G IoT, AI, and blockchain technologies is important for addressing grid stability and renewable energy integration challenges [4]. Predicting grid load fluctuations during peak demand requires advanced models, such as decision trees and artificial neural networks, which introduce computational complexity. Similarly, deep neural networks for forecasting solar energy production and reinforcement learning for photovoltaic energy management exemplify the intricate requirements of real-time optimization [4], [5].

Modern energy systems rely on renewable distributed energy resources and advanced monitoring systems, necessitating scalable techniques for handling large data volumes, model complexity, and uncertainties. Traditional methods struggle to meet these demands, while AI-driven approaches, including ML and deep learning, offer reliable solutions for dynamic security assessment and stability control [5], [6]. These computational intelligence applications are essential for ensuring timely responses to grid disturbances while maintaining system reliability [6]. AI-based management systems have demonstrated substantial gains in energy efficiency, improved grid stability, and reduced operational costs. However, the integration of historical and real-time data, coupled with optimization techniques for predictive accuracy and resource scheduling, presents substantial computational demands in decision-making processes [7]. The application of machine learning in smart grids has proven transformative, enabling real-time analysis, anticipating maintenance needs, and balancing energy loads dynamically. Yet, challenges such as data privacy, cybersecurity, and standardization in ML technologies persist. Addressing these challenges requires policy frameworks and technical advancements that promote innovation and scalability in smart grid systems. The intersection of ML and optimization techniques in IoT and smart systems underscores the need for intelligent decision-making frameworks. Enhanced energy efficiency, anomaly detection, and resource management illustrate the potential of these technologies, though they pose significant computational challenges in large-scale implementation [8].

The shift to decentralized and sustainable smart grids introduces additional obstacles. Traditional management techniques are inadequate for real-time optimization and predictive analytics, particularly with the growing integration of renewable energy and distributed resources. Advanced AI-based methods are necessary to mitigate cyber threats and ensure efficient electricity delivery [9]. Integrating AI-driven solutions into DERs presents unique challenges that include technical, economic, and ethical dimensions. It confirms the complex realities of automating decision-making processes. Coordinating DERs and integrating intermittent renewable energies requires regulatory frameworks and scalable computational models to ensure equitable energy management [10]. While AI and ML techniques offer transformative potential for decision-making in smart grids, significant computational challenges remain. Modern energy systems produce big volumes of real-time data, and their decentralized and dynamic nature depends on adaptive computational solutions capable of maintaining grid reliability, optimizing performance, and scaling effectively.

Decentralization marks a significant transformation from traditional systems, enabling setups that are both more flexible and reliable. The integration of DERs transforms the grid by enabling localized generation, storage, and consumption of energy. However, this shift brings both opportunities and challenges, necessitating distributed decision-making frameworks that leverage the computational continuum to manage real-time data processing and autonomous operations effectively [11], [12]. The transition from centralized to decentralized systems highlights significant challenges in operational, control, and data management layers. Traditional centralized control centers face vulnerabilities, including disruptions from cyberattacks or natural disasters, which can severely impact grid operations. Decentralized approaches, utilizing technologies such as autonomous smart substations and blockchain, aim to address these challenges by enhancing resilience, operability, and flexibility in energy systems [11], [13].

An important aspect of decentralization is the integration of DERs at the edge of the grid, which introduces variability and uncertainty into energy management operations. These resources, including solar panels, wind turbines, and battery storage systems, are inherently decentralized, requiring sophisticated decision-making approaches to optimize their participation in energy markets while managing these uncertainties effectively [12].

Furthermore, regulatory policies often are behind technological advancements, complicating the adoption and integration of DERs into existing grids [14]. In this context, significant advancements are needed in control and automation technologies. Existing architecture and standards often inadequately address the specific requirements for developing smart DER components, such as real-time execution, security, and interoperability. Innovative solutions focus on creating open and interoperable components to address these gaps, ensuring that renewable energy resources can be seamlessly integrated into decentralized energy systems [18].

Distributed decision-making in decentralized systems also faces computational and communicational challenges. Traditional connectivity models relying on cloud computing often struggle with high bandwidth demands and data overload. Embedding intelligence at the edge of the network through autonomous edge control devices mitigates these challenges by enabling local sensing, data processing, and control actions, reducing dependence on centralized processing [15], [17]. However, implementing such architecture requires addressing constraints related to costs, cybersecurity, and power management, which are important for the viability of decision-making systems [16].

Moreover, the transition to decentralized systems requires reconsidering long-held assumptions and infrastructure upgrades. Prosumers, or consumers who also generate energy, challenge traditional energy markets and necessitate new policies and regulatory frameworks to ensure equitable and efficient energy distribution [19]. This shift also underscores the importance of developing decentralized energy trading mechanisms, such as blockchain-based platforms and smart contracts, to enhance data sharing, transparency, and operational efficiency in energy distribution networks [14]. Leveraging the computational continuum and advancements in automation technologies will be important to overcoming energy grid decentralization challenges and realizing the full potential of the energy transition.

The integration of computational continuums within smart grids introduces significant security concerns. As smart grids evolve into complex, interconnected systems, ensuring cybersecurity becomes paramount. The vulnerabilities associated with computational continuum need to be addressed while facilitating seamless operational strategies and reliable grid functionality [20], [21].

Smart grids leverage IoT technologies to monitor power transmission and distribution. However, the interconnection between data and energy networks exposes smart grids, making them vulnerable to cyberattacks, which can compromise their reliability. To mitigate these risks, machine learning-based defense mechanisms are being adopted to detect anomalies and strengthen grid reliability. Such methods address the computational complexities of decision-making processes in the face of escalating cybersecurity threats [20], [22]. Furthermore, the extensive deployment of sensors across smart grids complicates network maintenance and exposes important components such as availability, integrity, and confidentiality to potential threats [23]. DERs and IoT have enabled bidirectional data communication and power flows. The interconnected of grid-edge devices necessitates hierarchical management strategies to ensure security and operational efficiency [23], [24]. The computational continuum provides a scalable solution for managing real-time data processing and enhancing cybersecurity. However, developing architectures capable of handling data heterogeneity and ensuring secure data fusion remains an important challenge [3]. The heterogeneity of data generated by intelligent energy generation and consumption terminals places immense pressure on centralized cloud systems. Edge computing enhances system resilience by allowing local decision-making, thereby minimizing the impact of potential cyberattacks on centralized systems. Nevertheless, this shift requires advanced security mechanisms to protect collaborative platforms and ensure user privacy in distributed environments [2]. Game and

decision theory have emerged as effective tools to model interactions among system components and between attackers and the grid. These frameworks aid in designing scalable controllers, secure network routing protocols, and effective information security management strategies. Despite these advancements, challenges in ensuring the resilience of the computing continuum persist, necessitating continuous innovation in decision-making techniques [1]. Therefore, it is essential to develop strong cybersecurity measures, adopt advanced computational methods for heterogeneous data, and design innovative system architectures that can maintain reliability and resilience against new threats.

The aim was to support distributed computing by selecting the best locations, across edge, fog, and cloud layers, for executing specific workloads. Two decision strategies were proposed and compared: a heuristic approach based on the Whale Optimization Algorithm (WOA) and a reinforcement learning method using Deep Q-Learning (DQN) and Advantage Actor-Critic (A2C) agents. The chapter builds on the earlier hybrid modeling layer, which incorporates Digital Twins, DAGs, and semantic annotations to describe energy nodes and computational capabilities.

The challenge became relevant for smart grids, where latency, bandwidth, and resource availability vary constantly. Offloading tasks too late or to a node with insufficient capacity leads to delays and resource underutilization. Conversely, aggressive or poorly targeted offloading can introduce unnecessary communication overhead or even compromise reliability. This motivated a dual approach: one fast and predictable for stable scenarios, and one adaptive for uncertain or rapidly changing conditions.

The WOA heuristic is grounded in nature-inspired optimization, specifically the bubble-net feeding strategy of humpback whales. In our implementation, each whale in the search space corresponds to a candidate offloading configuration represented by a binary vector (1 means the node is a candidate for the offload, 0 means it can remain in the same layer). The candidate solutions are updated iteratively to minimize two core objectives: round-trip time (RTT) and the computational distance between task requirements and node capabilities.

To enable this, each node in the smart grid was modeled using the DAG annotation framework introduced in Chapter 3. Nodes are assigned values for CPU speed, memory capacity, and bandwidth, while links between nodes have attributes for latency and distance. Each whale contains a specific calculated fitness value. The objective function penalizes configurations that fail to meet RTT or resource constraints, guiding the optimization toward feasible and efficient offloading plans. Figure 3 presents the offloading results based on RTT.

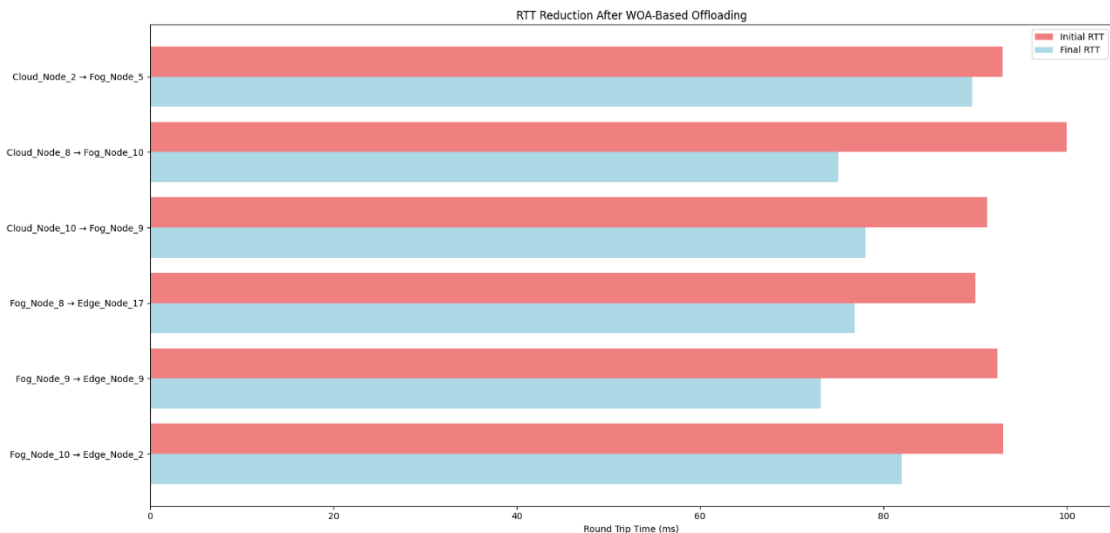


Figure 3. Task offloading results for the RTT.

WOA operates in three phases. In the encircling phase, agents adjust their position vectors by referencing the best current solution, effectively tightening the search space. The exploitation phase refines this further using a spiral motion formula to mimic the bubble-net behavior. The exploration phase introduces randomness to avoid premature convergence. We extended the classical WOA with adaptive feedback controls and inertia coefficients to enhance convergence and maintain search diversity. This allowed the algorithm to escape local optima and sustain accuracy over long execution periods.

We validated the WOA-based approach under several deployment configurations: one configuration with 50 edge and 30 fog nodes, and another with 70 edge and 35 fog nodes. In both cases, WOA demonstrated efficient convergence. For example, in Scenario 1, near-optimal solutions stabilized within 230 iterations, with a consistent execution time of 0.25 seconds per iteration. In Scenario 2, the algorithm required slightly more time, 0.5 seconds per iteration, due to the increased complexity from more nodes, but the convergence behavior remained stable.

Diversity metrics indicated that larger node pools improved solution variability but also demanded more exploration steps. Our measurements confirmed that the exploration-to-exploitation shift occurred smoothly around iteration 500 in Scenario 1 and 2700 in Scenario 2. This adaptability is important in real-time systems, where the optimization method must balance rapid response with flexible decision quality.

To handle cases where system conditions change rapidly and unpredictably, we developed an alternative learning-based strategy using reinforcement learning. The idea is to allow the orchestration agent to learn optimal offloading policies through interaction with the environment.

We modeled the orchestration environment as a knowledge graph. In this case, a node represents a computing element (edge, fog, or cloud) with annotated properties such as CPU, RAM, bandwidth, and current load. Tasks are also represented in the KG, along with links indicating dependencies, priorities, or physical constraints. This structure allows the RL agent to reason about the system as a graph traversal problem.

In our RL setup, the agent's state is defined by the current system configuration: node resources, link status, and task characteristics. The action space involves selecting a source-target node pair for offloading. The reward function is binary: successful offloading that respects RTT and resource constraints yields +1, while failure yields -1. The policy is updated to maximize cumulative rewards over time, encouraging the agent to prefer paths that consistently result in successful allocations.

We implemented and compared two RL models: Deep Q-Learning (DQN) and Advantage Actor-Critic (A2C). DQN uses a value-based learning strategy where Q-values for state-action pairs are updated using neural networks. In contrast, A2C integrates both policy and value functions, offering improved training stability and sample efficiency. To evaluate their performance, both agents were trained on an identical task graph, but with varying configurations to assess their scalability and adaptability under different system conditions.

DQN outperformed A2C in most scenarios, especially under long training windows (e.g., 500 steps per episode). It showed faster convergence and better task distribution across nodes. A2C was more effective in short-horizon cases, indicating potential in environments where interactions are limited, or rapid decisions are needed. Figure 4 shows the difference between DQN and A2C in terms of performance.

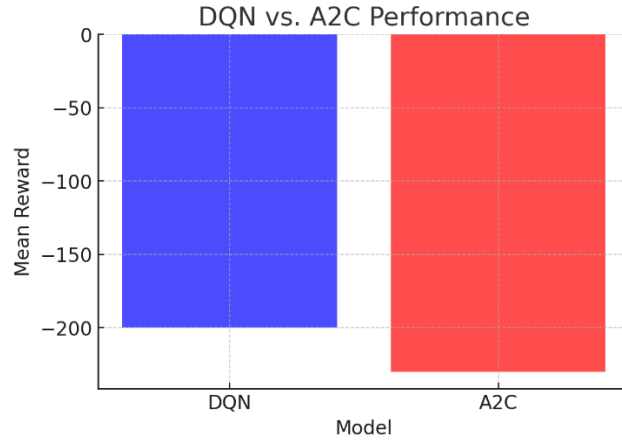


Figure 4. DQN vs A2C performance [49].

Scalability tests showed linear growth in training time as the number of nodes increased from 1000 to 1500. However, DQN maintained stable convergence across all settings, validating its adaptability.

We analyzed the offloading paths produced by the RL agent under the largest knowledge graph configuration. Most tasks were directed from smart energy assets to the edge layer, which aligns with the need for low-latency processing. Some intermediate workloads were offloaded to the fog for further processing, and a smaller fraction reached the cloud for long-term storage or analysis.

WOA is ideal in predictable or controlled environments, where rapid optimization with fixed parameters is feasible. RL, meanwhile, thrives in adaptive contexts, learning over time and improving as the environment evolves.

In smart grid applications, where conditions may fluctuate due to consumption spikes, sensor failures, or network congestion, a hybrid orchestration pipeline could leverage both strategies. WOA could handle baseline optimization, while RL refines decisions under unexpected shifts. This multi-layer orchestration capability is an important contribution of our platform, offering both modularity and flexibility.

By structuring orchestration logic on top of our hybrid semantic model and validating it through two distinct strategies, we demonstrated that intelligent offloading can significantly improve system efficiency, responsiveness, and scalability. Chapter 4 thus marks a pivotal point in the thesis, translating theoretical modeling into actionable, high-performance decision mechanisms for edge-to-cloud computing in the smart grid context.

5. Edge Orchestration Platform and Use-cases

In Chapter 5, we demonstrated the orchestration platform in action by applying it to two real-world inspired use cases, each from a different economic sector. The purpose of this chapter was to validate the proposed architecture, modeling approach, and decision strategies across the full computational continuum, from edge to cloud.

Traditional cloud-based orchestration models are insufficient for managing time-sensitive and location-specific smart grid workloads. High availability platforms are needed to orchestrate computing resources across different layers to maintain low latency, fault tolerance, and reduce the response time. In this sub-chapter, we analyse existing orchestration platforms, architectures, and frameworks that meet such requirements.

The need to bring computation closer to data sources has led to the evolution of edge-native and fog-aware platforms capable of maintaining availability despite connectivity fluctuations

or system failures. Frameworks like Kubernetes [36], when extended for edge computing through platforms such as KubeEdge [37], form the backbone of modern orchestration systems. Kubernetes extensions have been developed to support critical and non-critical services in smart grid systems by offering advanced node scheduling, edge awareness, and policy-driven control [36]. These adaptations enable containerized workloads to be orchestrated across edge-to-cloud hierarchies, maintaining service continuity even under resource or network pressure. Lightweight orchestration solutions, such as FogBus2 [38], provide containerized management in environments with limited compute capacity. Built on top of K3s, a lightweight Kubernetes distribution [38], FogBus2 supports hybrid clusters that integrate edge, fog, and cloud nodes. It includes performance monitoring, blockchain-backed trust, and IoT authentication, making it suitable for privacy-sensitive deployments in healthcare and industrial energy systems. Other platforms [39], address multi-tenant stream processing across fog and cloud layers. It uses RESTful APIs and real-time analytics pipelines to manage high-throughput telemetry and time-series data. The fog layer enables low-latency event pre-processing and filtering, reducing the load on upstream infrastructure and enhancing resilience. The MiCADO-Edge framework [37] extends microservices orchestration to the edge, enabling runtime adaptation and horizontal scaling based on changing workloads. Integrating with KubeEdge [37], it manages the deployment of services across hybrid environments. The Open Connectivity Foundation (OCF) [41] is a modular and infrastructure-agnostic platform designed to align with the OpenFog Reference Architecture. It supports dynamic deployment across heterogeneous devices and decouples optimization logic from the orchestration backend. OCF emphasizes interoperability, policy-based control, and modularity, offering a flexible structure for scalable orchestration.

Lambda Architecture (LA) [43], while not a platform per se, represents an efficient architectural model for integrating real-time and batch processing in energy systems. By separating data pipelines into speed and batch layers, LA allows smart grid systems to simultaneously react to immediate events and learn from historical trends. Tools like Apache Kafka, Storm, and Spark are commonly used to implement LA-based orchestration [43]. On-demand activation of real-time layers, as proposed in [44], further improves resource efficiency by activating stream processing only when anomaly detection or urgent control signals are required. This conditional processing reduces energy and computation overhead while maintaining readiness. Smart grid systems often employ CloudIEPS [32], [33], [34], [35] and other simulation-based digital twin platforms to validate orchestration strategies under realistic conditions. These platforms allow virtual testing of load distribution, fault recovery, and scalability without disrupting live operations, contributing to more adaptive orchestration strategies. Blockchain-integrated orchestration frameworks offer immutable audit trails and decentralized control [31]. These platforms enhance trust and security in energy trading and task delegation, ensuring traceability of decisions.

An important research direction addresses federated orchestration [30], where autonomous clusters independently manage their own resources while coordinating through consensus-based protocols. This reduces the dependency on central controllers and enhances resilience against localized failures. Federated orchestration aligns with the principles of privacy preservation and operational autonomy. Platforms are increasingly incorporating workload scheduling mechanisms that consider real-time energy availability, device battery levels, and renewable energy forecasts [45]. This enables not just computational continuity but also sustainability, aligning orchestration decisions with environmental constraints. Moreover, integration with AI for predictive orchestration can proactively reallocate tasks, anticipate congestion, and optimize resource allocation based on historical data and real-time telemetry

[40], [48]. AI-assisted orchestrators. These features allow smart grid platforms to maintain availability even under highly dynamic conditions.

Another important feature is runtime fault detection and self-healing. Orchestration platforms are being extended with capabilities to detect failed services or nodes and automatically redeploy, or restart affected workloads [46]. Security-driven orchestration models integrate anomaly detection, encrypted service communication, and role-based access control [47]. These features prevent unauthorized actions and mitigate the risk of service hijacking or data breaches, which are particularly important in grid control applications. Projects such as DIGITbrain and ASCLEPIOS [37] demonstrate the maturity of these high-availability platforms in managing Digital Twins and real-time edge workloads. Their architectures showcase modularity, interoperability, and runtime adaptability as enablers of fault-tolerant orchestration in energy applications.

Each use case was selected to highlight specific strengths of the platform and to test its flexibility under different conditions and system behaviors. One focused on energy demand forecasting, where predictive analytics are used to support operational planning. The other focused on adaptive e-fuel scheduling, triggered by environmental signals in urban mobility settings. Together, they provided evidence that the platform can support both batch and stream workflows, semantic reasoning, and dynamic offloading across heterogeneous nodes. Our platform architecture is presented in Figure 5.

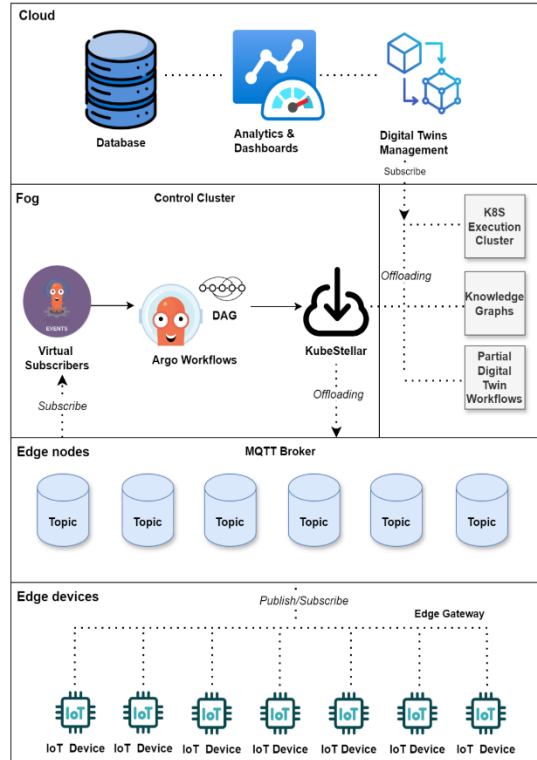


Figure 5. Proposed edge orchestration platform architecture.

The first use case was situated in the energy sector and aimed to forecast electricity consumption at the device level for the following day. In this scenario, smart meters deployed throughout the grid continuously measured half-hourly consumption. These readings were collected by edge devices, which published them using MQTT to the fog layer for further preprocessing, as shown in Figure 6.

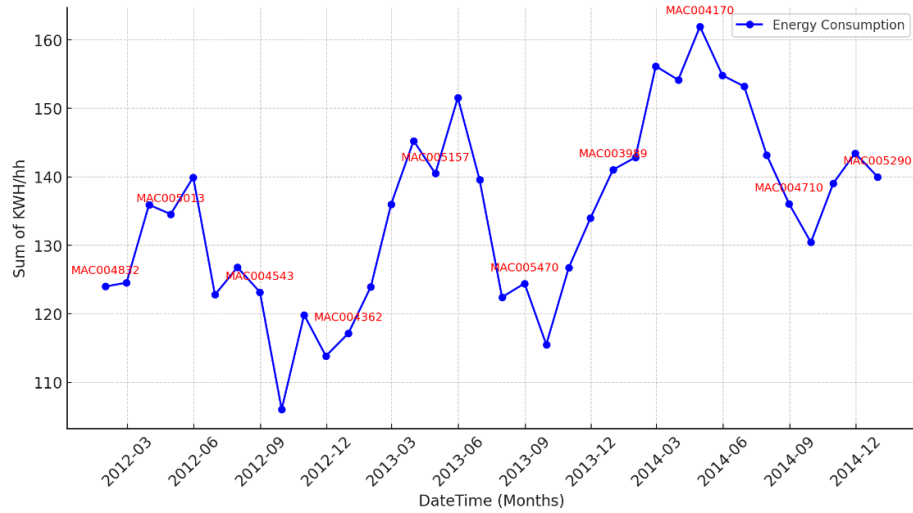


Figure 6. Energy consumption by device stream.

The fog nodes served as an intermediate control point, executing stream workflows that filtered noise, checked for anomalies, and prepared the data for storage. Once data arrived in the cloud layer, it was passed into scheduled batch workflows. These workflows trained and executed predictive models based on recent consumption patterns. Linear regression models were used to forecast the next 24-hour usage profile per device. Results were stored in a PostgreSQL database and made available to stakeholders through a real-time dashboard and REST APIs. The serving layer ensured seamless access to this data through a combination of visualization tools and external integration points.

What made this setup significant was the use of a full Lambda Architecture implementation tailored to the orchestration environment. The edge and fog layers processed real-time telemetry using stream pipelines, while the cloud hosted batch workflows for training and prediction. This division of labor preserved responsiveness while maintaining long-term accuracy. For example, if a device reported an unusual usage pattern, such as an unexpected spike during the night, the platform could flag this through live alerts. Meanwhile, the same data point could also influence tomorrow's prediction if included in the batch pipeline. This layered approach reflected the real-world requirements of energy infrastructure, where both immediate and planned responses are needed.

The results confirm the reliability and performance of the system on various loads. Using a historical dataset from over 5,000 households, we simulated a live data feed by replaying half-hourly meter readings through the edge-to-cloud pipeline. The system was able to sustain 300 concurrent workflows, with each workflow handling roughly 90,000 events across 160 smart meters. Workflow execution times averaged 38 seconds with minimal variance, demonstrating that the orchestration layer could maintain stability and consistency. Memory usage remained efficient, peaking at 70 MB per job even in complex scenarios. CPU utilization was adaptive, scaling between two and four cores depending on batch size and data volatility. Importantly, no workflows were dropped during the tests, and bandwidth use remained within acceptable thresholds due to preprocessing and aggregation at the fog level.

The second use case came from the mobility sector and focused on adaptive e-fuel deployment based on environmental conditions. In this case, air quality sensors installed in multiple Romanian cities monitored PM10 levels in real time. The edge layer ingested sensor data through MQTT and published it for stream processing at the fog layer. This layer handled immediate analysis, flagging pollution spikes or unusual trends. Simultaneously, historical data

were collected in the cloud, where a predictive model generated 24-hour forecasts of PM10 concentration. If a city was predicted to experience a spike beyond a fixed threshold, the system automatically scheduled an adjustment in the fuel blending ratio at the local filling stations. In this case, the proportion of e-fuels would be increased from a mixture of 20% to 30% during peak pollution periods. This proactive scheduling helped reduce CO₂ emissions and aligned fuel deployment with climate goals, as shown in Figure 7.

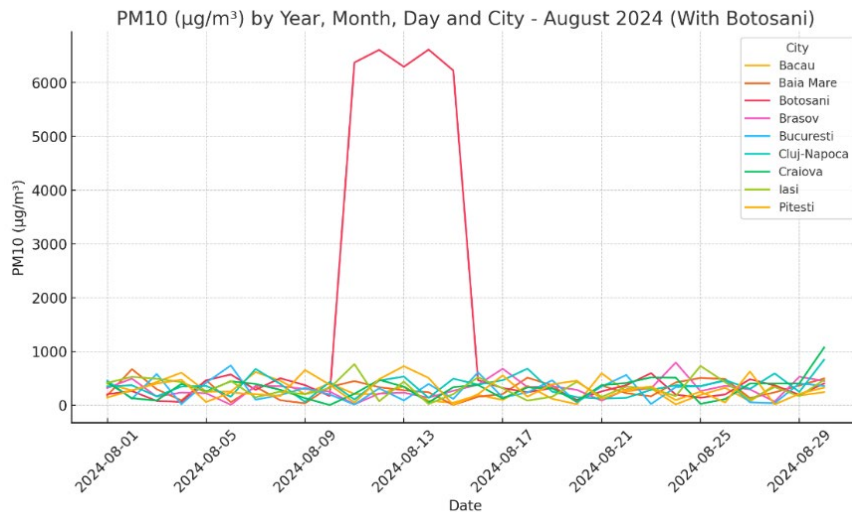


Figure 7. PM10 Levels by City over the past month.

The Lambda Architecture was again deployed to organize data processing across the three layers. At the edge, air quality readings were collected at high frequency. These were pushed through fog-based stream workflows, where initial analysis took place. In the cloud, scheduled batch jobs ran every 24 hours to generate predictions. These outputs were passed to the serving layer, where stakeholders, including city authorities and filling station operators, view upcoming fuel adjustments via dashboards. An API provided programmatic access for automation and integration with municipal systems.

This use case emphasized responsiveness and scalability in the face of unpredictable environmental inputs. The orchestration platform was required to operate across multiple cities, each with different sensor setups, pollution patterns, and infrastructure constraints. Despite this complexity, the platform adapted seamlessly. The batch forecasting process remained under 20 seconds per city, even when processing 6,480 records over 30 days. Ingestion latency stayed below one second per event, and the system scaled to 45 virtual filling stations without performance loss. Offloading strategies assigned most forecast processing to the cloud, while stream workflows were handled closer to the edge for real-time relevance. This balance preserved bandwidth and computational efficiency.

What was particularly valuable about the second use case was how it combined predictive analytics with policy-driven scheduling. Rather than waiting for pollution to rise before reacting, the system used forecasts to adjust behaviors in advance. This demonstrated a broader use of orchestration, not just as a tool for managing workloads, but as a driver of coordinated action based on semantic triggers. The Knowledge Graph stores domain rules such as fuel adjustment thresholds, enabling the system to reason about when to apply updates and how to scale them across stations and cities. This rule-based logic, embedded in a structured semantic layer, allowed flexibility in policy updates without changing the workflow code.

The components, stream ingestion, batch scheduling, semantic modeling, and rule-based decision logic, remained consistent. Only the domain-specific models and thresholds changed. This modularity allowed rapid adaptation and integration into new environments. Whether the application involved energy load forecasting or clean fuel planning, the system remained interpretable, scalable, and transparent to end users.

The use of semantic annotations and rule-based triggers allowed for expressive and adaptive behavior, ensuring that offloading and resource allocation remained aligned with the operational context. Stakeholders could interact with the data through dashboards, APIs, or embedded alert systems, improving both technical performance and human accessibility.

Chapter 5 validated the orchestration system in two distinct sectors, each with different priorities, data types, and operational constraints. The energy use case emphasized time-series forecasting and live monitoring, while the mobility use case showcased environmental responsiveness and policy-driven actuation. The results demonstrated that the platform could orchestrate intelligent behaviors across the edge-fog-cloud continuum, leveraging semantic models, offloading strategies, and multi-layer workflows. These findings support the core claim of the thesis: that orchestration across heterogeneous computing layers can be made adaptable, interpretable, and domain-independent through a unified modeling and execution framework.

6. Final conclusions

We started the thesis with the vision of developing a functional orchestration platform capable of managing tasks across edge, fog, and cloud layers while reacting in near real-time to system conditions. Each research objective posed specific challenges, which were addressed progressively, starting with foundational background and moving toward advanced innovations. Each chapter contributed a key element to the overall platform.

The initial focus was on modeling distributed energy and computational systems in a way that reflects their complexity, interconnections, and real-time behavior. Existing models lack scalability, adaptability, and reasoning capabilities. We introduced a model that uses Digital Twins, Directed Acyclic Graphs, and Knowledge Graphs. This model was designed, implemented, and validated through simulations, with clear criteria for evaluating its ability to represent and manage the computational continuum in smart grids.

This modeling framework supported the orchestration logic by providing visibility into infrastructure costs and adaptability. It accommodated both detailed, resource-aware use cases and high-level task prioritization. Semantic technologies enabled decisions to consider context, such as node relevance or latency requirements, beyond just raw data.

The next challenge was deciding how to offload tasks across the layers. We implemented two strategies: a heuristic method using the Whale Optimization Algorithm for stable environments, and a learning-based method using Deep Q-Learning and A2C for dynamic conditions. These strategies were tested through simulations, each proving effective in different scenarios. This dual approach allowed the platform to adapt to diverse environments, maintaining modularity and flexibility.

A modular orchestration platform was then built, based on Lambda and Kappa architectures, to support both real-time and batch processing. The hybrid model and offloading methods were integrated into this platform. We validated it through two real-world use cases: forecasting day-ahead energy consumption using IoT data and adjusting fuel blends based on air quality. The platform successfully managed distributed decision-making and maintained low-latency performance, even under stress testing.

These tests used real sensor data and operated under realistic constraints. The platform handled continuous changes and sustained active decision-making across all layers. This confirmed the effectiveness of the models, strategies, and architecture in real conditions. Each objective was met with components that were designed, built, integrated, and validated in a complete system.

The work also contributed to innovation beyond the thesis. I co-authored related publications, submitted an invention report on a time synchronization mechanism, and participated in projects such as Bosch’s Digital Fuel Twin and the HEDGE-IoT Horizon Europe project, where the platform was applied in real-world settings.

Future research can expand the system’s intelligence and autonomy. Federated learning could be added to train models locally, improving privacy and adaptability. Large language models could enhance interpretability by translating goals and policies into orchestration logic. Agent-based AI could enable autonomous components to make decisions and coordinate without centralized control.

The platform can also benefit from extensions in rule-based automation and semantic policy control, allowing orchestration strategies to reflect not only performance needs but also evolving compliance and sustainability requirements. The combination of real-time telemetry and reasoning capabilities already positions the system to support predictive and self-adaptive behavior in response to changing grid states.

Further integration with cross-domain data, such as weather patterns, traffic flows, or user behavior, can help the orchestrator make more informed decisions. These datasets can enrich the existing models and allow the orchestration engine to act more proactively, especially in energy forecasting, load balancing, and demand-response scenarios.

Scalability is another critical direction. As device counts and data volumes grow, maintaining orchestration responsiveness and fault tolerance will require distributed control, elastic resource allocation, and asynchronous coordination mechanisms across multiple layers and nodes.

The modeling layer, supported by YAML-based configurations and reusable descriptors, makes the system flexible and easy to update as infrastructure evolves. New nodes, assets, and services can be incorporated with minimal changes to logic, enabling faster deployment and domain portability.

The evaluation scenarios confirmed the ability of the platform to maintain stable operation even when exposed to high-frequency input streams and shifting conditions. By combining event-driven processing at the edge with batch learning and planning in the cloud, the platform balances speed with foresight.

The semantic layer, powered by Knowledge Graphs, can also support future use cases that involve constraint satisfaction, such as prioritizing green energy, enforcing co-location policies, or respecting critical infrastructure zones. These features will help orchestration move from efficiency-driven logic to purpose-driven operations.

The combination of simulation validation and real-world deployment confirms the readiness of the system. Each part of the architecture works both independently and. It adapts to technical and environmental changes; scales as needed and supports both automated execution and human oversight.

The foundation is already there. The next step is to increase autonomy, intelligence, and integration between learning, control, and human interaction in orchestration systems.