systems

MDPI

Article

Engineering IoT-Based Open MAS for Large-Scale V2G/G2V †

Nikolaos I. Spanoudakis [1,*,‡], Charilaos Akasiadis [2,§], Georgios Iatrakis [3,‡] and Georgios Chalkiadakis [3,‡]

[1] School of Production Engineering & Management, Technical University of Crete, 73100 Chania, Greece
[2] Institute of Informatics & Telecommunications, National Centre for Scientific Research 'Demokritos', 15341 Athens, Greece; cakasiadis@iit.demokritos.gr
[3] School of Electrical & Computer Engineering, Technical University of Crete, 73100 Chania, Greece; giatrakis@tuc.gr (G.I.); gchalkiadakis@tuc.gr (G.C.)
* Correspondence: nispanoudakis@tuc.gr; Tel.: +30-28210-37744
† This paper is an extended version of our paper published in PAAMS2022—The 21st International Conferenceon Practical Applications of Agents and Multi-Agent Systems, L'Aquila, Italy, 13–15 July 2022.
‡ Current address: University Campus, Technical University of Crete, 73100 Chania, Greece.
§ Current address: Patriarchou Grigoriou & Neapoleos 27, 15341 Aghia Paraskevi, Greece.

Abstract: In this paper, we aimed to demonstrate how to engineer Internet of Things (IoT)-based open multiagent systems (MASs). Specifically, we put forward an IoT/MAS architectural framework, along with a case study within the important and challenging-to-engineer vehicle-to-grid (V2G) and grid-to-vehicle (G2V) energy transfer problem domain. The proposed solution addresses the important non-functional requirement of scalability. To this end, we employed an open multiagent systems architecture, arranging agents as modular microservices that were interconnected via a multi-protocol Internet of Things platform. Our approach allows agents to view, offer, interconnect, and re-use their various strategies, mechanisms, or other algorithms as modular smart grid services, thus enabling their seamless integration into our MAS architecture, and enabling the solution of the challenging V2G/G2V problem. At the same time, our IoT-based implementation offers both direct applicability in real-world settings and advanced analytics capabilities via enabling digital twin models for smart grid ecosystems. We have described our MAS/IoT-based architecture in detail; validated its applicability via simulation experiments involving large numbers of heterogeneous agents, operating and interacting towards effective V2G/G2V; and studied the performance of various electric vehicle charging scheduling and V2G/G2V-incentivising electricity pricing algorithms. To engineer our solution, we used ASEME, a state-of-the-art methodology for multiagent systems using the Internet of Things. Our solution can be employed for the implementation of real-world prototypes to deliver large-scale V2G/G2V services, as well as for the testing of various schemes in simulation mode.

Keywords: internet of things (IoT); open multiagent systems; smart grid; engineering multiagent systems (EMASs); digital twin

Citation: Spanoudakis, N.I.; Akasiadis, C.; Iatrakis, G.; Chalkiadakis, G. Engineering IoT-Based Open MAS for Large-Scale V2G/G2V. *Systems* **2023**, *11*, 157. https://doi.org/10.3390/systems11030157

Academic Editors: Philippe Mathieu, Juan M. Corchado, Alfonso González-Briones and Fernando De la Prieta Pintado

Received: 4 February 2023
Revised: 10 March 2023
Accepted: 14 March 2023
Published: 19 March 2023

1. Introduction

The smart grid [1] constitutes an important emerging application domain for artificial intelligence and multiagent systems (MAS). In the smart grid, energy and information both flow over electricity distribution and transmission networks in all possible directions. As such, buildings, as well as electric vehicles (EVs), become active energy consumers and/or producers, and the need for their effective integration into the system arises. Not only is the smart grid an electricity network with diverse consumers and producers, it is also a dynamic marketplace where heterogeneous devices appear and need to connect and interoperate [2,3]. To date, several smart grid-related business models and information system architectures have been proposed, but they do not always adhere to particular standards [4]. This is not surprising, given the fact that energy markets can differ in scale, i.e., they can be global, regional, or isolated; that they may be regulated or owned by a

public authorities or the private sector; that they can involve renewable energy sources or non-renewable sources; and finally that they can allow for dynamic pricing based on demand and offers.

As such, energy markets can be naturally viewed as open multiagent systems [5–7]: Participants are agents that can freely enter or exit the system at any time and who are (proactively) setting and pursuing their own (presumably diverse) agendas, goals, and business models. They are able to compete, adapt, or react to their ever-changing, dynamic environment [8]. Moreover, they are socially able: they can negotiate, argue, and partner with others in coalitions [9].

Currently, the Internet of Things (IoT) offers a networking layer that interconnects distributed resources, such as charging controllers, power meters, various sensors and actuators, and processing and decision-support services [3,10]. Given this state of affairs, heterogeneous resources are rendered interoperable by the IoT, since they are henceforth able to exchange information and also reconfigure the parameters that are crucial for their operation [11]. Thus, the actual deployment of such approaches is now possible.

An IoT-enabled smart grid digital twin can represent the running states of the multitude of underlying interconnected physical devices (e.g., smart meters, controllers, energy storage devices), and has the ability to continuously collect the respective sensor measurements. The actual per-device grid state can thus be made available to the operators in real-time [12,13]. This monitoring capability can be further expanded with predictive maintenance techniques, allowing for the detection of malfunctions even before they occur [14]. At the same time, having access to historical per-device measurements allows the post-hoc analysis and/or training of machine-learning models [13]. To this end, agents can enable the digital twins of (cyber-) physical objects by being able to represent all (physical) assets of the smart grid domain. Agents can also represent classes of producers/consumers or even prosumers (i.e., entities that can be both consumers or producers). Electric vehicles, in particular, act as consumers while charging, but can also be producers if they provide energy from their batteries back to the Grid.

However, existing smart grid approaches do not provide functional open prototypes offering features such as the above, nor do they adequately exploit existing engineering MAS research paradigms. This is especially true for our particular domain of focus in this paper, the vehicle-to-grid (V2G)/grid-to-vehicle (G2V) problem. Regardless of this, there is a real need for diverse agents representing stakeholders in an open system to be equipped with predefined protocols which they can use in order to interact [7]. Importantly, stakeholders also need to be able to enrich such protocols with their own goals and/or algorithms.

The objective of our work in this paper was to fulfil such requirements in the V2G/G2V domain, contributing a novel IoT-based open MAS architecture designed to meet the aforementioned objectives. The innovative aspects of our work are, on the one hand, the fact that by employing our architecture, according to their particular goals, the various stakeholders are able to develop new agents, or to re-use existing ones, as they see fit, to cover their needs. Moreover, on the other hand, by employing an IoT platform that supports multiple application-layer protocols, we ensure that new, diverse agents can connect to the system to offer their services and to exchange energy, given pricing mechanisms that are possibly dynamic—i.e., designed to adapt and fluctuate so as to promote system stability and reliability in a game-theoretic manner [15].

In particular, our system employs SYNAISTHISI, a research-oriented IoT platform deployed in docker containers, which allows agents to connect and communicate using the Message Queuing Telemetry Transport (MQTT) publish/subscribe protocol [16], among others. We demonstrate the validity of the approach via simulation experiments involving three different charging scheduling algorithms and two dynamic pricing mechanisms proposed in the recent V2G/G2V research literature.

In this paper we extend upon a recent study presented at the 20th International Conference on Practical Applications of Agents and Multi-Agent Systems (PAAMS 2022) [17] in

three ways. First, we emphasize the engineering aspect of the development of this system. We describe the application of the agent systems engineering methodology (ASEME) [18] and provide insights on the agents' architecture, as well as the interaction protocols that they use. The ASEME methodology was selected as it has been used in the past for IoT-based MAS development [19] and is also referred to in surveys of this application domain [20–22]. Second, we show explicitly how agents enable the digital twins of vehicles and other stakeholders in the smart grid domain. Third, we conduct and document more experiments, and compare the performances of two different dynamic pricing mechanisms.

The rest of this paper is structured as follows. Section 2 presents the necessary background and discusses related work. Section 3 then puts forward our V2G/G2V-specific MAS-based architecture and offers a detailed description of the roles of the various agents, and a presentation of their interactions. Following that, in Section 4 we discuss our system's development process in detail. In particular, we present the IoT communications infrastructure; the ASEME-based statechart description of the inter-agent protocols and intra-agent control models dictating the agents' interactions and behaviour respectively; and the various V2G/G2V agent strategies currently incorporated in our framework. In Section 5 we conduct a thorough experimental evaluation of our architectural framework, verifying its applicability within realistic use-case scenarios of interest. In Section 6 we discuss the benefits arising for various stakeholders from its potential adoption and real-world integration, also focusing on the digital twin for the smart grid. Finally, Section 7 concludes this paper, and outlines directions for future work.

2. Background

In this section, we provide the necessary background for our work in this study. This includes the concept of a smart grid and the V2G/G2V problem in particular, for which we provide an overview of previous works and the motivations for our own. We then discuss the simulators, the IoT technology, and the SYNAISTHISI platform that we employed. Subsequently, we focus on the method used for engineering IoT-enabled MASs.

2.1. Smart Grids and the V2G/G2V Problem

As EVs further penetrate energy markets globally, electricity demand patterns are subject to change at levels that might become disruptive to the stability and the reliability of the current electricity grids [23]. A way to mitigate this risk is by introducing "smart charging", or grid-to-vehicle (G2V) capabilities, according to which the charging of EVs can be delayed and can take place at later time intervals than immediately after connecting to a charger [24], seeking, e.g., those with more renewable production, with less demand from other EVs, or with better pricing. In the opposite direction to that of G2V, vehicle-to-Grid (V2G) approaches can benefit from the capability of EV batteries to store energy, and thus coordinate their discharging to support situations of energy supply shortage [25].

Since the smart grid consists of multiple individual and economically minded entities, it is natural to model it as a MAS [26]. MASs provide a number of benefits in contrast to their centralized counterparts, such as faster computation times and scalability, since processing is performed in a decentralized fashion, and private data are not required to be shared.To date, many simulation tools and prototypes have been proposed that put forward V2G capabilities. Such approaches may either analyze low-level technical details regarding the operation of EVs, or are integrated into environments that include the respective individual stakeholders. We now proceed to briefly review several representative such systems.

The study presented in [27] introduced EVLibSim, a Java-based simulator of the operation of charging stations. This tool offers a user interface (UI) that can be used to manage charging stations. Its capabilities include the creation, modification, and monitoring of charging stations given the application of particular scheduling algorithms. In addition to being used by domain experts to test potential scenarios of use, it focuses on charging stations only, without incorporating different types of stakeholders.

The work presented in [28] involved describing a MAS that supports the decision-making of EV drivers in regard to locating charging stations and charging opportunities in the city of Valencia. This system incorporates multimodal information from various sources, such as traffic monitoring systems, social networks, and pricing, in order to optimize the placement of charging stations. Such approaches are valuable during the design of charging infrastructure, but do not help in deciding what will happen next, after the infrastructure is deployed and becomes operational.

The survey presented in [8] involved a large-scale literature analysis of the MAS-based control of smart grids, providing information regarding the related technologies and standards, and the application of intelligent agents commercial projects. The authors in [29] proposed coalition formation techniques for EVs, providing services related to V2G and demand-side management. An MAS architecture was designed, and the implementation of simulations was performed, using the Java Agent Development Framework (JADE [30] [1]). In that study, different kinds of intelligent agents were considered, i.e., EVs, aggregators that form EV coalitions, and a transmission system operator that acts as a mediator and regulator. Coalitions were formed with the objective of reaching minimum energy requirements for participating in the regulation market. However, this approach did not allow for more sophisticated selection processes, making it difficult to scale, and the presented evaluation involved only five EVs.

Another approach based on the use of JADE for the coordination of EV battery charging is that of [31]. In that study, individual EV driver preferences were taken into account, such as their willingness for V2G participation and the vehicle's charging availability. It was shown via experiments that the proposed MAS managed to satisfy EV owners' charging preferences individually, even in emergency conditions.

2.2. Frameworks and IoT-Based Real-World Trials

In recent years, great progress has been achieved with respect to the delivery of real-world trials that offer V2G/G2V and which might incorporate simulators as well. To begin, XBOS-V [32] is a software-based open platform that can be utilized for controlling the charging of EVs connected to small buildings. The implementation of the standardized communication method for V2G, ISO 15118[2], provides the connection specifications for chargers and EVs. Relevant approaches are the open charge point interface (OCPI), the open charge point protocol (OCPP), and the open smart charging protocol (OSCP) [33]. OpenV2G [34] provides the required modules to implement the V2G public key infrastructure, e.g., to guarantee security for the EV and charging station connections, and also to allow for simulations to be executed. Another approach, the grid-integrated electric mobility model (GEM) [35], simulates both electricity and mobility aspects. However, this approach only allows for analysis to be conducted at a higher level, without referring to charging station recommenders, for example, and other particular stakeholders. Another tool that can be used to manage the charging of batteries is ACN-Sim [36], which can be utilized by individual end-users but not by large-scale grid operators.

Regarding the IoT domain, SYNAISTHISI is a research-oriented platform composed of open-source frameworks that can host dockerized services and can also act as a translator between various application-layer protocols [16]. Service dockerization allows scalable deployments to occur independently of the underlying operating systems of the hosts. Furthermore, the platform's support for multiple protocols enables the orchestration of heterogeneous agents and services that may follow different implementation paths. Furthermore, the platform offers user authentication and authorization and restricts access to private and sensitive channels for the exchange of information, and also supports semantic annotations of exchanged information and available services.

2.3. Engineering MASs

The fields of agent-oriented software engineering and multiagent systems engineering have produced a wealth of abstractions, methods, and techniques for developing MASs. A

survey of this field is outside of the scope of this paper; however, the motivation for our choice of method was based on surveys in the field and in the selected platform (IoT).

In the development of our system, we followed the approach laid out by ASEME, the Agent Systems Engineering MEthodology [18]. ASEME can be naturally used in the design and modeling of IoT-based MAS systems, as well as in ambient intelligence applications [19–22]. It builds on statecharts and, more broadly, the unified modeling language (UML [37]) in order to perform system analysis and design models. It is agent-architecture- and agent-mental-model-independent, allowing the designer to select the architecture type and the mental attributes of the agent, thus supporting heterogeneous agent architectures. Moreover, ASEME puts forward a modular agent design approach and uses the so-called intra-agent and inter-agent control concepts. The former is implemented to coordinate the different modules that implement the agent's capabilities, thus determining its behavior, whereas the latter allows for the control of the society of agents by defining the protocols that govern its coordination.

Importantly, in agent communication, there typically exist predefined message sequences that can be applied in several situations that share the same communication pattern regardless of the application domain [30]. Such message sequences are defined as protocols.

ASEME uses two relatively common abstractions for modeling agents: capability and functionality. Busetta et al. [38] view capability as "a cluster of plans, beliefs, events and scoping rules over them". Braubach et al. [39] extended this idea and proposed that capabilities can contain sub-capabilities and have at most one parent capability. They defined the agent concept as an extension of the capability concept, aggregating capabilities. In the Prometheus methodology [40], each functionality identified in the analysis phase ends up being mapped to a capability in the design phase. In the agent modeling language (AML) [41], capability is a concept used to model an abstraction of a behavior in terms of its inputs, outputs, pre-conditions, and post-conditions. The behavior is the software component, and its capabilities are the signatures of the methods that the behavior realizes, accompanied by the method's pre-conditions and post-conditions. This approach is similar to that of service-oriented architectures, and thus considers the agent as an aggregation of services.

In ASEME, the agent coordinates its capabilities in the intra-agent control model. Capabilities are themselves decomposed to simple activities. For instance, one capability of a personal assistant agent is the ability to locate an appropriate charging station for its user's car. This task can be decomposed to specific activities, e.g., one activity is finding out which charging stations are in the user's vicinity, which charging sockets are available at each of them, and their free slots. Another activity is ranking the available slots according to the user's preferences. After identifying the activities associated with a capability, the next step is to connect them to a specific functionality , i.e., a specific method, algorithm, technology, or technique. This is an important managerial task as each activity can be connected to different functionalities. For example, the decision-making activity of ranking the available charging slots may be connected to an argumentation theory, implying an argumentation-based method, or to a utility function, implying a multi-criteria decision analysis method.

The inter-agent control model defines the capability of an agent to participate as a role in a specific protocol. ASEME allows the seamless integration of the inter-agent control model in the intra-agent control model as they follow the same formalism—i.e., statecharts [42]. The statecharts formalism does not exhibit the limitations (limited scalability, explosion of states) posed by other formalisms such as Petri-nets [43]. Therefore, in this study, we used the statechart formalism to define our open protocols and design patterns. Finally, ASEME automatically generates portions of the agent code or provides guidelines for the programmers to transform their design models into implementation models.

3. System Architecture

In this section, we provide an overview of the architecture of our system. First, we provide an overview of the application domain and identify the stakeholders. Then, we perform an analysis of the problem domain and identify the agents. Finally, we focus on defining their interactions. We use the ASEME System Actors Goals (SAG) model to depict the agents and their goals.

3.1. Overview of the Application Domain

We are interested in the vehicle-to-grid (V2G) and grid-to-vehicle (G2V) energy transfer problem domain. The first activity proposed in the ASEME methodology is to identify the stakeholders and their goals. After studying the related literature, we focused on the following smart grid stakeholder types [44–46]:

- Electric vehicle (EV) owners, who are usually also the drivers of EVs. To effectively use their EVs, they need to book a place for charging them at appropriate stations, and they pay for such a service. They might even be interested in charging them at a lower price if the charging station could discharge their EV batteries to contribute to the network when prices are high, i.e., acting as prosumers. Here, we consider owners of both battery electric vehicles (BEV) and plug-in hybrid electric vehicles (PHEV).
- Charging station owners buy energy from producers to charge electric vehicles. In some cases they can utilize (partially) charged EVs by employing them as energy producers when network prices are high and then recharge the EVs later at a better rate.
- Electricity producers are typical (possibly renewable) energy producers. They sell power to the network at rates that are usually based on supply and demand. To compute the latter, they depend on electricity imbalance indicators, which are usually monitored by the global network operator.
- Electricity consumers are typical households, industries, and other buildings and their corresponding infrastructure.
- Station recommender service providers represent groups of stations and act as mediators between EVs and charging stations. EV owners depend on them to find stations that suit their schedule and preferences, and stations use them to reach out to customers. The represented groups of stations may belong at the same firm or may operate in the same region.
- Electricity imbalance providers can be network operators or government agencies that monitor the grid balance and calculate/predict the periods of electricity shortage and surplus.
- Mechanism designers are intermediate trusted third parties responsible for calculating dynamic prices and managing the various payments between the stakeholders listed above.

3.2. The Agent-Based Approach

Since these diverse stakeholders have their own goals and business needs, they can all be represented by software *agents* in a system aiming to automate their function, and this system can be a digital twin, allowing for their simulation and study. Agents are a suitable paradigm, as they have the following capabilities [47]:

- They are autonomous , meaning they can operate without the direct control of humans, and with at least some control over their own actions, their internal state, and resource consumption;
- social, They are able to interact with other agents—including humans—and can choose their collaborators;
- They are reactive, perceiving and responding in a timely fashion to changes in the environment, according to their goals; and
- They are proactive, exhibiting goal-directed behavior by taking the initiative, being purposeful, and not simply acting in response to changes in the environment.

Thus, autonomous agents can represent stakeholders without direct intervention, with agents being able to locate their best collaborators and take initiative in constant pursuit of their goals. This can be achieved by adopting different software implementations that put forward heterogeneous strategies, for example, some charging stations may strive to charge EVs as soon as they are connected, whereas others might behave in a different manner and choose to manage demand congestion [48]. Note, however, that modeling the low-level power grid details, such as feeders or distribution-line constraints, lies beyond the scope of our work.

Special attention needs to be paid to the station recommender stakeholder, whose main goal is to bring together charging stations and EV owners. Such tasks are typically undertaken by "middle" agents. The "middle" agent can have different roles in a multiagent system (MAS) [49], such as that of a matchmaker (who brings service providers in contact with service requesters, who then communicate to make the transaction), a broker or facilitator (who facilitates the transaction), or a mediator (a combination of the previous two, who brokers the transaction but also brings the buyer in contact with the seller). In our case, we use the latter approach, since the charging station agent may need to negotiate the charging details directly with the EV agent.

Taking these into account, we present the specific cooperation protocols and the high-level architecture that can be used to deliver a MAS V2G/G2V framework. Our approach can be used to investigate and evaluate different implementations and strategies that agents may incorporate in a real-world setting. The protocols we put forward are open, as they can be easily extended and tailored to capture a plethora of real-world cases. In contrast to previous studies, detailed descriptions and semantic schemes for each service are provided, which enable the functionalities that agents would request in such settings.

By incorporating our framework, algorithms of the designers' choosing can be tested and compared, e.g., methods that generate recommendations for charging and that calculate charging schedules for large numbers of EVs, or alternative pricing schemes that might induce different effects in real-world use cases. The applicability of our approach is illustrated below by implementing a functional prototype that adopts the proposed architecture, and by using it to execute simulations of use-cases to demonstrate its overall functionality. An important feature of our implementation is that agents come as modular components, and thus can be easily augmented or even replaced with approaches that nevertheless follow the protocols that we have defined.

Our architecture assumes that agents exist within a microgrid infrastructure that can be linked to other segments of the smart grid via distribution and transmission networks. A microgrid can import power when its local generation falls short, and can export surplus energy to the grid for added producer profits, according to any energy market regulations in effect [2]. Figure 1 provides an overview of the agents and their interactions.

Specifically, the types of agents considered in the system include multiple (*a*) electric vehicle agents (EV), (*b*) charging station agents (CS), (*c*) electricity producing agents (EP), and (*d*) electricity-consuming agents (EC). We also assume the existence of a regulatory service, which may be for-profit private service, consisting of the following agents: a service aggregator or (*i*) a station recommender (SR), (*ii*) an electricity imbalance (EI) for load monitoring, and (*iii*) a mechanism design (MD) for the generation of electricity prices.

Figure 2 depicts these agents as actors with goals in the form of the System Actors Goals (SAG) model of ASEME. The SAG model for the requirements analysis phase includes a graph with actors and their goals. The goal of one actor (the owner of the goal) may depend on another actor (collaborator) to be realized. In the figure, we can see the actors represented as cyan circles and the goals as yellow rounded rectangles. The dependencies are shown as arrows directed from the owner to the goal and from the goal to the collaborator(s). In the analysis phase, the particular goals are analyzed and represented by capabilities.

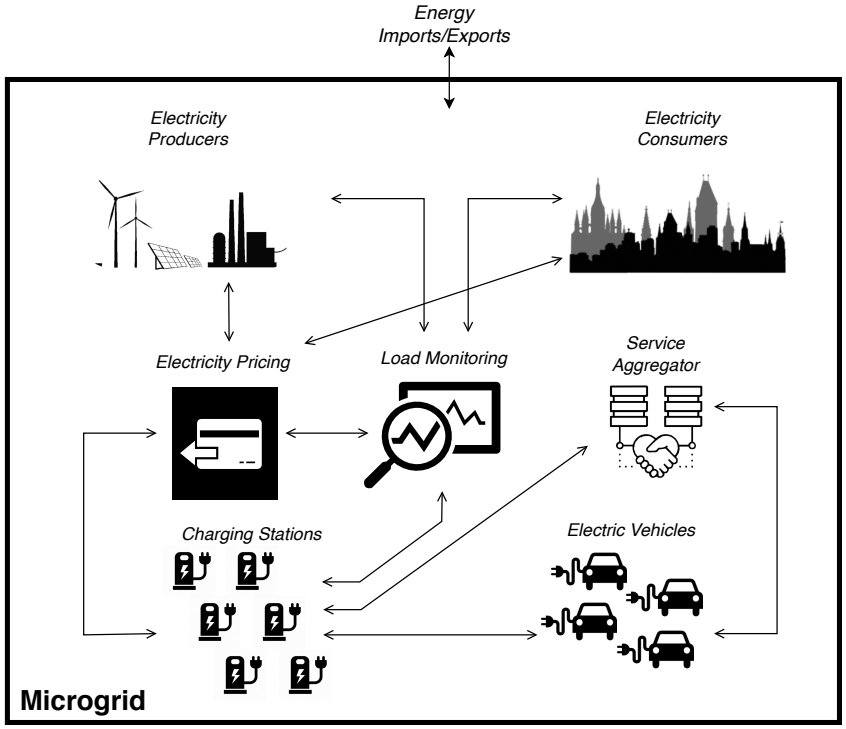

Figure 1. High-level overview of the V2G/G2V stakeholders and their interactions.

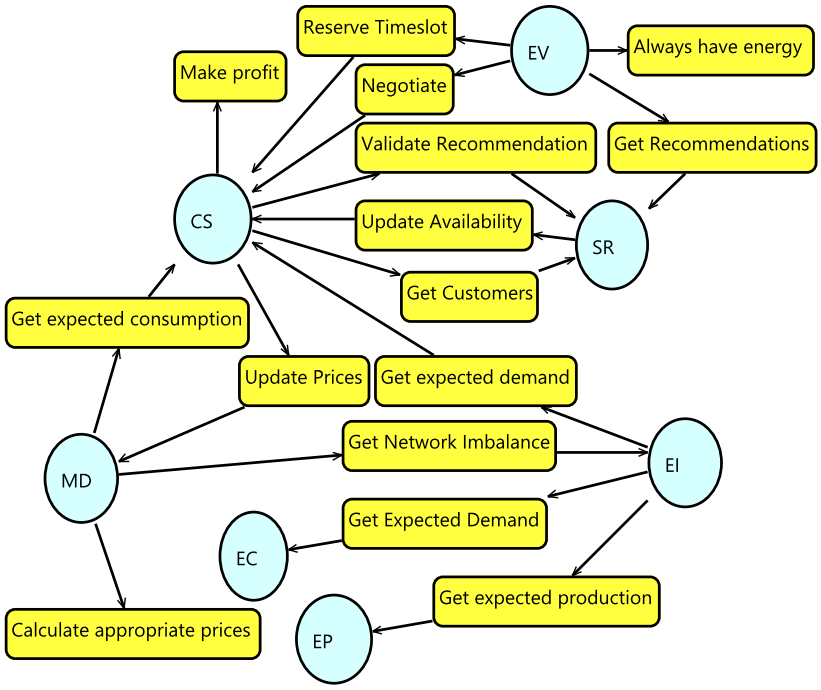

Figure 2. The System Actors Goals model of ASEME.

EV agents typically optimize utility functions that are set by the owner of each vehicle. Examples of utility functions include ones that guarantee that the EV will constantly have enough charge in the battery to perform its next trip, or to achieve this with the minimum cost possible, etc. The *EV agent* can monitor driving behavior and extract the underlying models to predict probable future activities and corresponding needs.

EVs may also communicate with a charging station to schedule a charging session or seek profit by participating in V2G and engage in negotiations with charging stations.

Such an agent might also consist of submodules, such as components for driver preference elicitation that monitor the typical habits and behavior of the driver, and possibly even forecast future preferences, or interoperable user interfaces, attainable either via a mobile device or the dashboard of the vehicle, which can be utilized by humans to operate respective procedures and monitor their conduct, for example, payments and negotiations, or simply to browse and select recommendations.

Furthermore, EV agents can adopt alternative tactics to automatically decide upon a charging schedule based on the defined preferences of each driver, such as cost reductions, faster access, preferred charging networks, location-based choices, and so on. An *EV agent* typically communicates with a *SR agent* with the aim of acquiring recommendations and with *CS agents* to reserve charging slots.

Subsequently, the *CS Agents* regulate the physical access points (e.g., connectors, parking spaces) through which EVs can connect to the grid, and may also earn revenue by charging their batteries. They communicate with EV agents regarding existing charging agreements and modify certain parameters to accommodate the charging of extra vehicles, in order to improve the utilization of the station infrastructure and generate increased profits. A *CS Agent* may encompass a charging scheduling module responsible for scheduling charging/discharging activities over a predetermined timeframe, a negotiation decision-making module for negotiating, a pricing module that calculates costs and payments, and a preference elicitation module that monitors the usage of charging slots and adjusts prices based on the station owner's needs. A *CS Agent* communicates with *SR*, *MD*, *EI*, and *EV* agents.

The *SR agent* notifies EVs with recommendations regarding a subset of the available CS and charging slots that match the most with their preferences, according to, e.g., the charging duration and the driving distance. This agent can be also augmented to take into account various grid constraints, for example, to help avoid herding effects. It consists of a recommendations engine module that generates charging station recommendations, an EV repository module that stores information about the past EV behavior in order to utilize it for future recommendations, and a charging station repository of all the CSs that have registered with the service. It communicates with the *CS* and *EV* agents.

The *EI agent* aggregates data from the *EP*, *CS*, and *EC* agents regarding their future anticipated energy consumption/production profiles, and calculates the periods during which electricity is in a state of shortage or surplus. In turn, it communicates the levels of energy imbalance with every interested party, in order for them to plan and optimize consumption and production. It includes a constraint extraction module that may incorporate different measures and methods relevant in such a scenario, e.g., monitoring any technical or practical limitations of V2G applicability, power flows, etc. It also calculates electricity imbalances over a planning horizon. Additional repositories may contain the stations, producers, and consumers that participate in the scheme.

The *MD agent* corresponds to an intermediate trusted third-party entity, which undertakes to calculate dynamic prices and to manage the payments of the various contributor types. Its goal is to assign appropriate and perhaps even personalized rates for energy consumption and production by *CS*, *EC*, and *EP* agents. It can put forward pricing mechanisms in order to incentivize agents to be truthful regarding their statements of expected values, as well as their actual behavior.

Finally, the *EP* and *EC* agents forecast and periodically report their expected production and consumption levels, respectively, accompanied by confidence values for these forecasts. *EP* and *EC* agents also exchange information with the *EI* and *MD* agents. User interfaces can be considered important submodules, since they are required by every agent type in order to provide monitoring capabilities to the operators if fully autonomous operation is enabled, or to allow human intervention in other operation modes (e.g., semi-automatic or manual).

3.3. Agent Interactions

Figure 3 illustrates the agent types and the protocols that they use to enable cooperation. Note that the high-level goals shown in Figure 2 have been elaborated to reflect specific interactions, which are labeled with an identifier so that we can easily refer to them. Briefly, the *cooperation protocols* dictating agent interactions are as follows:

CP1 Charging Recommendation: Initiated by an EV for the scheduling of a charging session. The EV submits its preference and current location to the SR and receives a list of recommended CSs, along with the available time slots.

CP2 Charging Station Reservation: Following CP1, the EV uses CP2 to reserve the selected charging slot at the respective CS.

CP3 Negotiation: An optional protocol, which may be initiated after CP2, whenever either the CS or the EV, for whatever reason, needs to reschedule a charging session that has been reserved.

CP4 Charging Station Registration: This interaction is used to register new CSs into the system. According to it, the CS informs the MD, EI, and SR agents about the required specifications.

CP5 Authenticate Recommendation: After CP2, the CS asks the SR for validation that in fact the SR was the one that proposed the particular matchmaking between the EV and the CS.

CP6 Electricity Prices: This follows CP7 and involves the MD calculating updated prices and submit the new values to every CS.

CP7 Electricity Imbalance: This immediately follows CP10 or CP8. In the case that the expected production or consumption levels change, the EI must broadcast the updated values to the MD and every CS.

CP8 Charging Station Update Schedule: After CP5, the CS makes a reservation of the requested time slot and notifies the EI and the MD accordingly.

CP9 Producer Consumer Registration: Registers new producers and consumers. New stakeholders must inform EI and MD about their types.

CP10 Update Expected Production/Consumption: This is initiated periodically (e.g., at the beginning of each day). In this step, every producer and consumer agent informs the EI and MD agents regarding the coming day's expected production and consumption levels.

CP11 Update Energy Profile Confidence: This is initiated periodically (e.g., at the beginning of each day). In this step, every producer and consumer agent informs the EI and MD agents regarding the confidence that accompanies the forecasts described in CP10.

CP12 Update Station Availability: Following CP2, this interaction is used by the CSs to update their information for the SR regarding available charging slots after new reservations.

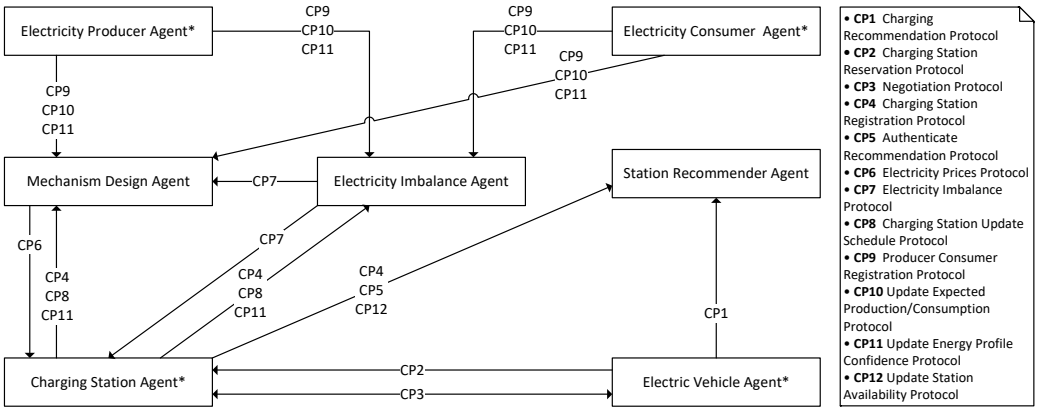

Figure 3. The proposed architecture. (*) denotes agent types with multiple instances. Arrows start from the agent that initiates the interaction and point to the receiver agents.

4. System Development

In the following, we first discuss the communications and deployment infrastructure. This enables the reader to understand the terminology behind the agent communication protocol definitions that are subsequently presented, using the language of statecharts. Then, we discuss the development of the agent models. We give an example in each case. Finally, we briefly describe the implemented strategies for the mechanism design agent and for the charging schedulers.

4.1. Communication Using the IoT Platform

Here, we describe the IoT platform that is used for agent communications and the incorporated cooperation protocols. Our implementation is based on the SYNAISTHISI platform [16], but any other solution that would offer the desirable features that we analyze below could be incorporated as well. We chose this particular platform for a number of reasons. First, it is offered with a non-commercial license and is mainly oriented towards research. By design, end-users are allowed to onboard new software services of their own and can be combined into more complex application designs. Furthermore, from a technical perspective, the platform supports widely used application-layer protocols (MQTT[3], HTTP/REST[4], etc.), along with translations of messages from one protocol to another. Importantly, the platform is deployed in docker containers, making it portable, interoperable with other software, as well as being scalable for large-scale deployments. Finally, to secure real-world deployments, user authentication and authorization processes are integrated in order to prevent unauthorized access to private information, such as locations, schedules, or other personal data that might be required to be shared for the purposes of V2G/G2V operations.

According to the modular service design paradigm, the interconnection among services is performed via an exchange of messages, for which we utilize the MQTT publish/-subscribe application-layer protocol in our design. A service can subscribe to topics to receive messages or publish to topics that other services have subscribed to in order to send information and commands. To access or transmit data through specific topics, however, the service owner must possess the necessary privileges, which can be managed through the platform's graphical user interface (GUI). The same applies to the deployment and the execution monitoring of deployed services.

Mobile assets such as EVs that need to exchange messages are required to have wireless internet connections. For immobile objects, such as charging stations, supervisory control and data acquisition (SCADA) systems, etc., appropriate connectors can be interfaced with the platform, using either wireless or wired internet connections. The platform's *broker* is responsible for notifying subscribers when a message is published.

4.2. Agent Interaction Protocols

The MAS cooperation protocols are defined here using the ASEME inter-agent control (EAC) model. To illustrate the process, we present the charging recommendation protocol (denoted as CP1 in Figure 3) defining the relevant interaction between an electric vehicle agent (EV) and a station recommender agent (SR) in Figure 4. The protocol is defined as a statechart (following the semantics of Harel [42] and the graphical model syntax of the ASEME statechart editor [19,50]) with the AND-state *CP1_ChargingRecommendation* as the root.

AND-states (depicted with a light blue color in the figure) contain OR states, and being in an AND-state entails being in all its OR-states simultaneously, implying that the latter are executed in parallel. OR-states (with yellow-colored labels) contain other sub-states, only one of which can be entered at any time. Basic states (shown in green) are where agent activities are executed. START-states show the beginning of the execution (black dots) and END-states show where execution ends (black dots within a circle). Transitions among states occur (*i*) when the activity of the source state is finalized and there is no event on the arrow, or (*ii*) when the event on the arrow takes place.

Returning to Figure 4, the two OR-states in the AND-state represent the participating agents—i.e., EV and SR. According to this protocol, the EV first enters the *SendRecommendationRequest* state and the SR enters the *ReceiveRecommendationRequest* state. The EV prepares its request by filling in the preferences and location data structures and publishes it to the broker. Via the latter action, the *publish* ("*EV/+/RequestChargingRecommendations*", [*preferences, location*]) event takes place and the EV transitions to the *ReceiveRecommendations* state to wait for the SR's results. The '+' sign is replaced by the agent's ID, as each agent has its own topic for publishing requests.

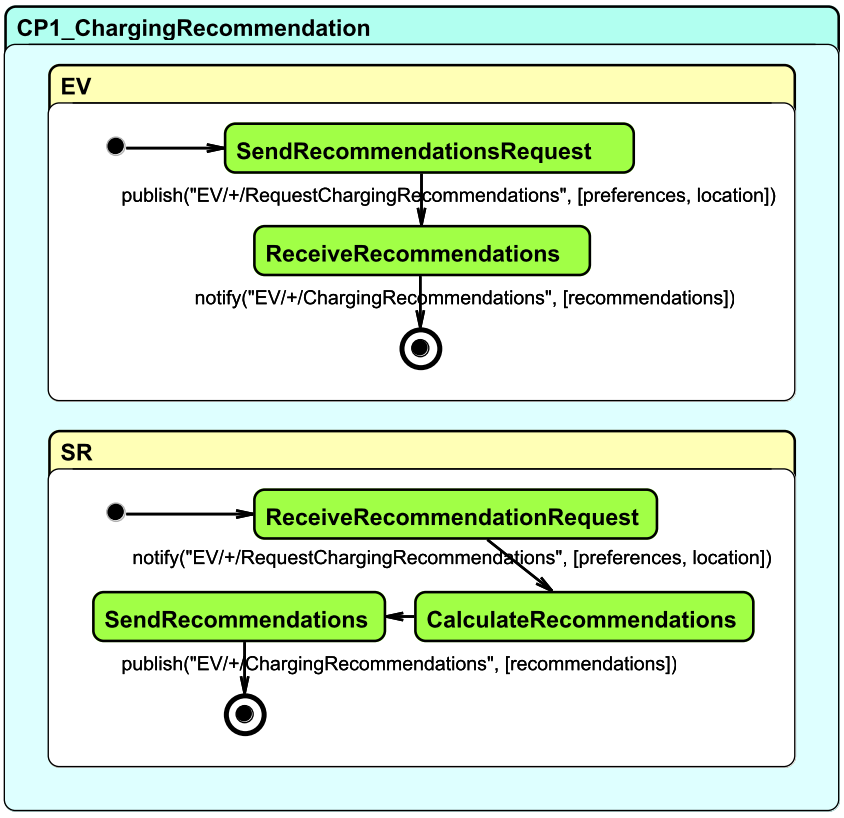

Figure 4. The charging recommendation protocol (CP1).

The broker receives the event and generates the relevant notification for the SR. The SR receives the event and enters the *CalculateRecommendations* state to find the best options for the EV. As soon as it completes this process, it enters the *SendRecommendations* state,

where it sends its reply by publishing a message to the appropriate topic. The EV is notified and the protocol is finished for both agents.

All the protocols shown in Figure 3 are defined using the ASEME EAC model. This allows the development of the protocols independently of the agents' development. The protocols are reusable modules and the agent developers can use them in order to ensure the agent's flawless participation in the open MAS.

4.3. Agent Model

The agent's are modeled using the intra-agent control (IAC) model in ASEME, which is represented by a statechart as well. Thus, it is very easy for an agent to integrate into its model the capability of participating in an interaction protocol. The orthogonal component of the assumed role in the EAC model is inserted as in the IAC model.

To illustrate this process, we depict the IAC model for the EV agent in Figure 5 (we do not provide the transition expressions so as not to visually clutter the diagram). Note that to simplify representation, we show the protocol roles that the agent realizes as basic states. These can be expanded to the relevant OR-states in the respective protocols. For example, the *CP1_ChargingRecommendation:EV* BASIC-state must be replaced by the *EV* OR-state of Figure 4[5].

At the beginning of its operation, the agent enters the *Init* state and is initialized. Then, the *Negotiate* and *Reserve* orthogonal components follow (Figure 5). Arriving at the *Reserve* component, it then transitions to the *DecideNextAction* basic state. There, the agent decides the charging preferences and the desired location.

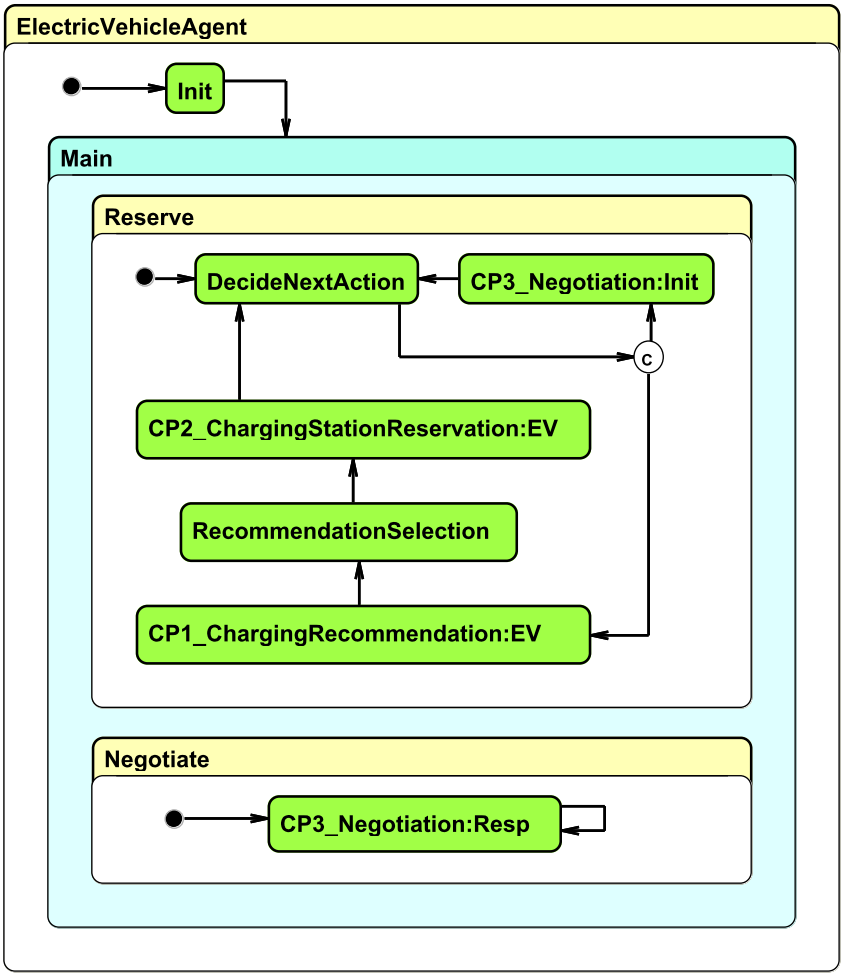

Figure 5. The intra-agent model of the electric vehicle agent.

Then, the *EV agent* has three different choices to make: (*a*) it monitors and predicts the battery state and driver preferences so as to autonomously decide if and how the charging will be arranged; (*b*) it receives required information from predefined datasets[6]; and (*c*) it may prompt the user via a GUI for charging preferences and manual protocol initiation. These three possibilities correspond to different implementations of the *DecideNextAction* state activity.

Whenever the agent decides to arrange a forthcoming charging process, it enters the *CP1_ ChargingRecommendation:EV* state, then the *RecommendationSelection* state (to select the best offer), and finally the *CP2_ChargingStationReservation:EV* to reserve the selected slot. Then, it returns to the *DecideNextAction* state, from which it will have to transit in order to make a new reservation or to negotiate a change in an existing arrangement using the *CP3_Negotiation:Init* state.

As the negotiation protocol (CP3) can be initiated by both parties (EV or CS), the roles it defines are that of the initiator (*Init*) and that of the responder (*Resp*). As the reader can see, the EV can act either as an initiator (entering the *CP3_Negotiation:Init* state) or as responder (entering the *CP3_Negotiation:Resp* state). The latter merely sends a proposal, and if the *CS* agent replies, the rest of the negotiation process will be taken care of by the *NegotiationProtocol:resp* state (using the *responder* role of the respective protocol) at the *Negotiate* component. This is always executed in parallel to the *Reserve* component, as a *CS* agent could itself initiate a negotiation at any time.

4.4. Scenario Demonstration

Herein we present one system use-case scenario. In this scenario, an *EV* makes a reservation at a *CS* after receiving recommendations from the *SR*. In more detail, the agent interactions required for an EV to reserve a charging slot are depicted in the UML sequence diagram shown in Figure 6, which has been slightly simplified for ease of exposition. Note that these interactions require the execution of several protocols (in particular, CP1, CP2, CP5, CP6, CP7, CP8, and CP12), which are already provided in our implementation.

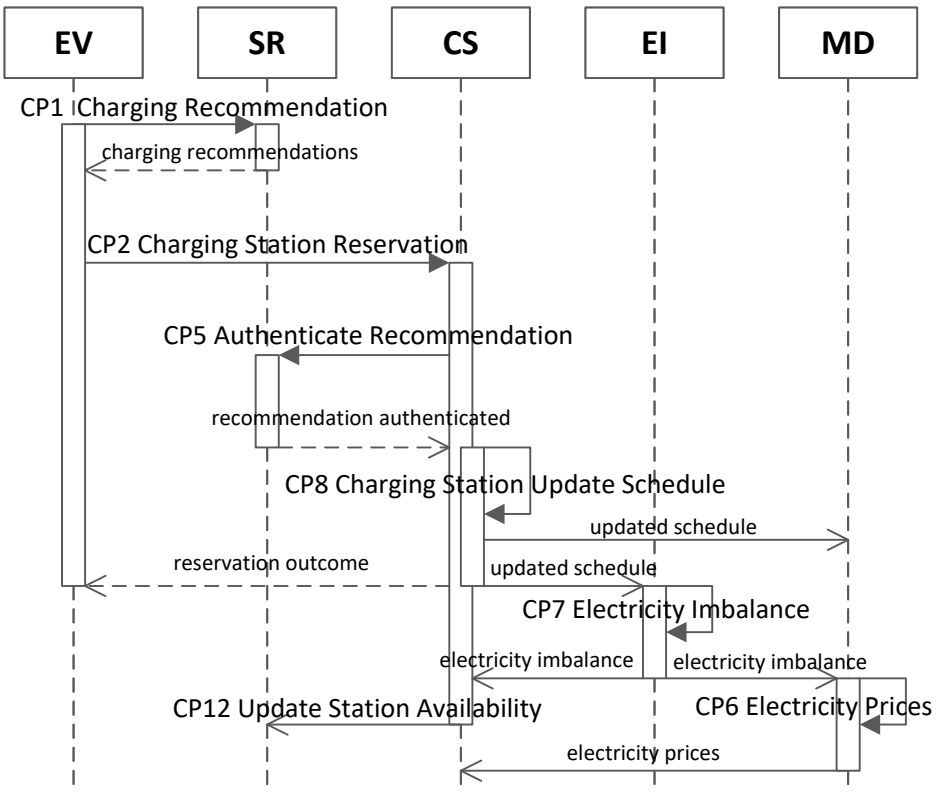

Figure 6. Agent interactions involved in reserving a charging slot.

The execution of this scenario is initiated when the EV agent sends a *CP1 Charging Recommendation* message, with its preferences and its location, to the SR. Then, the SR responds with the *charging recommendations*, which contain a list of the best-matched charging stations that are available for charging and that match the EV's preferences. The EV agent selects one charging station from the recommendations list and informs the CS of its selection by sending a tuple with the recommendation that was chosen and information about its battery state and the desired preferences, using *CP2 Charging Station Reservation*.

Once the CS agent has this information, it submits a *CP5 Authenticate Recommendation* to the SR in order to validate that the recommendation the EV selected is genuine. The SR checks the recommendations that it provided and responds accordingly to the CS agent; if the recommendation is valid, it notifies the CS agent that the *recommendation has been authenticated*.

Then, the CS calculates its new energy needs by performing a *CP8 Charging Station Update Schedule*, and sends the *updated schedule* to the EI and MD. Simultaneously, a *reservation outcome* is sent from the CS to the EV agent with the reservation information, the charging schedule, and the buy and sell prices for each time interval of the charging session.

The new reservation induces changes in the CS energy needs; thus, it sends a *Charging Station Update Schedule* to the EI and MD with the new consumption and production information. Then, the EI and MD respond with a *schedule update outcome* regarding their ability to record that CS's change.

The SR responds to the CS with an acknowledgment *availability update outcome*. Afterwards, the EI calculates the *CP7 Electricity Imbalance* for the time intervals that changed, and broadcasts an *electricity imbalance* with the updates to all CSs and the MD.

In turn, the CS informs the SR with a *CP12 Update Station Availability* about its new availability for charging slots. Finally, the MD executes the *CP6 Electricity Prices* protocol to calculate prices by taking into account the updated imbalance, and announces them to all CSs.

4.5. Implemented Agent Strategies

To validate the applicability of our framework, it was necessary to test the incorporation of different agent strategies and compare their effects on the ecosystem's behavior via simulations. For this purpose, we implemented two pricing algorithms that could have been used by an MD agent in the real world in order to test if and how they affected the stability of the grid, i.e., whether they led to more balanced production and consumption. We also implemented three charging scheduling approaches that were able to determine when and how much energy was exchanged between the CS and EV agents.

4.5.1. Electricity Price Calculation Algorithms Implemented by the Mechanism Design Agent

(A) NRG-Coin pricing algorithm: A mechanism inspired by [51], which aims to incentivize the participants to balance supply and demand. For its implementation, we used the parameter values as presented in [52]. In more detail, let the aggregate supply and demand at each time interval t be S_t, and D_t and the individual agent i's desirable amounts of energy for selling and buying be s_t^i and d_t^i. The closer D_t and S_t are, the better prices are offered for buying and selling. The price for selling energy is:

$$P_t^{sell}(s_t^i, S_t, D_t) = (0.1 \cdot s_t^i) + \frac{0.2 \cdot s_t^i}{e^{(\frac{S_t - D_t}{D_t})^2}}$$

and the price for buying energy is:

$$P_t^{buy}(d_t^i, S_t, D_t) = \frac{(0.65 \cdot D_t) \cdot d_t^i}{D_t + S_t}$$

(B) Adaptive pricing algorithm: This is a pricing mechanism proposed by [53]. According to this mechanism, one estimates the evaluation of energy with respect to the cost induced by the EV agents by calculating an $\hat{\alpha}$ value:

$$\hat{\alpha} = \frac{\sum_{t=2}^{N} \frac{P_1^{buy} - P_t^{buy}}{2 \cdot (d_1^{i'} - d_t^{i'})}}{N-1}$$

where N is the number of time intervals in the planning horizon, and $d_t^{i'}$ is the demand of EV agent i during the interval t. The mechanism can adjust prices to motivate agents to charge their EVs when there is an energy surplus on the grid. Buying prices for the intervals $t \in \{1, \ldots, T\}$ are given by:

$$\hat{P}_1^{buy} - 2 \cdot \hat{\alpha} \cdot (S_1 - D_1) = \ldots = \hat{P}_T^{buy} - 2 \cdot \hat{\alpha} \cdot (S_T - D_T)$$

Note that *adaptive pricing* does not determine prices for *selling* energy back to the grid—i.e., it does *not* support V2G activities.

4.5.2. Charging Scheduling Approaches

We now shift our focus to agent (EV) charging scheduling strategies, i.e., strategies tofordeciding the time intervals at which to charge the vehicles' batteries[7]. Specifically, the different charging scheduling methods that we tested in our simulations were the following:

(A) First slot: According to this method, EVs choose to charge their batteries immediately after they connect to a charger, regardless of how cheap or expensive electricity is at that particular time instant.

(B) Lowest Prices: In this case, EVs attempt to reduce overall costs by taking into account prices during the whole period that they are connected to a charger and end up selecting the intervals for which energy prices are the lowest.

(C) V2G : According to this approach, EVs are allowed to discharge their batteries and provide energy back to the grid considering high price time intervals, and select to charge it back at intervals with lower prices within the period of their connection to a charger. For this purpose and inspired by [26,57], we used linear programming to minimize an objective function representing charging costs in the presence of constraints regarding the EV preferences and charging specifications.

The cost function that is minimized is:

$$\min \sum_{t}^{T} C_t^{G2V} + C_t^{deg} - I_t^{V2G} \tag{1}$$

subject to:

$$C_t^{G2V} = d_t^{G2V} * P_{max}^{G2V} * p_t^{buy} * dt \tag{2}$$

$$C_t^{deg} = f^{deg} * d_t^{V2G} * P_{max}^{V2G} * dt \tag{3}$$

$$I_t^{V2G} = d_t^{V2G} * P_{max}^{V2G} * p_t^{sell} * dt \tag{4}$$

$$d_t^{G2V} + d_t^{V2G} \leq 1, \;\; d_t^{G2V}, d_t^{V2G} \in [0,1] \tag{5}$$

$$\sum_{t}^{T} d_t^{G2V} * P_{max}^{G2V} * dt - d_t^{V2G} * P_{max}^{V2G} * dt = E_{need} \tag{6}$$

$$\sum_{t}^{k} d_t^{G2V} * P_{max}^{G2V} * dt - d_t^{V2G} * P_{max}^{V2G} * dt + E_{init} \leq c_{max}, k \in [1, T] \quad (7)$$

$$\sum_{t}^{k} (d_t^{G2V} * P_{max}^{G2V} * dt - d_t^{V2G} * P_{max}^{V2G} * dt) + E_{init} \geq c_{min}, k \in [1, T] \quad (8)$$

where t is the charging interval of the charging period, C_t^{G2V} is the cost of charging, C_t^{deg} is the battery degradation cost, and I_t^{V2G} is the profit earned by selling energy to the grid. d_t^{G2V} and d_t^{V2G} are decision variables for G2V and V2G in our optimization problem and they can take values between zero and one, and intermediate values are assigned when it is optimal to charge or discharge at a fraction of the max charging (P_{max}^{G2V}) or discharging (P_{max}^{V2G}) power; p_t^{buy} and p_t^{sell} are the buying and selling prices of energy. f^{deg} is a degradation factor, based on the method presented in [58], which is used to evaluate the degradation cost C_t^{deg}, and dt is the duration of each time interval.

The constraints in expressions (5)–(8) must be satisfied during the EV scheduling optimization process. In (5), it is guaranteed that an EV will charge, discharge, or stay idle in each time interval. It can charge and then discharge in the same interval but this is unusual, because there must be a selling price greater than the buying price within the same interval. Constraint (6) states that at the end of the charging session the EV battery must be charged at the desired capacity E_{need} that the owner has set as a target. In constraints (7) and (8), we limit the allowable range of the battery charging state to be between the minimum (c_{min}) and the maximum (c_{max}) capacity by adding the net energy that has been received up to the end of each time interval, plus the initial amount of energy already stored in the battery.

5. Experimental Evaluation

In this section, we present four use-cases that showcase the real-world applicability of our solution. Through these use-cases, we evaluate and compare the implemented strategies that we discussed previously. The programming language that we chose to use was Python, and the datasets we utilized originated from a collection of real data from a number of publicly available online resources[8]; The simulation time horizon for each use-case was ten days The experiments were executed on a PC with an AMD Ryzen 5 1500X @ 3.5 GHz processor and 8 GB of RAM.

Overall, agents were implemented as different programs that were deployed in independent docker containers. Such containers could either be hosted on cloud infrastructure executed locally on the stakeholder's premises. Furthermore, to investigate the effects of different strategies, we set up simulations to test and evaluate particular desired algorithms. This was made possible via additional orchestrator scripts that utilized the API of the IoT platform to register, deploy, and configure services in batches. Moreover, the agent actions and final outcomes in the simuations were logged so that we could perform a post-hoc analysis of the results. For the purposes of the simulations, we also define the duration of a simulated hour in actual time, which in our experiments was set to two seconds. In an actual system deployment, the required data would be obtained in real-time via sensor measurements or user input forms. In our simulations, however, this information was retrieved from the datasets indicated above.

Our first use-case served the purpose of comparing the various EV charging scheduling methods, using the two pricing mechanisms that we described above. As explained above, (*i*) the *first slot* charging scheduling method involves charging the EV during the first slot when it is connected to a charger; (*ii*) the *lowest prices* method involves charging the EV during intervals with the lowest consumption price; and (*iii*) the *V2G* charging scheduling method allows an EV to also sell back to the grid some of its energy and recharge later, as long as it is ensured that it is profitable to do so (given the price difference between discharge and recharge intervals). Figure 7 depicts the average cumulative EV costs

for the entire planning horizon when the MD agent implemented the *NRG-Coin* pricing mechanism; whereas Figure 8 depicts these costs in the case in which the *adaptive* pricing mechanism was adopted. It is clear that, regardless of the pricing mechanism in use, the *first slot* method resulted in the highest costs for the EV. This was expected since, in this case, the EV agent chooses to charge their vehicle immediately, without taking into account the energy price. At the same time, by adopting the *lowest prices* method, the total cost of EV charging dropped by about 33% by the end of the time horizon for both pricing mechanisms examined. Regardless of the difference in the magnitude of the prices for the two mechanisms, for (*adaptive* responses to higher electricity prices), the drop in costs was relatively the same, and it was accrued via the better utilization of cheaply produced energy (e.g., from renewable sources). The difference in the absolute price values between the two mechanisms was not so important in our simulations since the two runs were independent from one another, and were subject to change according to the parametrization of the pricing functions that would have been adopted by real-world businesses. What *is* of interest is the relative difference between the prices of the time intervals for each mechanism, which, as we show in our third use-case below, ultimately affects the decisions of the EV agents in a similar manner. Finally, by allowing *V2G* operations, the charging costs dropped even more, being 15% lower than those of the *lowest prices* method, and 43% lower than those of the *first slot* method. This was not tested in the case of the *adaptive* pricing mechanism, since this mechanism was originally designed for smart charging and does not support V2G.

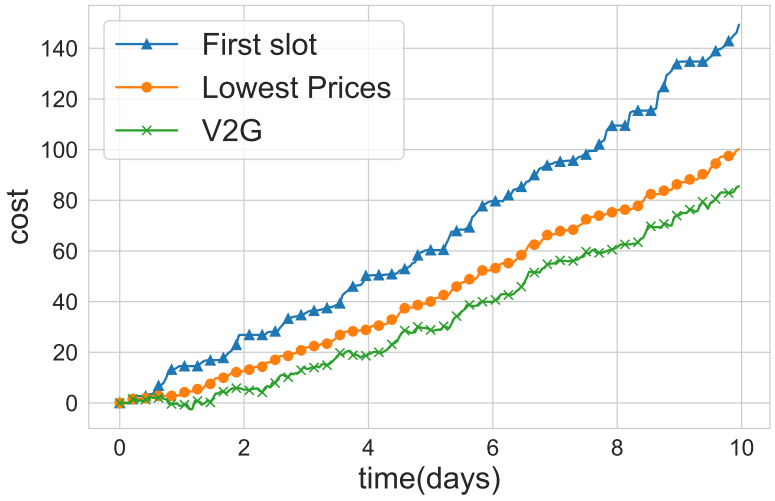

Figure 7. Average cumulative cost per EV for different charging scheduling methods (NRG-Coin pricing).

We then studied the impact of the charging scheduling methods on the aggregate energy imbalance. Our baseline was a grid imbalance without EV demand. We calculated the sum of the absolute imbalance values among the intervals, the sum of only the positive imbalance intervals (i.e., the total exported or "wasted" energy), and the sum of only the negative intervals (i.e., the total energy imports).

As seen in Table 1 we observed significant and differing impacts of different EV charging strategies on the energy imbalance. The employment of the *first slot* method by the EVs clearly affected the system negatively: the total imbalance was increased by 7%, and the imported energy increased by 104.2%, meaning that more than double the energy had to be produced to meet demand. At the same time, however, the EVs absorbed energy that would otherwise be wasted; thus, the corresponding amount dropped by 21.8%.

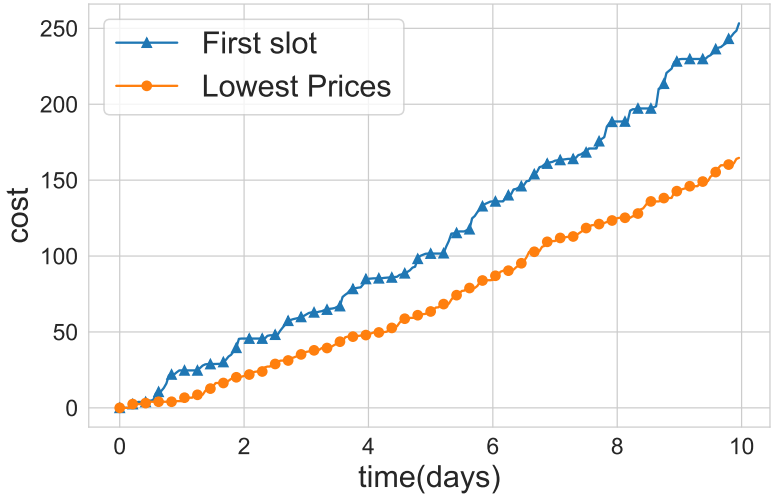

Figure 8. Average cumulative cost per EV for different charging scheduling methods (adaptive pricing).

In contrast, the use of the *lowest prices* method demonstrated a tangible positive effect on the system: there was a drop of 31.44% in the energy imbalance, whereas the amount of wasted energy was reduced by 45.6%. However, "imported" energy increased as well, albeit by a much smaller value, i.e., by 16.4%. This is because the (one hundred) EVs did introduce a significant demand that had to be met, whereas their charging strategy did not take the potential high energy prices into full account, nor did they contribute energy to tackle grid shortages.

Table 1. Energy differences in charging scheduling methods compared to the "no EV" baseline. The MAPE of the original imbalance curve was 63.9%.

Method	Imbalance	Wasted	Imported	MAPE
First Slot	+7.0%	−21.8%	+104.2%	−12.4%
Lowest Prices	−31.4%	−45.6%	+16.4%	−44.5%
V2G	−37.3%	−49.1%	+2.5%	−55.7%

An even more positive impact on the system imbalance was obtained when using the more "intelligent" *V2G* charging strategy. The EVs now optimized their charging/discharging plans, taking energy prices into full account and also contributing energy back to the grid. As a result, there was an even greater reduction in the grid imbalance of 37.3%. Moreover, the wasted energy was now reduced by 49.1% and there was only a slight increase of 2.5% in "imported" energy. Furthermore, this method demonstrated a reduction in the *mean absolute percentage error* (*MAPE*) that was significantly larger than those of the previous two methods. MAPE measures the difference between the induced imbalance and a totally flat curve with a value of zero, which resembles perfect matching between supply and demand. This is clear when plotting the imbalance across the time horizon for each method, as we have shown in Figure 9. Indeed, it is clearly visible there that the *V2G* strategy resulted in much lower induced peaks in the imbalance curve than those induced by *First Slot* or *Lowest Prices* methods.

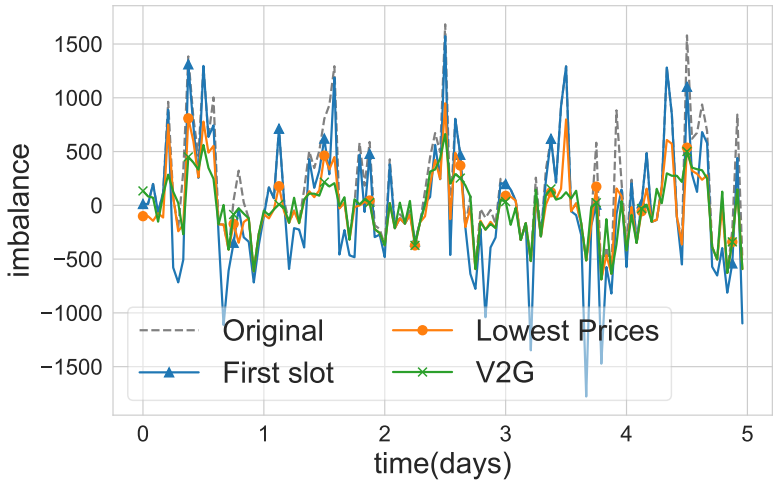

Figure 9. Tackling the energy imbalance using different charging scheduling methods.

In the second use-case, we measured the total cumulative costs of EVs when increasing the duration of their connection to chargers up to 12 h compared to the original data (i.e., without performing any charging rescheduling), following the three different charging scheduling methods as in the first use-case.

The results shown in Figures 10 and 11 demonstrate that by increasing the duration of connection, the *lowest prices* and *V2G* methods managed to gradually reduce the battery charging costs. This occurred since the longer an EV is connected to a charger, the greater the probability that it will be able to find the most advantageous intervals at which to buy energy from the grid—and also to sell it back to the grid in the case of V2G. As anticipated, again, the *V2G* method led to lower charging costs than the other two methods, and the difference (mirroring *V2G*'s advantages) increased as the duration of the connection to a charger became longer.

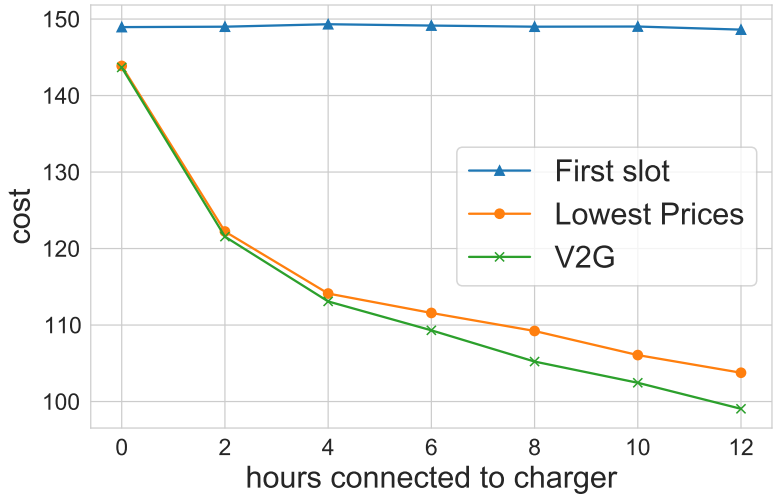

Figure 10. Cost comparison of varying time periods for which EVs were connected to chargers (NRG-Coin pricing).

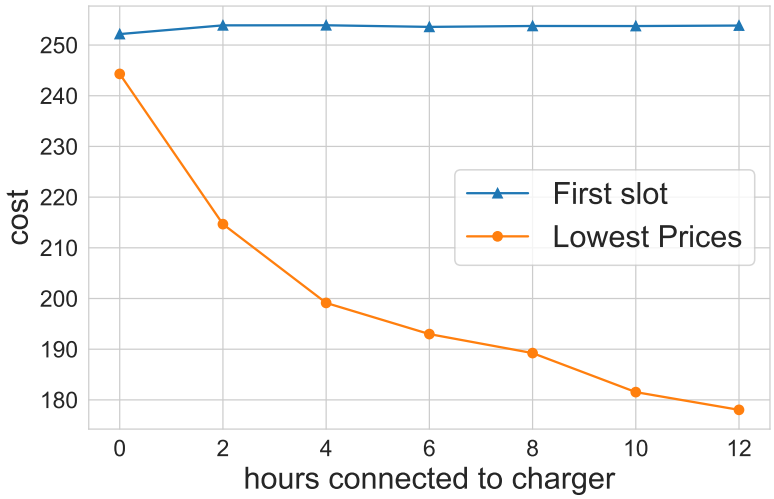

Figure 11. Cost comparison of varying time periods for which EVs were connected to chargers (adaptive pricing).

In our third use-case, we compare the performance of different pricing strategies for the MD agent. Specifically, we tested the *NRG-Coin* pricing and the *adaptive pricing* methods described in Section 4.5. Both algorithms aim to balance supply and demand by setting higher consumption prices during intervals in which there is a negative imbalance and lower consumption prices for intervals in which the imbalance is positive. In this use case, we assumed that EVs were charged following the *lowest prices* scheduling approach[9]. Assuming that EV agents were rational and aimed to reduce their expenses, the application of the two pricing algorithms resulted in demand being shifted to utilize the generated energy more effectively, thus leading to smaller peaks in the imbalance curve. Figure 12 shows that the algorithms had similar effects on the stability of the grid. In Table 2, we can observe similar behavior, namely, reducing the wasted energy, with this mechanism slightly outperforming the *NRG-Coin* pricing mechanism in terms of imported energy and MAPE reductions.

Table 2. Pricing Algorithms: energy differences compared to the "no EVs" baseline.

Method	Imbalance	Wasted	Imported	MAPE
NRG-Coin	−31.4%	−45.6%	+16.4%	−44.5%
Adaptive Pricing	−31.3%	−45.6%	+17.1%	−42.7%

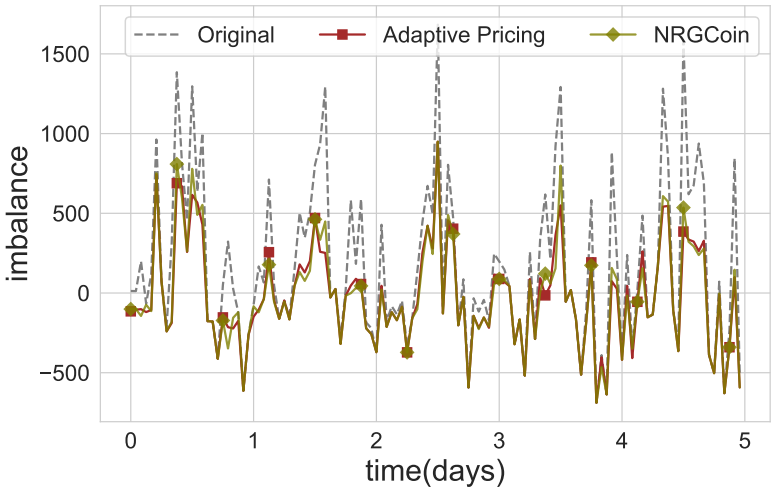

Figure 12. Comparison of adaptive pricing and NRG-Coin pricing mechanisms.

Finally, the fourth use-case was conducted to examined the scalability of our framework in terms of communication complexity as the supported EV population increased. To this end, we plotted in Figure 13 the total number of exchanged messages required for the scheduling of EV charging using our proposed cooperation protocols over a 10 day period, against increasing numbers of supported EVs. It is clear from Figure 13 that there was a *linear* increase in the number of messages exchanged (over 10 days) as the number of EVs increased. As such, this result attests to the scalability of our approach.

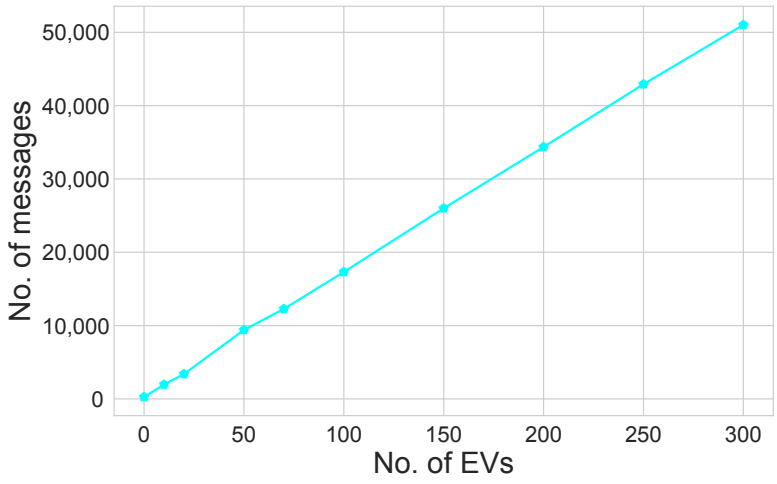

Figure 13. Message count for 10 days vs. the number of EVs.

6. Discussion: Enabling Digital Twins and Real-World Integration

In this study, we have taken several steps towards enabling the use of digital twins for V2G/G2V, and more generally for the smart grid domain, as well as enabling their real-world deployment. We know that currently there are a number of limitations in regard to the real-world application of V2G , e.g., the relatively small number of EVs, the existence of cheaper and better alternatives, high cost and complexity, consumer resistance, etc. [60]. The work presented in this study constitutes a step towards delivering V2G scenarios in the real world since it helps to reduce administration complexity and end-user costs, as we showed in our analysis.

First, we presented the inter-agent and intra-agent control models of ASEME that can be used for different development goals:

- The inter-agent control model (EAC): This can be shared with third parties. It contains both the activities (the basic states) and the topics that must be used by the agents to effectively participate in a protocol.
- The intra-agent control model (IAC): This is used for modeling the agent. It reuses parts of the EAC to ensure that the developed agents can be seamlessly integrated into the open MAS.
- Note that an EAC implementation can also be reused "as-is" by developers, who can use the same platform (e.g., Python) for developing their agents.

In energy markets, stakeholders develop their own business models and can have diverse goals in negotiating with regard to their consumption and offers, and can join and leave the system at any time. All these characteristics point to the relevance of *open multiagent systems* and agent technology in general [5,6]. These open interactions call for common ontologies, communication protocols, and suitable broker and coordination infrastructures to enable interoperability [61].

We have used existing standards (i.e., MQTT) to propose an architecture that is truly open, allowing players to reuse existing agents or build their own. The MAS V2G/G2V framework that we presented in Section 3.3 defines specific communication protocols among the stakeholders and is the basis of an implementation approach that allows for

the evaluation and comparison of different functionalities that could be offered by the various modules in such a setting. By focusing on openness, we allow the extension and customization of such protocols so that they comply with the diversity of real-world approaches. In contrast to previous studies, the semantic schemes of the services offered to agents have been described here in detail. Using our framework as a basis, designers may evaluate their own algorithms, e.g., for generating charging recommendations, for the scheduling of EV charging on a large scale, or for analyzing the effects of alternative pricing strategies. Taken together, the openness and extendability of our proposed architecture, along with its experimental evaluation in simulated settings, as presentred in this paper, verify its appropriateness and potential for real-world integration.

We can also provide some guidelines for the deployment of the system. The first step is to determine the entity that will deploy, manage, and maintain the system's backbone, that is, the IoT platform. This entity can be any independent service provider, a power grid regulator, or a government agency, which will also be responsible for giving access to new users. Potential users include all the stakeholders that we have identified in our architecture (see Section 3). Each stakeholder may purchase or develop (or outsource) an application incorporating an agent that will represent them to the platform.

For example, car owners may download an appropriate application in which they can create a profile and connect to the platform. The complexity of the application can be determined based on their needs. A simple application, for example, would reserve a place after a user request. A more sophisticated application could also employ machine-learning techniques to learn the user's habits and automatically reserve a place when needed by informing its user.

Moreover, additional sensors and actuators must be interfaced with the IoT platform, allowing agents to receive measurements and submit actions. Examples include the sensory equipment of EV batteries, the charger controllers, and the various smart meters installed in the buildings. The EV agent could be deployed in an owner-controlled machine, e.g., inside the EV, and appropriate encryption could be established with regard to the messages exchanged in order to ensure the protection of private information. Privacy is also a concern for the other stakeholders in the ecosystem as they too exchange private data, for example, the buying and selling prices of each charging station, buildings' consumption profiles, and so on.

Therefore, our approach is readily deployable and can support real-world trials, thus enabling the use of digital twins in the smart grid domain [12,13]. The engineering approach that we followed possesses the following capabilities that are important in regard to digital twins [62]:

- It allows for synchronization between the physical world and the cyber domain. The design of our system is such that it can incorporate real-time updates even in simulation mode, which reflect the changes in the real world. For example, if a new charging station emerges, then a new CS agent appears in the system and starts pursuing its goals.
- It allows the co-simulaton and modeling of subsystems. This V2G/G2V system could be considered as a subsystem for the overall smart grid, or as an instance of many interconnected smart grids. These grids could be hierarchical, i.e., the available power could be determined by a producer or by a higher authority that manages grids. Moreover, one of the participating agents, e.g., a charging station, could itself be a multi-agent system of charging connectors participating in the V2G/G2V system, in the form of one station that can accommodate many vehicles.
- The simulation mode can be used to test and compare various agent strategies, e.g., for coordinated charging/discharging [48,63] or for dealing with battery degradation issues [64,65], and in general for any approach that must be tested before it is finally deployed.

- Each resource is modeled as an agent, thus allowing the system to be scaled regardless of the complexity of interactions. The system scales linearly, as we demonstrated in our experimental evaluation (Figure 13).

Altogether, in comparison with the state of the art, in this study we have put forward a functional system that (*a*) enables large-scale V2G/G2V, (*b*) is supported by the use of a digital twin for simulations, (*c*) uses open protocols for easy adoption and realization by business stakeholders, and *(d)* allows each participant to adopt their own strategies and algorithms.

7. Conclusions and Future Work

In this paper we have demonstrate how to engineer an open system for the V2G/G2V energy transfer problem domain, and provided its architecture and the implementations of agents as flexible microservices that are interconnected by means of an IoT platform.

We illustrated the development process, starting by capturing the requirements of such a system by reviewing the stakeholders of the application domain and their goals. Then, using the ASEME methodology for IoT-enabled multiagent systems development [18,19], we proceeded to analyze the requirements, proposing the architecture, and then developing the prototype with the innovation of enabling the support of large-scale deployments using IoT technology.

We achieved our objective of proposing an open architecture [7] and od covering diverse business models via the definition of a number of key agent types and the development of open protocols.These types and protocols can be easily extended by any interested stakeholder, according to their needs. Our simulation experiments verify the applicability of our approach, and we have outlined the steps to be taken for its effective integration in the real world, along with the benefits arising from such an integration.

As the first item of our future work, we intend to populate the agents' components with actual machine-learning and recommender algorithms, in order to support the decision-making of agents in relation to various activities and tasks. Furthermore, the deployment and comparison (in simulation mode) of heterogeneous agent behaviors would be of significance, e.g., comparing various strategies for charging/discharging, or comparing different pricing mechanisms adopted by different charging networks. The choice of which behaviors to simulate could be performed according to the corresponding cultural and social values that are prevalent at different deployment sites [60,66].

Another interesting line of work would be to augment our solution with specialized graphical user interfaces. Such interfaces would be quite different for each agent role. For example, the interface for EVs would focus on usability for the elicitation of preferences, whereas that of the MD would focus on data and market analytics, and there could be different versions for the same agent type, as long as they all followed the protocols we defined via their back-end functionality.

Moreover, the openness of our architecture allows for the creation of alternative or additional protocols—e.g., protocols to serve the specific needs of various real-world stakeholders, and to help conceptualize and realize digital twins of actual real-world systems, of which the systematic analysis and the recognition of related opportunities and shortcomings are left as future works. Indeed, it is our aim to conduct a pilot, real-world study of our architectural framework. This will allow us to better evaluate its applicability and to identify interesting business models and necessary extensions of the framework.

The architecture can be extended to allow the incorporation of service-level agreements (SLAs) in the form of smart contracts between the different stakeholders. Smart contracts can define the obligations of the contracting parties, as well as issues related to the quality of service, such as performance, availability, and security [67]. The stakeholders' functionality, modeled using statecharts, allows for the automatic monitoring of the execution of SLAs and the handling of possible violations, e.g., with property statecharts [68] or Symboleo [69].

Finally, it would be interesting to employ our generic engineering approach to different application domains. For instance, this approach could be utilized within the domain of

digital twins for manufacturing, in which agent-based modeling with the use of statecharts has recently been proposed [62]. More generally, we believe that the ideas presented in this paper can be of use and tested in any domain that calls for the engineering of IoT-based open MAS architectural frameworks. In this direction, it would be interesting to develop a code generator for an IoT platform for ASEME models, similarly to the research conducted on an automatic code generator for the JADE framework [70].

Author Contributions: The authors confirm their contributions to the paper as follows: Conceptualization, N.I.S., C.A. and G.C.; investigation, N.I.S., C.A. and G.I.; methodology, N.I.S., C.A., G.I. and G.C.; software, N.I.S., C.A. and G.I.; validation, N.I.S., C.A. and G.I.; supervision, N.I.S. and G.C.; data curation, C.A. and G.I.; writing—original draft, N.I.S., C.A. and G.I.; writing—review and editing, N.I.S., C.A., G.I. and G.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The associated code is available online: https://github.com/iatrakis/IoT-V2G-G2V (accessed on 18 March 2023).

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

Notes

1 https://jade.tilab.com/ (accessed on 18 March 2023)

2 https://github.com/SwitchEV/RISE-V2G (accessed on 18 March 2023).

3 The message queue transport telemetry (MQTT) protocol is an OASIS standard messaging protocol for the Internet of Things, mqtt.org https://mqtt.org/ (accessed on 18 March 2023).

4 REpresentational State Transfer (REST) over Hypertext Transfer Protocol (HTTP).

5 More detailed descriptions of the inter- and intra-agent controls and a detailed description of the protocols, including the message syntax and semantics, can be found in our online repository: https://github.com/iatrakis/IoT-V2G-G2V (accessed on 18 March 2023).

6 This is very useful for experimentation with large agent populations.

7 Several battery charging models have been introduced in the past, considering load transfer constraints and mobility patterns (see, e.g., [54–56]). In our study, we do not require any particular models for calculating travel duration and battery SOC, since such values are acquired directly as sensor measurements. Of course, any of the battery charging models proposed to date could be incorporated in each individual agent implementation if deemed necessary by the strategy.

8 Specifically, consumption and production data originated from a synthetic dataset generator [59], which was trained on information from the ENTSOE https://transparency.entsoe.eu (accessed on 18 March 2023) platform, and on EV data from the MyElectricAvenue https://eatechnology.com/resources/projects/my-electric-avenue/ (accessed on 18 March 2023) project.

9 This was selected because the *lowest prices* was shown in the first use-case to perform better than the *first slot* method. Note that using the top-scoring *V2G* scheduling method was not an available option, since one of the pricing methods we intended to evaluate in this third use-case, *adaptive pricing*, does not support V2G activities (cf. Section 4.5).

References

1. Burke, M.J.; Stephens, J.C. Energy democracy: Goals and policy instruments for sociotechnical transitions. *Energy Res. Soc. Sci.* **2017**, *33*, 35–48. [CrossRef]
2. Ketter, W.; Collins, J.; Reddy, P. Power TAC: A competitive economic simulation of the smart grid. *Energy Econ.* **2013**, *39*, 262–270. [CrossRef]
3. Ghasempour, A. Internet of things in smart grid: Architecture, applications, services, key technologies, and challenges. *Inventions* **2019**, *4*, 22. [CrossRef]
4. Espe, E.; Potdar, V.; Chang, E. Prosumer Communities and Relationships in Smart Grids: A Literature Review, Evolution and Future Directions. *Energies* **2018**, *11*, 2528. [CrossRef]
5. Wooldridge, M.; Jennings, N.R. Intelligent Agents: Theory and Practice. *Knowl. Eng. Rev.* **1995**, *10*, 115–152. [CrossRef]
6. Huynh, T.D.; Jennings, N.R.; Shadbolt, N.R. An integrated trust and reputation model for open multi-agent systems. *Auton. Agents-Multi-Agent Syst.* **2006**, *13*, 119–154. [CrossRef]

7. Hattab, S.; Chaari, W.L. A generic model for representing openness in multi-agent systems. *Knowl. Eng. Rev.* **2021**, *36*, e3. [CrossRef]
8. Mahela, O.P.; Khosravy, M.; Gupta, N.; Khan, B.; Alhelou, H.H.; Mahla, R.; Patel, N.; Siano, P. Comprehensive overview of multi-agent systems for controlling smart grids. *CSEE J. Power Energy Syst.* **2020**, *8*, 115–131.
9. Chalkiadakis, G.; Elkind, E.; Wooldridge, M. *Computational Aspects of Cooperative Game Theory*; Synthesis Lectures on Artificial Intelligence and Machine Learning; Morgan & Claypool Publishers: San Rafael, CA, USA, 2011.
10. Hossein Motlagh, N.; Mohammadrezaei, M.; Hunt, J.; Zakeri, B. Internet of Things (IoT) and the energy sector. *Energies* **2020**, *13*, 494. [CrossRef]
11. Noura, M.; Atiquzzaman, M.; Gaedke, M. Interoperability in internet of things: Taxonomies and open challenges. *Mob. Netw. Appl.* **2019**, *24*, 796–809. [CrossRef]
12. Brosinsky, C.; Westermann, D.; Krebs, R. Recent and prospective developments in power system control centers: Adapting the digital twin technology for application in power system control centers. In Proceedings of the 2018 IEEE International Energy Conference (ENERGYCON), Limassol, Cyprus, 3–7 June 2018; pp. 1–6.
13. Zhou, M.; Yan, J.; Feng, D. Digital twin framework and its application to power grid online analysis. *CSEE J. Power Energy Syst.* **2019**, *5*, 391–398. [CrossRef]
14. Shahinzadeh, H.; Moradi, J.; Gharehpetian, G.B.; Nafisi, H.; Abedi, M. IoT architecture for smart grids. In Proceedings of the 2019 International Conference on Protection and Automation of Power System (IPAPS), Tehran, Iran, 8–9 January 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 22–30.
15. Nisan, N. Introduction to Mechanism Design (for Computer Scientists). In *Algorithmic Game Theory*; Nisan, N., Roughgarden, T., Tardos, E., Vazirani, V., Eds.; Cambridge University Press: Cambridge, UK, 2007; pp. 209–242.
16. Akasiadis, C.; Pitsilis, V.; Spyropoulos, C.D. A Multi-Protocol IoT Platform Based on Open-Source Frameworks. *Sensors* **2019**, *19*, 4217. [CrossRef]
17. Akasiadis, C.; Iatrakis, G.; Spanoudakis, N.; Chalkiadakis, G. An Open MAS/IoT-Based Architecture for Large-Scale V2G/G2V. In *The PAAMS Collection, Proceedings of the Advances in Practical Applications of Agents, Multi-Agent Systems, and Complex Systems Simulation, L'Aquila, Italy, 13–15 July 2022*; Dignum, F., Mathieu, P., Corchado, J.M., De La Prieta, F., Eds.; Springer International Publishing: Cham, Switzerland, 2022; pp. 3–14.
18. Spanoudakis, N.I.; Moraitis, P. The ASEME Methodology. *Int. J.-Agent-Oriented Softw. Eng.* **2022**, *7*, 79–107. [CrossRef]
19. Spanoudakis, N.; Moraitis, P. Engineering Ambient Intelligence Systems Using Agent Technology. *IEEE Intell. Syst.* **2015**, *30*, 60–67. [CrossRef]
20. Zambonelli, F. Key Abstractions for IoT-Oriented Software Engineering. *IEEE Softw.* **2017**, *34*, 38–45. [CrossRef]
21. Savaglio, C.; Ganzha, M.; Paprzycki, M.; Bădică, C.; Ivanović, M.; Fortino, G. Agent-based Internet of Things: State-of-the-art and research challenges. *Future Gener. Comput. Syst.* **2020**, *102*, 1038–1053. [CrossRef]
22. Fortino, G.; Savaglio, C.; Spezzano, G.; Zhou, M. Internet of things as system of systems: A review of methodologies, frameworks, platforms, and tools. *IEEE Trans. Syst. Man Cybern. Syst.* **2020**, *51*, 223–236. [CrossRef]
23. International Energy Agency. *Global EV Outlook: Towards Cross-Modal Electrification*; IEA: Paris, France, 2018.
24. Danner, D.; Duschl, W.; de Meer, H. Fair charging service allocation for electric vehicles in the power distribution grid. In Proceedings of the e-Energy '19, Phoenix, AZ, USA, 25–28 June 2019; pp. 406–408.
25. Sarkar, R.; Saha, P.K.; Mondal, S.; Mondal, A. Intelligent Scheduling of V2G, V2V, G2V Operations in a Smart Microgrid. In Proceedings of the e-Energy '20, Virtual Event, 22–26 June 2020; pp. 417–418.
26. Karfopoulos, E.L.; Hatziargyriou, N.D. A Multi-Agent System for Controlled Charging of a Large Population of Electric Vehicles. *IEEE Trans. Power Syst.* **2013**, *28*, 1196–1204. [CrossRef]
27. Rigas, E.S.; Karapostolakis, S.; Bassiliades, N.; Ramchurn, S.D. EVLibSim: A tool for the simulation of electric vehicles' charging stations using the EVLib library. *Simul. Model. Pract. Theory* **2018**, *87*, 99–119. [CrossRef]
28. Jordán, J.; Palanca, J.; del Val, E.; Julian, V.; Botti, V. MASEV: A MAS for the Analysis of Electric Vehicle Charging Stations Location. In *The PAAMS Collection, Proceedings of the Advances in Practical Applications of Agents, Multi-Agent Systems, and Complexity, Toledo, Spain, 20–22 June 2018*; Demazeau, Y., An, B., Bajo, J., Fernández-Caballero, A., Eds.; Springer International Publishing: Cham, Switzerland, 2018; pp. 326–330.
29. Kamboj, S.; Kempton, W.; Decker, K.S. Deploying power grid-integrated electric vehicles as a multi-agent system. In Proceedings of the 10th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2011), Taipei, Taiwan, 2–6 May 2011; pp. 13–20.
30. Bellifemine, F.L.; Caire, G.; Greenwood, D. *Developing Multi-Agent Systems with JADE*; Wiley Series in Agent Technology; John Wiley and Sons Ltd.: Hoboken, NJ, USA, 2007.
31. Papadopoulos, P.; Jenkins, N.; Cipcigan, L.M.; Grau, I.; Zabala, E. Coordination of the Charging of Electric Vehicles Using a Multi-Agent System. *IEEE Trans. Smart Grid* **2013**, *4*, 1802–1809. [CrossRef]
32. Lipman, T.; Callaway, D.; Peffer, T.; von Meier Alexandra, E.A. *Open-Source, Open-Architecture Software Platform for Plug-In Electric Vehicle Smart Charging in California*; California Energy Commission: Sacramento, CA, USA, 2020.
33. Van Aubel, P.; Poll, E. Security of EV-charging protocols. *arXiv* **2022**, arXiv:2202.04631.
34. Käbisch, S.; Peintner, D.; Heuer, J.; Schmutzler, J.; Gröning, S.; Lauterbach, M. The OpenV2G Project. Available online: https://openv2g.sourceforge.net (accessed on 18 March 2023).

35. Sheppard, C.; Jenn, A. *Grid-Integrated Electric Mobility Model (GEM) v1.0*; US Department of Energy: Washington, DC, USA, 2021.
36. Lee, Z.; Johansson, D.; Low, S.H. ACN-Sim: An Open-Source Simulator for Data-Driven Electric Vehicle Charging Research. In Proceedings of the e-Energy '19, Phoenix, AZ, USA, 25–28 June 2019; ACM: New York, NY, USA, 2019; pp. 411–412.
37. *Unified Modeling Language, Superstructure, V2.1.2*; Technical Report formal/07-11-02; Object Management Group: Needham, MA, USA, 2007.
38. Busetta, P.; Howden, N.; Rönnquist, R.; Hodgson, A. Structuring BDI agents in functional clusters. In Proceedings of the International Workshop on Agent Theories, Architectures, and Languages, Orlando, FL, USA, 15–17 July 1999; Springer: Berlin/Heidelberg, Germany, 1999; pp. 277–289.
39. Braubach, L.; Pokahr, A.; Lamersdorf, W. Extending the capability concept for flexible BDI agent modularization. In Proceedings of the International Workshop on Programming Multi-Agent Systems, Utrecht, The Netherlands, 26 July 2005; Springer: Berlin/Heidelberg, Germany, 2005, pp. 139–155.
40. Padgham, L.; Winikoff, M. *Developing Intelligent Agent Systems: A Practical Guide*; Wiley Series in Agent Technology; Wiley: Hoboken, NJ, USA, 2004; p. 240.
41. Trencansky, I.; Cervenka, R. Agent Modeling Language (AML): A comprehensive approach to modeling MAS. *Informatica* **2005**, *29*, 391–400.
42. Harel, D.; Naamad, A. The STATEMATE Semantics of Statecharts. *ACM Trans. Softw. Eng. Methodol.* **1996**, *5*, 293. [CrossRef]
43. Mazouzi, H.; Seghrouchni, A.E.F.; Haddad, S. Open Protocol Design for Complex Interactions in Multi-agent Systems. In Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems: Part 2, AAMAS '02, Bologna, Italy, 15–19 July 2002; ACM: New York, NY, USA, 2002; pp. 517–526.
44. Rathor, S.K.; Saxena, D. Energy management system for smart grid: An overview and key issues. *Int. J. Energy Res.* **2020**, *44*, 4067–4109. [CrossRef]
45. Käbisch, S.; Schmitt, A.; Winter, M.; Heuer, J. Interconnections and Communications of Electric Vehicles and Smart Grids. In Proceedings of the 2010 First IEEE International Conference on Smart Grid Communications, Gaithersburg, MD, USA, 4–6 October 2010; pp. 161–166.
46. El-hawary, M.E. The Smart Grid—State-of-the-art and Future Trends. *Electr. Power Components Syst.* **2014**, *42*, 239–250. [CrossRef]
47. Weiss, G. *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*; MIT Press: Cambridge, MA, USA, 1999.
48. Carli, R.; Dotoli, M. A Distributed Control Algorithm for Optimal Charging of Electric Vehicle Fleets with Congestion Management. 15th IFAC Symposium on Control in Transportation Systems CTS 2018. *IFAC-PapersOnLine* **2018**, *51*, 373–378. [CrossRef]
49. Decker, K.; Sycara, K.; Williamson, M. Middle-agents for the internet. In Proceedings of the 15th International Joint Conference on Artificial Intelligence (IJCAI 97), Nagoya, Japan, 23–29 August 1997; pp. 578–583.
50. Spanoudakis, N.; Moraitis, P. An Agent Modeling Language Implementing Protocols through Capabilities. In Proceedings of the 2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology, Sydney, Australia, 9–12 December 2008; IEEE Computer Society: Washington, DC, USA, 2008; Volume 2, pp. 578–582.
51. Mihaylov, M.; Jurado, S.; Avellana, N.; van Moffaert, K.; de Abril, I.M.; Nowe, A. NRGcoin: Virtual currency for trading of renewable energy in smart grids. In Proceedings of the 11th International Conference on the EU energy market (EEM14), Krakow, Poland, 28–30 May 2014; IEEE: Piscataway, NJ, USA, 2014; pp. 1–6.
52. Akasiadis, C.; Chalkiadakis, G. Decentralized Large-Scale Electricity Consumption Shifting by Prosumer Cooperatives. In Proceedings of the ECAI, Hague, The Netherlands, 29 August–2 September 2016; pp. 175–183.
53. Valogianni, K.; Ketter, W.; Collins, J.; Zhdanov, D. Sustainable electric vehicle charging using adaptive pricing. *Prod. Oper. Manag.* **2020**, *29*, 1550–1572. [CrossRef]
54. Koufakis, A.M.; Rigas, E.S.; Bassiliades, N.; Ramchurn, S.D. Offline and online electric vehicle charging scheduling with V2V energy transfer. *IEEE Trans. Intell. Transp. Syst.* **2019**, *21*, 2128–2138. [CrossRef]
55. Angelidakis, A.; Chalkiadakis, G. Factored MDPS for Optimal Prosumer Decision-Making. In Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems, AAMAS '15, Istanbul, Turkey, 4–8 May 2015; International Foundation for Autonomous Agents and Multiagent Systems: Richland, SC, USA, 2015; pp. 503–511.
56. Kelly, J.C.; MacDonald, J.S.; Keoleian, G.A. Time-dependent plug-in hybrid electric vehicle charging based on national driving patterns and demographics. *Appl. Energy* **2012**, *94*, 395–405. . [CrossRef]
57. Liao, J.T.; Huang, H.W.; Yang, H.T.; Li, D. Decentralized V2G/G2V Scheduling of EV Charging Stations by Considering the Conversion Efficiency of Bidirectional Chargers. *Energies* **2021**, *14*, 962. [CrossRef]
58. Kempton, W.; Tomic, J.; Letendre, S.; Brooks, A.; Lipman, T. *Vehicle-to-Grid Power: Battery, Hybrid, and Fuel Cell Vehicles as Resources for Distributed Electric Power in California*; UC Davis Research Reports; UC Davis, Institute of Transportation Studies: Davis, CA, USA, 2001.
59. Charalambidis, G.; Akasiadis, C.; Rigas, E.S.; Chalkiadakis, G. A Realistic Dataset Generator for Smart Grid Ecosystems with Electric Vehicles. In Proceedings of the Thirteenth ACM International Conference on Future Energy Systems, e-Energy '22, Virtual Event, 28 June–1 July 2022; Association for Computing Machinery: New York, NY, USA, 2022; pp. 432–433. [CrossRef]
60. Noel, L.; Zarazua de Rubens, G.; Kester, J.; Sovacool, B.K. Navigating expert skepticism and consumer distrust: Rethinking the barriers to vehicle-to-grid (V2G) in the Nordic region. *Transp. Policy* **2019**, *76*, 67–77. [CrossRef]

61. Weyns, D.; Michel, F. Agent Environments for Multi-agent Systems–A Research Roadmap. In *Proceedings of the Agent Environments for Multi-Agent Systems IV, Paris, France, 6 May 2014*; Weyns, D., Michel, F., Eds.; Springer International Publishing: Cham, Switzerland, 2015; pp. 3–21.
62. Ragazzini, L.; Negri, E.; Fumagalli, L. Modelling Manufacturing Systems for Digital Twin Through Communicating Finite State Machines. In *Proceedings of the Service Oriented, Holonic and Multi-Agent Manufacturing Systems for Industry of the Future, Valencia, Spain, 3–4 October 2019*; Borangiu, T., Trentesaux, D., Leitão, P., Eds.; Springer International Publishing: Cham, Switzerland, 2023; pp. 85–95.
63. Stüdli, S.; Crisostomi, E.; Middleton, R.; Shorten, R. A flexible distributed framework for realising electric and plug-in hybrid vehicle charging policies. *Int. J. Control* **2012**, *85*, 1130–1145. [CrossRef]
64. Scarabaggio, P.; Carli, R.; Cavone, G.; Dotoli, M. Smart Control Strategies for Primary Frequency Regulation through Electric Vehicles: A Battery Degradation Perspective. *Energies* **2020**, *13*, 4586. [CrossRef]
65. Yan, G.; Liu, D.; Li, J.; Mu, G. A cost accounting method of the Li-ion battery energy storage system for frequency regulation considering the effect of life degradation. *Prot. Control Mod. Power Syst.* **2018**, *3*, 1–9. [CrossRef]
66. Rigas, E.; Akasiadis, C.; Vardaki, E.; Chalkiadakis, G. AI and Social Anthropology for Large-Scale Vehicle-to-Grid Schemes. In Proceedings of the 12th Hellenic Conference on Artificial Intelligence, Corfu, Greece, 7–9 September 2022; pp. 1–4.
67. Bunse, C.; Klingert, S.; Schulze, T. Greenslas: Supporting energy-efficiency through contracts. In Proceedings of the Energy Efficient Data Centers: First International Workshop, E 2 DC 2012, Madrid, Spain, 8 May 2012; Revised Selected Papers 1; Springer: Berlin/Heidelberg, Germany, 2012; pp. 54–68.
68. Mens, T.; Decan, A.; Spanoudakis, N.I. A method for testing and validating executable statechart models. *Softw. Syst. Model.* **2019**, *18*, 837–863. [CrossRef]
69. Sharifi, S.; Parvizimosaed, A.; Amyot, D.; Logrippo, L.; Mylopoulos, J. Symboleo: Towards a specification language for legal contracts. In Proceedings of the 2020 IEEE 28th International Requirements Engineering Conference (RE), Zurich, Switzerland, 31 August–4 September 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 364–369.
70. Spanoudakis, N.; Moraitis, P. Modular JADE Agents Design and Implementation Using ASEME. In Proceedings of the 2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology, Washington, DC, USA, 31 August–3 September 2010; Volume 2, pp. 221–228. [CrossRef]