Design and Implementation of Microservices for Smart EV Charging/Discharging using the SYNAISTHISI IoT Platform

Technical University of Crete
School of Electronic and Computer Engineering

Diploma Thesis:
Georgios Iatrakis

Committee:
Associate Professor Georgios Chalkiadakis (Supervisor)
Associate Professor Vasilios Samoladas
Dr. Nikolaos Spanoudakis

A thesis submitted in fulfillment of the undergraduate requirements in the: SCHOOL OF ELECTRICAL AND COMPUTER ENGINEERING .

July 2022

Abstract

The increasing number of electric vehicles and the integration of renewable sources can cause imbalance in a Grid. Its upgrade via the development of services that expand its capabilities by utilizing the new features of the system is necessary. In this work, we put forward an open multi-agent systems (MAS) architecture for the important and interesting problem of charging grid-to-vehicle (G2V) and discharging vehicle-to-grid (V2G).

To promote scalability, our solution is provided in the form of modular microservices that are interconnected using a multi-protocol Internet of Things (IoT) platform. On the one hand, the low-level modularity of Smart Grid services allows the seamless integration of different agent strategies, pricing mechanisms and algorithms; and on the other, the IoT-based implementation offers both direct applicability in real-world settings, as well as advanced analytics capabilities by enabling digital twins models for Smart Grid ecosystems. These features are provided by the SYNAISTHISI IoT platform that is used to provide the necessary infrastructure to interconnect heterogeneous agents, devices and services over the network.

We describe our MAS/IoT-based architecture and present results from simulations that incorporate large numbers of heterogeneous Smart Grid agents, which might follow different strategies for their decision making tasks. More specifically, our simulations compare different charging scheduling algorithms and pricing mechanisms that can be found in the literature. Furthermore, our framework enables the testing of various schemes in simulation mode, and can also be used as the basis for the implementation of real-world prototypes for the delivery of large-scale V2G/G2V services.

Περίληψη

Η αυξανόμενη χρήση ηλεκτρικών οχημάτων και η ενσωμάτωση ανανεώσιμων πηγών ηλεκτρικής ενέργειας μπορούν να προκαλέσουν αστάθεια σε ένα δίκτυο κατανομής ενέργειας. Η αναβάθμιση του δικτύου μέσω της ανάπτυξης υπηρεσιών που θα επεκτείνουν τις δυνατότητές του αξιοποιώντας τα νέα χαρακτηριστικά του συστήματος είναι απαραίτητη. Σε αυτή την διπλωματική εργασία προτείνουμε μια ανοιχτή αρχιτεκτονική πολυπρακτορικών συστημάτων (MAS) για τα σημαντικά και ενδιαφέροντα προβλήματα φόρτισης από-το-δίκτυο-στο-όχημα (G2V) και εκφόρτισης από-το-όχημα-στο-δίκτυο (V2G) .

Για την προώθηση της επεκτασιμότητας, η λύση μας παρέχεται στο μορφή αρθρωτών μικροϋπηρεσιών που διασυνδέονται χρησιμοποιώντας μια πλατφόρμα Διαδικτύου των Πράγματων (IoT) με πολλαπλά πρωτόκολλα. Από τη μία πλευρά, η σπονδυλωτότητα χαμηλού επιπέδου των υπηρεσιών Έξυπνου Ηλεκτρικού Δικτύου επιτρέπει την απρόσκοπτη ενσωμάτωση διαφορετικών στρατηγικών πρακτόρων, μηχανισμών τιμολόγησης και αλγορίθμων. Από την άλλη, η υλοποίηση που βασίζεται στο (IoT) προσφέρει αμφότερα άμεση εφαρμογή σε πραγματικές ρυθμίσεις, καθώς και προηγμένες δυνατότητες ανάλυσης, ενεργοποιώντας ψηφιακά δίδυμα μοντέλα για οικοσυστήματα Έξυπνου Ηλεκτρικού Δικτύου. Αυτές οι δυνατότητες παρέχονται από την πλατφόρμα IoT ΣΥΝΑΙΣΘΗΣΗ που χρησιμοποιείται για την παροχή της απαραίτητης υποδομής για τη διασύνδεση ετερογενών πρακτόρων, συσκευών και υπηρεσιών μέσω του δικτύου.

Περιγράφουμε την βασισμένη σε MAS/IoT αρχιτεκτονική μας και παρουσιάζουμε αποτελέσματα από προσομοιώσεις που ενσωματώνουν μεγάλους αριθμούς ετερογενών πρακτόρων Έξυπνου Ηλεκτρικού Δικτύου, οι οποίοι ενδέχεται να ακολουθούν διαφορετικές στρατηγικές για τις εργασίες λήψης αποφάσεων. Πιο συγκεκριμένα, οι προσομοιώσεις μας συγκρίνουν διαφορετικούς αλγορίθμους προγραμματισμού φόρτισης και μηχανισμούς τιμολόγησης που μπορούν να βρεθούν στην βιβλιογραφία. Επιπλέον, η δομή μας επιτρέπει τη δοκιμή διαφόρων σχημάτων σε λειτουργία προσομοίωσης και μπορεί επίσης να χρησιμοποιηθεί ως βάση για την υλοποίηση πρωτοτύπων πραγματικού κόσμου για την παροχή υπηρεσιών V2G/G2V μεγάλης κλίμακας.

Acknowledgements

First of all I would like to thank my supervisor Dr. Georgios Chalkiadakis for his guidance and support throughout this work and for giving me the opportunity to work on this very interesting topic.

I would also like to express my gratitude to Dr. Vasilios Samoladas and Dr. Nikolaos Spanoudakis of being members of the committee and for evaluating my thesis.

Moreover, I am grateful to Dr. Charilaos Akasiadis for his guidance and comments throughout this work, the moral support, and encouragement.

I would also like to thank my friends that supported me all these years during my studies in Chania and Iraklio.

Last but most important I would like to thank my family for supporting me in each step of the way, all these past years.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

Around the world, the impact of global warming is noticeable. Storms, floods and wildfires are increasing rapidly in frequency and intensity. The health of tens of millions of people is affected by air pollution and extreme weather events cause untold damage to homes and livelihoods too. The current global warming is caused by large-scale use of fossil fuels, which are burned to convert their stored chemical energy to thermal which can be easily managed [56], allowing people to operate power plants that generate electricity and heat. Also, energy-dense fuels such as gasoline or diesel fuel, liquids derived from fossil fuels are powering internal combustion engines (ICEs). ICEs are the main power supply for vehicles such as cars, planes and ships. However, the burning of such fuels generates by-products such us carbon dioxide (CO_2), carbon monoxide (CO), sulfur dioxide (SO_2), nitrogen oxides (NOx) and other chemical substances that can negatively affect human health and the environment [57].

For this reason, electric energy production is shifting to more sustainable sources by using resources that are renewable and more environmentally friendly [58]. The most recently adapted and quickly growing sources are wind and solar energy. As a result, coal and oil generated energy is decreased by percentage. Figure 1.1 shows the Change in power production policy of the world to shift away from fossil fuels to an energy mix dominated by low-carbon sources of energy. However, Renewable Energy Sources (RES) suffer from their intermittent nature that can lead to power grid instability. A solution to this problem is to store the additional energy in periods that there is a surplus in relation to consumption and consumed in times of shortage.

"Advanced" operations such as energy storage are enabled by the emerging Smart Grid [7, 11, 12, 62, 63], where energy and information flow towards all possible directions over distribution and transmission networks. As such, buildings but also vehicles can become active producers and consumers of energy, and need to be integrated into the Grid. Not only is the Smart Grid an electricity network with diverse producers and consumers, it is also a dynamic marketplace where heterogeneous devices appear and need to connect [8]. To date, several Smart Grid-related business models and information

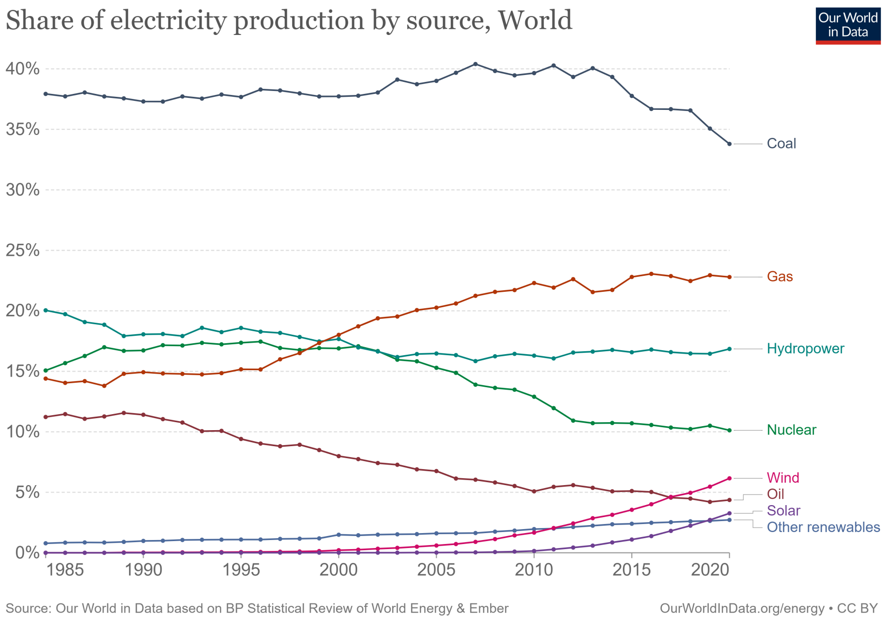


Figure 1.1: Share of electricity production by source [37].

systems' architectures have been proposed, but they do not adhere to particular standards [13]. This is normal, as the energy markets involved can be global, regional, or isolated; can be based mostly on renewable energy or not; and can be regulated by a public authority or allow dynamic pricing based on demand and offer.

Such energy markets naturally reflect systems where not one player can force others to use her products; players or stakeholders can come along their own business models; and stakeholders can have diverse goals in negotiating their consumption and offer. Moreover, these systems allow for pro-activeness of the players who pursue their goals and sociability—as they can form dynamic partnerships or coalitions, but also react and /or adapt to a changing dynamic environment. In addition, it is natural for participants to be generally able to freely join and leave the system at any time. All these characteristics point to agent technology and open *multiagent systems (MAS)* in particular [9, 14].

At the same time, the advances in the domain of the Internet of Things (IoT) allow the deployment of such approaches in the real world, as IoT offers a networking layer that interconnects distributed resources, e.g. power meters and other sensors, charging controllers and similar actuators, decision support agents and various processing services [20, 33]. A key IoT concept is that these resources, although heterogeneous, are interoperable in the sense that they can exchange information and are able to reconfigure particular parameters that are crucial for their operation [21].

Enabled by IoT, a Smart Grid digital twin can represent the running states of the multitude of physical devices that compose the power system and are interconnected, such as controllers, smart meters, etc. By collecting respective sensor measurements during frequent intervals, the delays associated with data gathering procedures of the past are overcome and, in this way, the actual per-device Grid state is made available to the operators in real-time [23, 24]. This monitoring capability can be further expanded with predictive maintenance techniques that provide the possibility of detecting malfunctions before these take place [30]. Also, in the opposite direction, having access to historical per-device measurements allows “interrogative” features, for post-hoc analysis or the training of machine learning models [23].

To the best of our knowledge, however, the existing approaches for the Smart Grid in general, and for the Vehicle-to-Grid (V2G)/ Grid-to-Vehicle (G2V) problem. Vehicle-to-Grid [64, 65, 67] is the process that electric vehicles push the stored energy back to the Grid, matching the time of generation to time of load. Grid-to-Vehicle is the process that the an electric vehicle connects to the Grid to charge the battery, so the energy flows from Grid to the Vehicle.

In particular, do not provide functional open prototypes that offer such features, or adequately exploit existing engineering MAS research paradigms. Now, in an open system, diverse agents representing stakeholders need to use predefined protocols to interact; but also need to work the protocols with their algorithms and goals.

Multi-agent systems are an excellent technological solution for smart grid applications such as EV charging management (V2G/G2V), as they enable the communication of complex information and the execution of difficult computations effectively and in a distributed fashion [61].

1.1 Thesis Contributions

Given this, the main contribution of this thesis is a novel MAS/IoT architecture we put forward for the V2G/G2V domain. The proposed approach has a number of merits and enables a variety of “advanced” capabilities:

- Our modular architecture allows the different stakeholders to reuse existing agents in new deployments, or to develop new ones, according to the individual needs and goals.
- Agents can connect to the system to offer their services "on the fly", and to con-

sume/produce electricity 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].

- Communication protocols are developed to provide connection between the agents.
- Our implementation goes one step closer to the creation of a digital twins for Smart Grids.
- The instantiation of our system uses 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 [22] among others.
- The validity of the approach is illustrated by simulation experiments with two different dynamic pricing mechanisms and three charging scheduling algorithms inspired by the existing literature.

A paper based on this diploma thesis work appears in Proceedings of 20th International Conference on Practical Applications of Agents and Multi-Agent Systems (PAAMS'22) [66].

1.2 Thesis Structure

The rest of the thesis is structured as follows. Chapter 2 describes the background knowledge about electric vehicles, the Smart Grid, the Internet of Things, and the IoT application enabler platform, as well as the MQTT protocol that is used for message exchange. Chapter 3 provides the related work that solves similar problems to ours. Chapter 4 presents a general description of agents in our architecture and the way that they are interconnected. Chapter 5 includes a detailed presentation of the communication protocols and the algorithmic behavior of each agent. Following that, in Chapter 6, we evaluate the applicability of our approach with realistic use case scenarios of interest. Finally, Chapter 7 concludes this thesis and outlines directions for future work.

We also include the Appendix A that provides information about each topic and the structure of each message and the Appendix B that explains the concepts and the attributes that used in our implementation.

Chapter 2

Background

This chapter is a brief review of the key concepts of our work. We provide information on the invention of electric vehicles, the operation of the Smart Grid, and how V2G/G2V services can be used to balance RES supply and demand. Then, we review the IoT technology that enables the ability to various devices to communicate with one another. Moreover, we provide an overview of the SYNAISTHISI IoT platform which allows to deploy services that can be easily copied, reconfigured, or modified according the needs of each stockholder. A critical component is also the MQTT application layer protocol [22], which is a standard for messaging, very popular on lightweight applications and the agents use it to communicate with each other.

2.1 Electric Vehicles

The Electric Vehicle (EV) is not a new concept. After the invention of the electric motor in 1828, scientists made some automation to create electric vehicles that powered it with battery cells. At first, batteries were not rechargeable, but after the invention of a lead-acid battery in 1859 and its improved version in 1881 [36] they were able to be used in practice. They even had advantages over gasoline cars, which in the early 1900s had increased levels of vibration, smell, and noise. However, their limited range and the massive production of cheaper ICE cars are two factors that made electric vehicles less appealing to consumers.

Given recent battery improvements and the raising concerns regarding GHG emissions, electric vehicles are starting to penetrate the market. Many well-known car manufacturing companies that produced ICE vehicles and even newly created start-ups are focusing their research on technologies that will increase the electric vehicle adoption. Some challenges that need to be addressed are high purchase and maintenance cost, range anxiety, the lack of charging infrastructure and power management [38].

In general, road transportation produces 11.9% of global GHG emissions. This means that powering the whole road transportation sector and transitioning to a fully decarbonized power mix, global emissions may be reduced by 6 billion tonnes of greenhouse gases each year [39]. Furthermore, the change of energy source from gasoline and diesel to electricity will have a huge impact with the additional demand on the current production and distribution system of electrical energy. We can decrease some of these effects by developing the Smart Grid and electric vehicle charging schedules that can charge when there is surplus of sustainable energy and stop charging or discharging when there is deficit.

2.2 Smart Grid

The term Smart Grid is used widely with various definitions and meanings. In the United States, the meaning of Smart Grid is broad, referring to the transformation of the electric industry from a centralized, producer-controlled network to one that is more interactive with consumers. In a lot of countries, Smart Grid refers to a physical network-based approach to ensure reliable, secure, more responsive and environmentally sustainable energy delivery and economics, as well as, the broad participation of the public in the renewable energy production. Lately, some governments have paid significant attention on improving the infrastructure to bring more social and economic benefits and, furthermore, to introduce a national market-driven demand management framework and system [55].

The Smart Grid is primarily an electrical Grid to which utilities and consumers are connected both electrically and over a communication network. The development of Smart Grids aims also to integrate renewable energy sources and electric vehicles. With the communication and cooperation of local consumers and producers, the current infrastructure aspires to efficiently manage the growing energy demand problem and minimize the social impact of replacing traditional energy sources with renewable ones.

Moreover, the installation of smart metering devices will enable more frequent consumption measurements that can help people reduce their energy bills. Inside a smart home, many smart sensors and devices are connected to the Internet, can be controlled remotely, or can be programmed to operate under specific scenarios. For example,a smart agent controlling a household could delay the use of energy-intensive appliances, e.g. a washing machine when there is a shortage in supply. This provides a reduction in peak energy demand and the consequent increased electricity costs.

On the other hand, the use of renewable energy generation makes the energy mix produce fewer GHG emissions. Regrettably, a disadvantage of using solar and wind power is its volatility and dependence on the weather. The integration of Smart Grid restricts this problem by informing the consumers about the energy capabilities of the Grid and allowing large-scale production and consumption coordination [54, 68].

With Smart Grid technology and consumer participation, utilities can more easily meet growing demand for power for electric vehicles and satisfy charging needs with Smart Grid technology. By adding more plug-in electric vehicles to the Grid we have the potential to reduce fuel costs, lower our dependency on fossil fuels and help reduce greenhouse gas emissions [59].

Meanwhile, the connection of electric vehicles will affect the Smart Grid. The insertion of a large number of electric vehicles into the Grid will increase the imbalance between supply and demand in case they are charging their batteries at peak hours. However, if the electric vehicles charge at hours that there is a surplus of RES production, they will help reducing the instability of the Grid. They can even support the Smart Grid by discharging their batteries and providing their stored energy back to the Grid, a procedure known as Vehicle-to-Grid (V2G). This power exchange is beneficial to the Grid and can also be proven profitable for the car owners depending on the purchase and sale price of energy, always taking into account the costs for battery degradation [40].

2.3 Internet Of Things

The Internet has made it possible for the entire world to interact and connect in real time. The "Internet" we have known since the 1990s has been a network based on the exchange of data between computers. People around the world share information, experience, knowledge, and technology over the Internet. In fact, technologies in various industrial sectors have made incredible breakthroughs in development over the last few decades.

However, the expansion of the network creates more opportunities than mere human-to-human communication. There are many other beings apart from humans like animals, plants, and even unliving objects that can connect to the Internet. When all those entities can "communicate" with humans, we can make better decisions and move towards a world of optimal convenience and efficiency. This idea made the concept of the Internet of Things (IoT) possible [50].

The term Internet of Things was first introduced by Kevin Ashton [43] in 1999, but there already existed a similar application since 1982 with a modified Coca-Cola vending machine at Carnegie Mellon University becoming the first ARPANET-connected appliance [42]. According to McKinsey, the Internet of Things can be defined as : "Sensors and actuators embedded in physical objects are linked through wired and wireless networks, often using the same Internet Protocol (IP) that connects the Internet." [44]

Currently, IoT comprises a large ecosystem. The market is flooded with smart products that connect to the Internet, can be controlled remotely, and inform the users for their status. In a smart home one can find appliances, monitoring, and automation devices. Consumers can easily track their heart rate statistics using a smart watch paired with a smart phone or can watch television series on the subscription streaming service using a smart television. There is also a large adoption of IoT in the industry with heavy machinery, agriculture and factories [45] and in modern cities, applied in public services, transportation, public safety, sustainability, and infrastructure to serve the citizens [46].

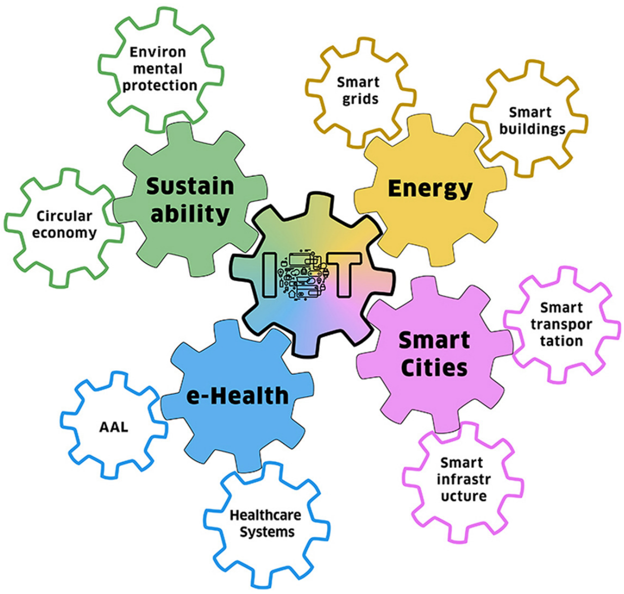


Figure 2.1: Application domains of IoT technology [47]

2.4 The SYNAISTHISI Platform

The underlying IoT platform is a critical component of any IoT deployment, integrating hardware, connectivity, software, and application layers to manage and configure devices, collect and analyze data, activate applications, and connect to the cloud or on-site server [51].

SYNAISTHISI IoT is a platform that uses open source middleware frameworks and integrates them into single, operable, dockerized instances. The SYNAISTHISI platform is mainly composed of the following:

- a Message-Oriented Middleware (MOM), which is a central message broker running on the cloud and is accessible from all internet–enabled devices. Its goal is to support inter– and intra–machine communication.
- a REST web server, whose role is to provide a control layer over the available resources.
- a Resource piping mechanism, which allows for rapid development and deployment of custom applications [48].

It supports and cross-translates multiple Application Layer Protocols (ALPs), i.e., Message Queuing Telemetry Transport (MQTT), Representational State Transfer (REST) / HyperText Transfer Protocol (HTTP), Websockets, Constrained Application Protocol (CoAP), and Advanced Message Queuing Protocol (AMQP), and allows for semantic descriptions, user access control, and IoT resource management [49]. There are two most common architecture approaches, request / response or publish / subscribe, that the available protocols often follow. Each approach has different advantages and drawbacks; in our work we choose to use MQTT, that is publish/subscribe due to the advantages that it offers in supporting mobility by means of asynchronous, non blocking, and one-to-many message distribution semantics for event notification [52]. More details can be found in Section 2.5.

The platform provides authentication and authorization to restrict access to topics that hold private information and allows users with the necessary privileges to publish or subscribe to the corresponding topic. SYNAISTHISI is a suitable solution for implementing an open architecture system, since it enables the large-scale reusability of services for fast system deployment.

Moreover, it provides users with the ability to start or stop their services. They can run the source code files they have uploaded for that service on the platform's infrastructure. When the service is deployed (for example, if the source code file is running as an operating system process or in another Docker container), subscribe to all IoT service input broker topics and follow the implemented instructions for these [49].

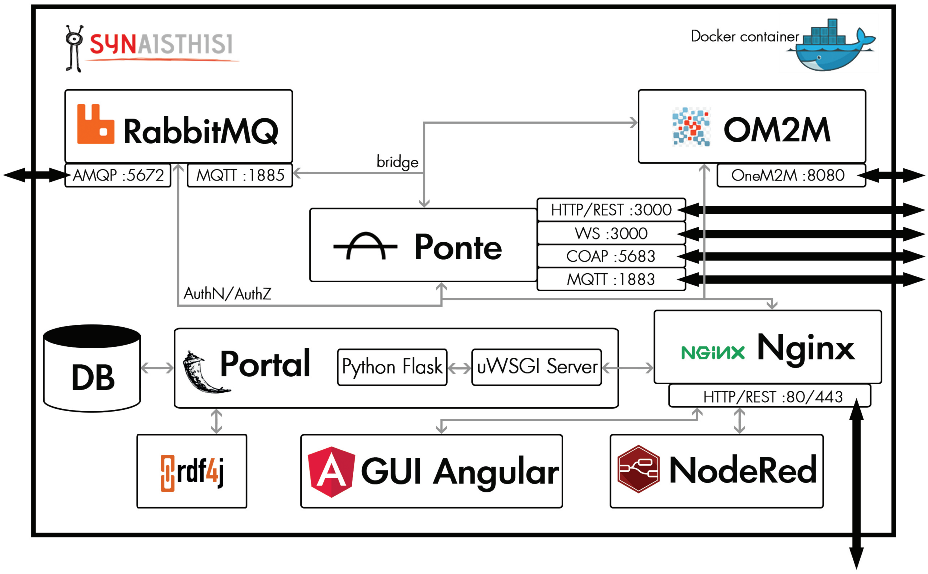


Figure 2.2: SYNAISTHISI Platform components [49]

2.5 MQTT

MQTT[1] is an Organization for the Advancement of Structured Information Standards (OASIS) standard messaging protocol that was released by IBM in 1999. It is designed as an extremely lightweight publish/subscribe messaging transport that is ideal for sending data accurately under the long network delay and low bandwidth network condition. In the publish/subscribe messaging model participating users subscribe and publish to topics in order to exchange messages. The subscriber subscribes to topics which must be informed and by that receive every message that is published to those topics. On the other hand, publishers can publish messages to topics in a way that allows all subscribers to access messages from those topics [41].

[1] mqtt.org

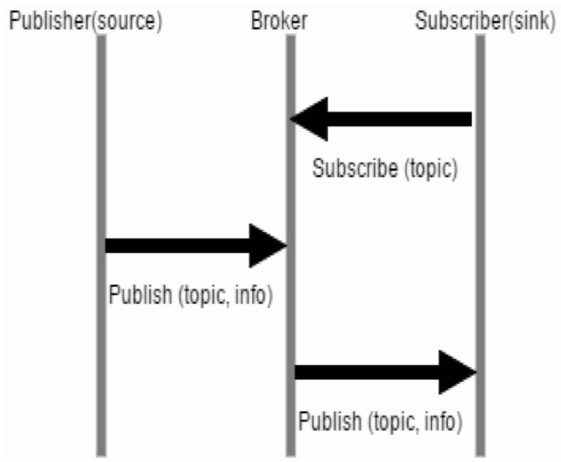


Figure 2.3: Publishers, subscribers and broker connection using MQTT [53]

That operation is achieved with the use of a broker that stands between publishers and subscribers and forwards the new messages to the subscribers that are related to the topic that is published. The Figure 2.3 shows the connection of publishers, subscribers and the broker. The subscriber starts with a *Subscribe(topic)* message to the Broker that action informs the Broker for which topic the Subscriber is interested to get information. The Publisher sends a *Publish(topic,info)* to the Broker with the new information about the topic. Then, the Broker forwards the new message *Publish(topic,info)* to any Subscriber that has subscribed to the topic.

The topic in MQTT is a UTF-8 string that the broker uses to filter messages for each connected client. The topic consists of one or more topic levels, and each topic level is separated by a forward slash (topic level separator), for example, *CS/CS100/RegisterChargingStation*.

As long as a client subscribes to a topic, a client can subscribe to the exact topic of the published message or to multiple topics at the same time using wildcards. The wildcard can only be used for topic subscription, not for message publication. There are two different kinds of wildcards the single-level denoted with (+) and multi-level denoted with (#). Any topic matches a topic with single-level wildcard if it contains an arbitrary string instead of the wildcard. For example, if a client subscribes to the topic *CS/+/RegisterChargingStation* where the second level contains the id of the charging station, it will receive the registration messages from every charging station. The multi-level wildcard covers many topic levels. The hash symbol represents the multi-level wild card in the topic. To determine which topics match the broker will need to place a multi-level wildcard as the last character of the topic and precede it by a forward slash. If a client subscribes to a topic with a multilevel wildcard, the client receives all messages from a topic that begins with a pattern before a wildcard, no matter how long or deep the topic is. If the charging station with ID 100 wants to subscribe to all topics related to it, it can subscribe to *CS/CS100/#* and will receive all messages that start with that pattern.

The MQTT default port is TCP/IP port 1883. MQTT has different implementations, and libraries such as mosquito, hivemq, and paho MQTT are some of the most widely used and well documented. MQTT supports Transport Layer Security (TLS)/Secure Sockets Layer (SSL) security communication protocols over port 8883. TLS / SSL provide communication security through the computer network that is used in different applications such as email, web browsing, Internet faxing, voice over IP (VoIP) and instant messaging [53].

Chapter 3

Related Work

Many simulation tools and real-world prototypes have been developed to date that incorporate a wide range of features related to the V2G/G2V domain. These systems are either integrated environments (for simulation or real-world application) that incorporate many entities that are related to the V2G/G2V domain to some extent, or they focus on underlying difficulties related to EV operation in the smart grid.

EVLibSim is a Java tool for the simulation of EV charging station operation [2]. Building on their previous work [1], the authors construct a tool that provides a friendly and comprehensive User Interface (UI) for the management of charging stations. Using this tool, charging stations can be easily created, modified, and monitored with respect to the execution of specific scheduling algorithms. However, proven quite useful by test usage scenarios involving actual domain experts, this approach is restricted to charging stations only and does not incorporate additional stakeholder types.

In their work, Jordán et al. [4] propose a MAS to support the decision-making process on the determination of locations of EV charging stations in the city of Valencia. The system integrates various information from heterogeneous data sources, such as traffic, social networks, charging station pricing, and optimizes charging station locations using a genetic algorithm. This approach is handy for designing charging infrastructure; however, it does not capture "what happens next", once it is offered for use.

Kamboj et al. [10] present a method for forming coalitions of EVs to provide V2G and demand-side management (DSM) services to the electricity grid. The authors incorporate a MAS architecture and implement simulations using JADE. The system considers an intelligent agent for each EV, an aggregator agent responsible for forming coalitions of EVs, and a TSO agent that communicates with aggregators and regulates the V2G/DSM process. EVs are selected so that the minimum energy requirement to participate successfully in the regulatory market is met. The evaluation, however, involved only five electric vehicles. Moreover, the system does not allow for complex EV selection processes and, thus, it cannot scale.

Papadopoulos et al. [5] propose a MAS implemented in JADE that coordinates the battery charging of EVs considering the individual preferences of the EV drivers, by using specific search techniques and neural networks. Driver preferences include willingness to participate in V2G, as well as the charging availability of the vehicle. The proposed MAS is shown experimentally to have the ability to satisfy autonomous EV owners' charging preferences, both under normal and emergency grid conditions.

Some simulators with (potentially smaller-scale) real-world trials are proposed in the literature for the delivery of V2G and G2V services. XBOS-V [29], for example, is a system for managing plug-in EV charging at homes and small businesses settings.

RISE-V2G (github.com/SwitchEV/RISE-V2G) implements a V2G communication interface ISO 15118, which is a standardized communication method, that provides lower level connection infrastructure between electric vehicles and charging stations. Similar examples are the Open Charge Point Protocol (OCPP), the Open Charge Point Interface (OCPI), and the Open Smart Charging Protocol (OSCP) [32].

OpenV2G [31] is an implementation of the necessary components of the V2G public key infrastructure. The focus of the approach is to securely connect electric vehicles and charging stations. It provides simulation capabilities for being able to connect EVs and CSs.

The Grid-integrated Electric Mobility (GEM) model [18], another approach, simulates the operation in both the mobility and the electricity domains. However, the simulation approach followed represents a higher level and does not include particular stakeholder types, such as a station recommender.

ACN-Sim [28] is a tool for managing battery charging. Its implementations focus on optimize the line current using different transformers and by changing the EV charging schedule accordingly. So, they are focus on a more electrical direction rather than a grid-level.

Spanoudakis et al. [65] implemented an open multi-agent systems services architecture for V2G/G2V problem. Elements of our architecture are based on their work.

Chapter 4

System Architecture

In this chapter, we present the general architecture of the system. The various stakeholders that participate in our design are modeled as intelligent agents. Those agents can read values from sensors, request data from database, make decisions that serve their needs. They can even communicate with other agents, make deals, and negotiate existing agreements.

4.1 Agents

We assume that agents co-exist in a microgrid infrastructure that can be interconnected with other parts of the Smart Grid through distribution and transmission networks. When a microgrid requires power that can not be generated locally, it can import it, while, when it has a local energy surplus, it can export it to the (broader) Grid and create additional profits for its electricity producers, according to energy market regulations [8]. Figure 4.1 provides an overview of the agents in our architecture and their interactions.

The agents are connected with the use of the SYNAISTHISI IoT Platform where each agent is represented with a service. The owner or an user with the appropriate permissions of a service can duplicate, edit, delete, start, or stop it using the platform. The agents uses the Communication Protocols, sets of rules that orchestrate the agents to publish message to MQTT topics, to contact with the others and we will see in more detail in Section 5.1.

In particular, the agent types in our system are: the *(a)* Electric Vehicle agents (EV), the *(b)* Charging Station agents (CS), the *(c)* Electricity Producer agents (EP), and the *(d)* Electricity Consumer agents (EC). We also assume the existence of a regulatory service, or possibly a private for-profit service, consisting of *(i)* a Station Recommender agent (SR), *(ii)* an Electricity Imbalance agent (EI), and *(iii)* a Mechanism Design agent (MD). In what follows, we refer to this service by its three distinct agents separately, since our focus is on the technical details of the functionality provided by each and not on the business and regulatory aspects involved. Note that each agent type may consist of certain "private" sub-modules, whose specific functionality can further differentiate agent

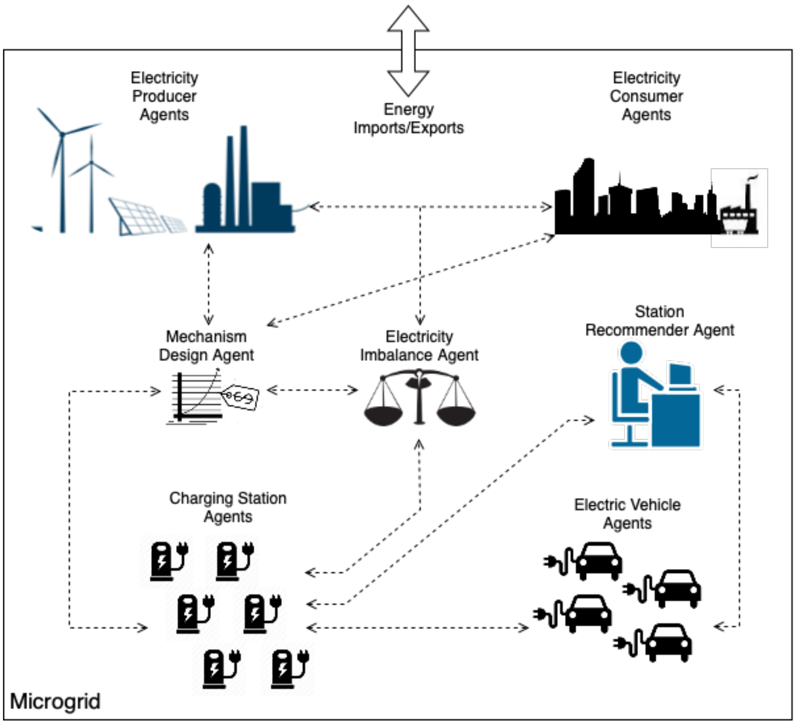


Figure 4.1: Overview of the proposed MAS architecture for the V2G/G2V problem.

behaviors.

4.1.1 Electric Vehicle Agent

Each Electric Vehicle must have an intelligent agent that is installed and running on the computing infrastructure that is part of the electric vehicle, or on mobile devices owned by its drivers. The goal of the Electric Vehicle Agent is to ensure that the vehicle is always enough charged for the driver's next trip. It observers who its owner acts and behaves, collects data, models and predict a strategy that meet its needs. This strategy contains actions to charge the battery if it is insufficient or the energy price is low. Electric Vehicle agent can provide its stored energy to the grid by enabling V2G energy transfer, when the price is high to make a profit and help the stability of the grid. However, it is probable that the charging schedule is such that the agreement that was made between the Electric Vehicle agent and the Charging Station does not fully satisfy the needs of the owner. In this situation, he can initiate a negotiation that can change the latter reservation or cancel it and make a new reservation.

An Electric Vehicle agent contains a Preference Elicitation module that is responsible to monitor the driving habits and behavior of the driver, and predict its future charging preferences; a Financial Management module that is responsible for making the payments regarding the charging of the Electric Vehicle and receiving payments when the Electric Vehicle participates in V2G activities; a Negotiation Decision Making module that contains the algorithmic components and the logic according to which the agent conducts negotiations; and a Recommendations Selection module that contains the algorithmic components for the agent to determine which charging recommendation to select. An Electric Vehicle agent communicates with the Station Recommender agent and the Charging Station agents (see below).

4.1.2 Charging Station Agent

Charging Station Agent provides the physical gateways (i.e., connectors, parking slots) to electrical vehicle that connect to the grid. The Charging Station agent communicates with the Electric Vehicle, Station Recommender, Mechanism Design, and Electricity Imbalance agents.

The connection enables the Electric Vehicle agents to charge their batteries when the price of energy is low and discharging them to support the stability of the grid when the price is higher. The Charging Station agent gets commission proportional to the energy unit it exchanges; for example, it can charge a few cents per kWh additional to the cost from the grid. To maximize the profit, the Charging Station must maximize the energy that is transferred between Electric Vehicles and grid. So without expanding the parking area, adding or upgrading the charging connectors the agent can manage the charging schedules of the cars that are reserving slots. The connectors with the highest charging rate must be idle as least time as possible because it can service more Electric Vehicles in less time and generate more profit. In case of an emergency situation (i.e., damaged connector) or when it can generate additional profit the agent can negotiate with the Electric Vehicle owner to change the parameter of charging that they agreed earlier. Some parameters that can change are the time of arrival or departure, the parking slot, the charging rate, or the price.

Each time the Charging Station agent receive a reservation must message from an Electric Vehicle it must verify that the recommender create that recommendation and Electric Vehicle did not change it. Only in case of the recommendation is genuine from the Station Recommender agent the Charging Station can accept it and make the reservation. Charging Station agent, also, contacts regularly with Mechanism Design and Electricity Imbalance agents to report the updated energy production, consumption, and being informed about the price of the energy and the amount of imbalance.

4.1.3 Station Recommender Agent

The Station Recommender Agent's goal is to propose charging slots inside charging stations that fully match the Electric Vehicle's preferences. It communicates with the agents of the Electric Vehicle and Charging Station agents.

When a Electric Vehicle needs to charge, it sends a request to Station Recommender agent with its location and requirements about the charging session (i.e arrival and departure time, the state of charge of the Electric Vehicle's battery and the charging rate that prefers to charge). The Station Recommender agent, then, checks the available Charging Stations, finds the Charging Stations that are closer to Electric Vehicle location and match most of the Electric Vehicle's preferences. It then sends the list with the best recommendations to the interested Electric Vehicle that can freely choose between the recommendations according to its strategy (i.e lowest price, lowest distance).

The connection of Station Recommender and Charging Station agents is obviously necessary. After a Charging Station makes a reservation to an Electric Vehicle, it must inform the Station Recommender agent about the charging session. The agent needs to know the arrival and departure time as well as the charging slot that the Electric Vehicle will use. That action is needed to prevent the Station Recommender agent from making the same recommendation multiple times. However in case of conflict on charging slot and time, the Charging Station agent has the ability to negotiate with Electric Vehicles to find a free charger or to reject the reservation if it is full, and the Electric Vehicle will be addressed to another Charging Station.

The Station Recommender interacts with Charging Stations in a different way. Each Charging Station needs to know if the recommendation that the Electric Vehicle sent is generated from the Station Recommender. The Charging Station forwards the received recommendation to Station Recommender to verify that it is genuine and only then the Charging Station agent can accept the recommendation and make a reservation for the Electric Vehicle. This procedure adds an extra layer of security to the system and guarantees that only the Station Recommender agent can create recommendations and the other agents cannot modify them.

4.1.4 Electricity Imbalance Agent

The Electricity Imbalance agent is responsible for calculating and predicting the periods of electricity shortage and surplus in the grid. The Electricity Imbalance agent is connected with the Electricity Producer, Electricity Consumer, Charging Station, and Mechanism Design Agents.

The Electricity Producer, Electricity Consumer and Charging Station inform the Electricity Imbalance agent about their energy exchange profile periodically, for example every day, or after a big change on the expected energy profile, if there is a change on weather forecast or some Electric Vehicles need to charge in a specific Charging Station.

The Electricity Imbalance publishes the imbalance of electric power for each timestep (i.e., every hour) to Mechanism Design and the Charging Station agent. The Mechanism Design agent needs this information to calculate the price of electric energy that will be lower in periods of surplus and higher when there is shortage. If the Charging Station agent knows the imbalance can provide more support to the Grid by making a finer grain management when each Electric Vehicle is charging and modifying the charging schedule of the Electric Vehicles that can charge in a non-peak period.

4.1.5 Mechanism Design Agent

The Mechanism Design Agent is an intermediate trusted third party entity which is responsible for the electricity prices calculation and the payment flows. The Mechanism Design agent communicates with Electricity Producer, Electricity Consumer, Charging Station and Electricity Imbalance Agents.

The calculation of the prices needs a good estimation of energy surplus and shortage, that is provided by the Electricity Imbalance Agent. When the production is greater than the consumption, we consider that there is a surplus of energy. Energy must be used in that period, otherwise it will be wasted. So, the Mechanism Design reduces the price to encourage the consumers to use the additional energy. On the other hand, when the production is less than the consumption, the grid is in a shortage. The Mechanism Design agent increase the price of the electricity to make the consumer to change their plan and consume in a non-peak period or encourage agents that can store energy to sell it back to the Microgrid (i.e the Electric Vehicle agent using V2G).

We consider that the Microgrid can always import or export energy to the main Grid in case of emergency, but usually it is not efficient because the buying price is greater than the selling price.

The spread between the selling and the buying prices can be observed in a lot of markets (e.g. stoke exchanges) and there is also inside the Microgrid. The nature of the electric energy (expensive storage, losses on transmission, inefficiencies on transformation) obligates it to be a good that is more expensive to buy than to sell. The spread of those prices is calculated from the Mechanism Design agent and corresponds to the demand and supply in that period. It is worth mentioning, that the buying and selling prices are proportional values. When one is increasing so and the other and vice versa.

4.1.6 Electricity Producer Agent

The Electricity Producer agent is typical energy producer that uses mainly renewable sources of energy to generate electricity. The Electricity Producer communicates with the Electricity Imbalance and the Mechanism Design agents.

We expect that the most producer will use solar and wind energy and fewer will invest in large scale geothermal or hydroelectric. The photovoltaic panels and the wind turbine farms have the disadvantage that there closely correlated with the weather in the area. Therefore, the Electricity Producer agent must be informed frequently about a reliable weather forecast and publish the expected amount of energy production to the Electricity Imbalance agent. This is very crucial for the stability of the Microgrid because if the Electricity Imbalance agent expected to have a large amount of solar energy during a sunny day, but that day there are a lot of clouds the price forecast will be inaccurate and the consumers will ask for more energy than what it will finally produce. So producers that depend heavily on the weather conditions must update their energy supply profile every time it deviates from their last.

The Electricity Producer agent also informs the Mechanism Design agent so it can calculate the payment for each agent that provided energy to the Microgrid. This procedure is well tested with the use of an energy metering device that records the exact amount of electricity is given to the Grid. Then, the Electricity Producer agent sends it to the Mechanism Design agent that manages it and calculates the amount of money is needed to pay to the producer.

4.1.7 Electricity Consumer Agent

The Electricity Consumer agent is a typical consumer of electric energy. Some categories of Electricity Consumer are households, industries and other buildings and infrastructure. The Electricity Consumer agent communicates with the Electricity Imbalance and the Mechanism Design agents.

The use of devices that can connect to the Internet is growing rapidly and brings us closer to a situation in which the owner of each device can schedule when it will operate. Therefore, consumers can take advantage of the knowledge of the electric energy price and use some energy consuming devices (e.g. washing machine, water boiler) in periods when the price is low.

The Electricity Consumer agent is responsible to report the amount of energy is expected to consume each timestep (i.e., every hour) for a specified time period (i.e., a day). It must also provide a level of confidence that corresponds to how sure it is about its expected consumption. The agent meters the amount of energy it is actually consumed, informs the Mechanism Design for it and then the Mechanism Design agent calculate the amount of money is needed to pay for the consumed energy.

Chapter 5

System Implementation

In this chapter, we review information on the implementation of the work. The Communication Protocols (i.e. inter-agent control) that describe the way that the agent is connected and the Agents Behavior (i.e. intra-agent control) that explain the internal operation of each agent are presented here in more detail. Also, we present the different algorithms we implement, and the Mechanism Design and Charging Station agents uses in their strategies.

For the implementation of agents' microservices we used the SYNAISTHISI IoT Platform. It allows agent to communicate using MQTT publish/ subscribe protocol that is a widespread connection method for IoT devices. SYNAISTHISI provides a friendly and comprehensive Python API for agent development and management. The services and sub-modules made in the platform come in docker containers, allowing scalable and operating system independent deployments.

We followed a methodological approach to system analysis and design, based on the Agent Systems Engineering Methodology (ASEME) [70], which has been employed in the past for modeling Ambient Intelligence applications [6] and referred to by the literature on modeling IoT-based MAS [34, 35]. It is common the use of engineer multi agent systems [69] in the design of statecharts. ASEME builds on existing languages, such as Unified Modeling Language (UML) and state charts, in order to represent 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 first defines the agent's behavior by coordinating the different modules that implement its capabilities, while the latter defines the protocols that govern the coordination of the society of the agents.

The inter-agent and intra-agent control is illustrated using statechart as described in Unified Modeling Language (UML)[2.5 version]. The statechart are designed using the diagramming and vector graphics application Microsoft Visio.

5.1 Communication Protocols

The Communication Protocol (CP) is set of standard actions that agents use to communicate with each other.

In Figure 5.1 we show the connection of each agent and the protocols that uses to pass the necessary information through the MQTT topics.

For each protocol we present a state diagram that illustrates the agents that participate, their actions, the messages they send and the topic that sent over.

The protocol is defined as a state chart (following the semantics of Harel [16] and the graphical model syntax of the ASEME statechart editor [17, 6]) with the AND-state *CP1_ChargingRecommendation* as the root. AND-states contain OR states, and being in an AND-state entails being in all its OR-states simultaneously.

The figures of the communication protocol are split into two parts. In the left part there is the initiator of the protocol and in the right the responder. START-states show the beginning of the execution (circle). Basic states (shown with green color) are where agent activities are executed and END-states show where execution ends (black dots within a circle). Transitions from one state to the other occur *(i)* when the activity of the source state finishes and there is no event on the arrow, or *(ii)* when the event on the arrow takes place.

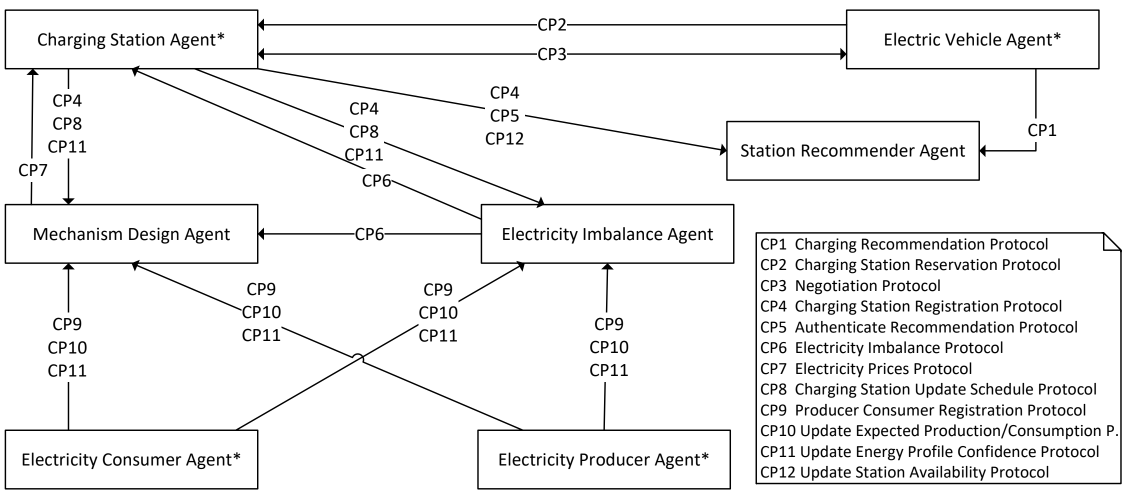


Figure 5.1: Overview of the proposed architecture depicting also the communication protocols used. Stars (*) denote agent types with multiple instances. Arrows start from the agent that initiates the protocol and point to the receiver agents.

Next, for each communication protocol, we provide a detailed description of the operation and a figure to show it graphically.

5.1.1 Charging Recommendation Protocol (CP1)

In CP1 the Electric Vehicle starts the protocol and enters the state *SendRecommendationRequest*. Electric Vehicle publishes a message on topic *"EV/+/RequestChargingRecommendations"* that contains an array with *preferences* and *location*. The plus sign (+) in the topic is replaced with the ID of Electric Vehicle that sends the message and it is used like MQTT single-level wildcard defines. Station Recommender receives the message and enters the *ReceiveRecommendationRequest* state where it confirms that the request is valid and finds the ID of the Electric Vehicle. After that, the Station Recommender enters the *CalculateChargingRecommendations* state that according to the Electric Vehicle's preferences, the location and the availability of the Charging Station that is informed finds the most suitable electrical chargers. Then, Station Recommender enters the *SendChargingRecommendations* state that sends a message on topic *"EV/+/ChargingRecommendations"* that contains a list with the most matched Charging Slots that are available. Electric Vehicle receives the message, enters the *ReceiveRecommendations* state, and finishes the protocol.

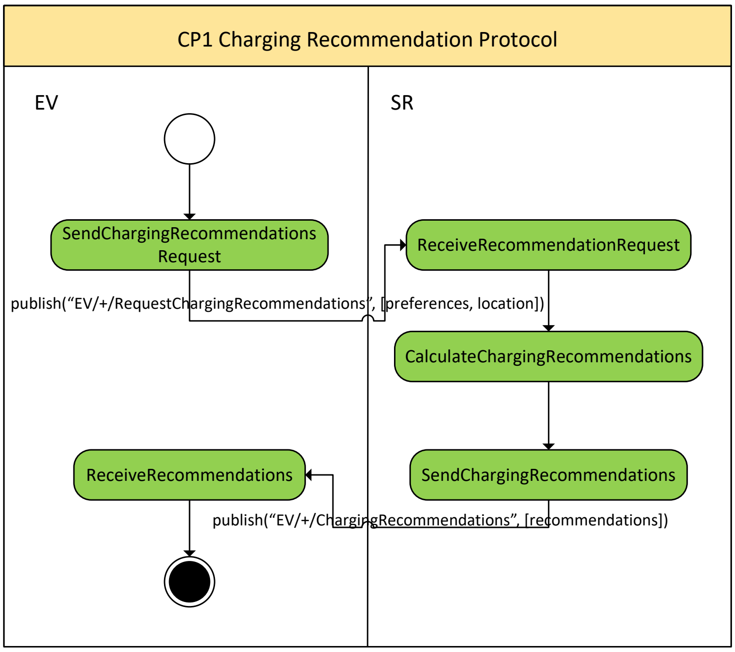


Figure 5.2: The model of the Charging Recommendation Protocol (CP1)

5.1.2 Charging Station Reservation Protocol (CP2)

After the Electric Vehicle agent completes the Charging Recommendation Protocol (CP1), a frequent sequence is the selection of a recommendation and the interaction with the corresponding Charging Station agent using the Charging Station Reservation Protocol (CP2).

The Electric Vehicle agent begins the protocol and enters the *SendStationReservationRequest* state where it publishes on the topic *"CS/+/ReserveChargingSlot"* (the plus sign is replaced with the ID of the addressed Charging Station) and a message that contains the chosen recommendation, the battery status, and information about its preferences. The Charging Station agent receives the message and enters the *ReceiveChargingReservationRequest*. Then it enters the *HandleChargingReservationRequest* state, where the agent checks if the recommendation is genuine by using CP5, creates the reservation and calculates the charging schedule according to the selected algorithm from 5.3.2. It then transits to the *SendReservationOutcome* and publishes on topic *"EV/+/ReservationOutcome"*, replacing "+" with the Electric Vehicle's ID, the generated reservation, the schedule that represents the amount of energy the Electric Vehicle will charge or discharge each timestep and the price of buying and selling an amount of energy (i.e 1 kWh) each timestep. The Electric Vehicle receives the outcome message, enters the *ReceiveReservationOutcome* and terminates the protocol by entering the final state.

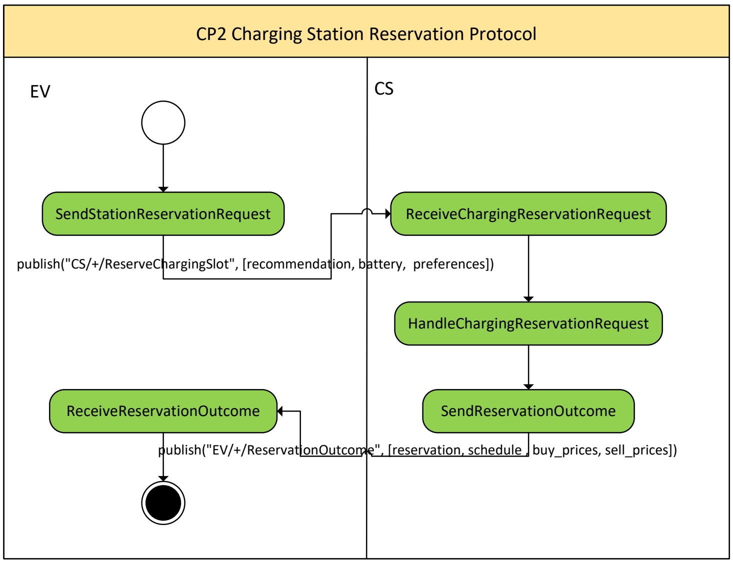


Figure 5.3: The model of the Charging Station Reservation Protocol (CP2)

5.1.3 Negotiation Protocol (CP3)

The Negotiation Protocol (CP3) is the communication protocol that enables the negotiation between the Electric Vehicle and Charging Station agents about a reservation that has already been agreed. There are two roles in the protocol, the initiator and the responder. This protocol is symmetrical that means that either the Electric Vehicle can initiate it and the Charging Station will be the responder or vice versa. The initiator starts the protocol, enters the *SendNegotiationMessage* state, and sends the message with *NegotiationObject*, which can be the arrival time, on the topic *"CS/+/EV/+/EV_start/start_negotiation"* if the Electric Vehicle is the initiator or on *"CS/+/EV/+/CS_start/start_negotiation"* otherwise. The first plus sign is replaced with the ID of the Charging Station agent and the second with the ID of the Electric Vehicle agent. The responder receives the message that is published and enters the *ReceiveNegotiationMessage* state. Following is the *NegotiationDecisionMaking* state where the decision process takes place and the responder decides if it accepts to modify its reservation according to the provided negotiation object. Then it enters the state *SendDecisionMakingOutcome* and publishes on the topic *"CS/+/EV/+/EV_start/accept_negotiation"* or on *"CS/+/EV/+/CS_start/accept_negotiation"* the *Decision* that can be *"Accept"* or *"Reject"*. The initiator takes the outcome message, enters *ReceiveDecisionMakingOutcome* where it processes the decision, and goes to the final state that terminates the protocol.

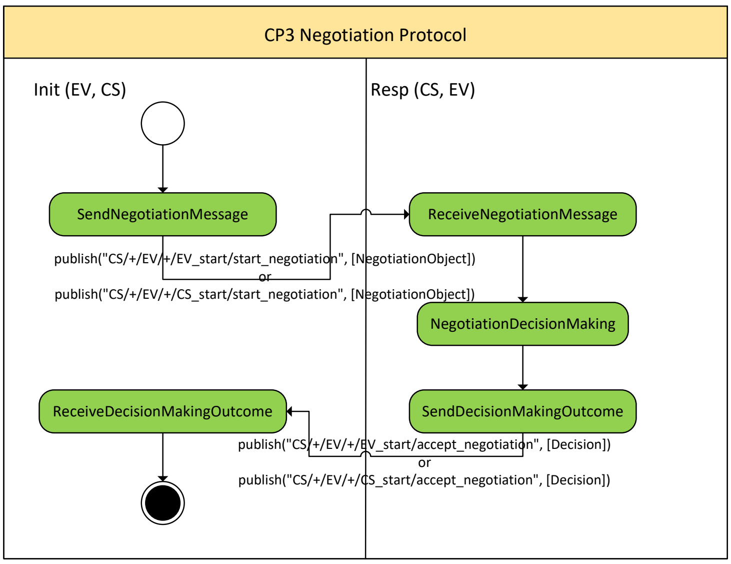


Figure 5.4: The model of the Negotiation Protocol (CP3)

5.1.4 Charging Station Registration Protocol (CP4)

In the Communication Protocol that manages the registration of Charging Station, there is a connection between the Charging Station and service providers that includes the Electricity Imbalancer, the Mechanism Design, and the Station Recommender agents. The Charging Station agent starts the protocol and enters the *SendChargingStationRegistration* state. In this state, information is collected on the location, the number, and type of chargers, and other constant characteristics of the station. This information is contained in message *CSinfo* and is published on topic *"CS/+/RegisterChargingStation"*. The service providers that subscribe to that topic receive the message and enter in the *ReceiveChargingStationRequest* state. Then follows the *HandleChargingStationRegistration* state where the transaction takes place and the processing and storage of the information and the registration of the charging station. The data process indicates whether it was successful or failed if there is missing information or if the station is already registered. It enters the state *SendStationRegistrationOutcome* and sends the *Outcome* on the topic that each service provider has. For example, the Electricity Imbalance publishes to the topic *"EI/+/RegistrationOutcome"* and the Mechanism Design and the Station Recommender replace the *EI* with *MD* and *SR* respectively. The Charging Station receives the outcome message, enters the *ReceiveRegistrationOutcome* and terminates the protocol by entering the final state.

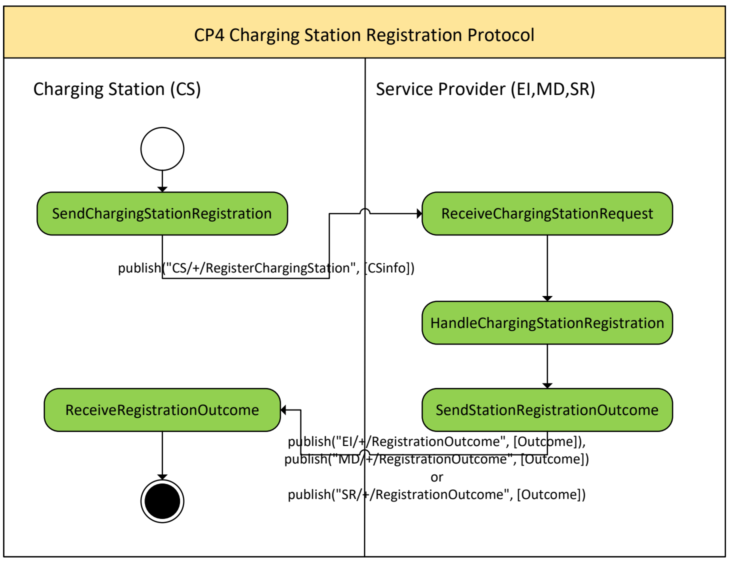


Figure 5.5: The model of the Charging Station Registration Protocol (CP4)

5.1.5 Authenticate Recommendation Protocol (CP5)

As we noted in subsection 5.1.2, the successful completion of a reservation requires authenticating recommendation. The Authenticate Recommendation Protocol (CP5) ensures that the Charging Station agent that the recommendation it received from the Electric Vehicle agent is created from the Station Recommender agent and it not generated or modified from any malicious user.

The Charging Station agent begins the protocol, enters the *SendAuthenticateRecommendationQuery* state and sends the *recommendation* on topic *"CS/+/AuthenticateRecommendation"*, where the plus sign will be replaced with the Charging Station agent's ID. The Station Recommender receive the message, enters the *ReceiveRecommendationAuthenticationQuery* state and checks in the recommendation history data if there is the received recommendation. Then, enter *SendAuthenticationOutcome* and send on *"CS/+/AuthenticateRecommendationOutcome"* the *outcome*, which will be *"True"* if the recommendation found in the history and *"False"* otherwise. The Charging Station receives the message, enters the *ReceiveAuthenticationResponse* state, and finishes the protocol.

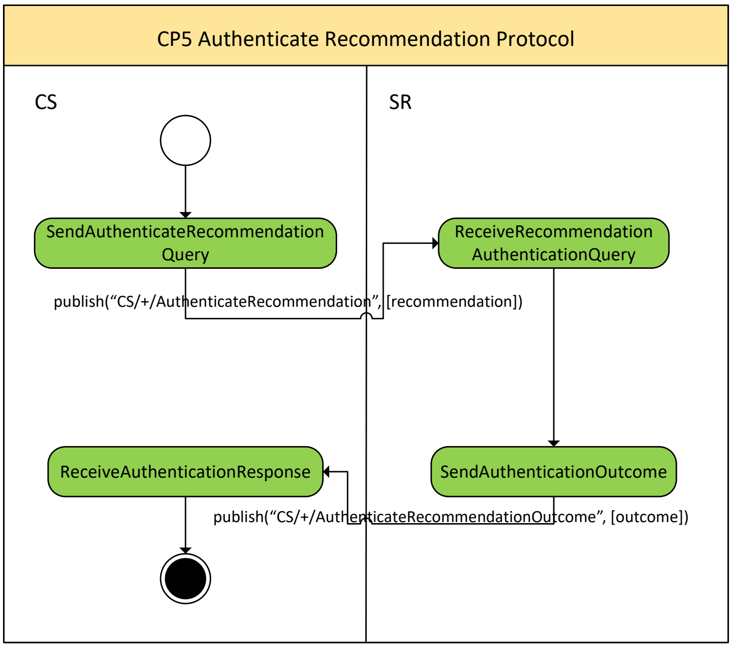


Figure 5.6: The model of the Authenticate Recommendation Protocol (CP5)

5.1.6 Electricity Imbalance Protocol (CP6)

This protocol connects the Mechanism Design, the Station Recommender, and the Charging Station agents, which we will call Service Users, with the Electricity Imbalance agent to be informed about the imbalance of energy (i.e., the shortage or surplus) during each timestep for a day.

The Service User initiates the protocol and enters the *SendElectricityImbalanceRequest* state. It sends on topic *"EI/+/ElectricityImbalanceRequest"* the *day_index* it needs the imbalance. The *day_index* will be "0" for the current day, "1" for the next day "-1" for the previous day etc. The Electricity Imbalance agent receives the message and enters the *ReceiveElectricityImbalanceRequest* state. Follows the *HandleElectricityImbalanceRequest* state, where the calculation of the electric energy imbalance takes place, and enters the *SendElectricityImbalance* state. The *imbalance* is sent on topic *"EI/+/ElectricityImbalance"*, the Service User receives the information, enters the *RecieveElectricityImbalance* state, and finishes the protocol.

There is, also, another capability that can be enabled to reduce the number of messages and the computational power on the system. The electrical imbalance agent can periodically broadcast the new imbalance or after a big update in the new imbalance. The *imbalance* is published on topic *"EI/ElectricityImbalance"* where all the Service Users can be subscribed. In this way, we provide more frequent and useful updates, when it is needed and more used efficient, when there are not changes on imbalance. However, the request and the broadcast features must coexist and the first can still be used from Service Users the did not received the periodic message or just registered on the system and needs to know some historical data.

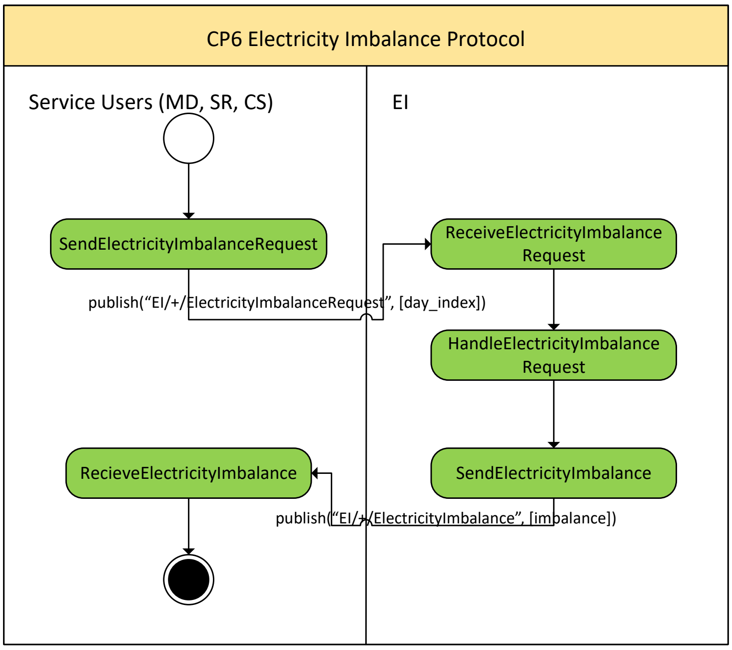


Figure 5.7: The model of the Electricity Imbalance Protocol (CP6)

5.1.7 Electricity Prices Protocol (CP7)

The Electricity Prices Protocol provides the communication between the Charging Station and Mechanism Design agents. The Mechanism Design informs the Charging Station about the purchasing and selling prices each time step for a day.

The Charging Station starts the protocol, enters the *SendElectricityPricesRequest* state and publishes on topic *"MD/+/ElectricityPricesRequest"* the *day_index* related to the current day as described previously on 5.1.6. The Mechanism Design receives it, enters the *ReceiveElectiricityPricesRequest* state, and with knowledge of the imbalance calculates by using the selected pricing algorithm (5.3.1) the prices for each time step in the *HandleElectricityPricesRequest* state. Then, in the *SendElectricityPrices* state, the Mechanism Design publishes the *buy_prices* and *sell_prices* on the *"MD/+/ElectricityPrices"* topic. The Charging Station agent receives the prices, enters the *ReceiveElectricityPrices* and terminates the protocol

The Electricity Prices Protocol can behave as we described the Electricity Imbalance Protocol on 5.1.6. The Mechanism Design has the capability to send periodically or after it receives a big change of the imbalance the updated prices on the topic *"MD/ElectricityPrices"* that can be subscribed from every Charging Station Agent.

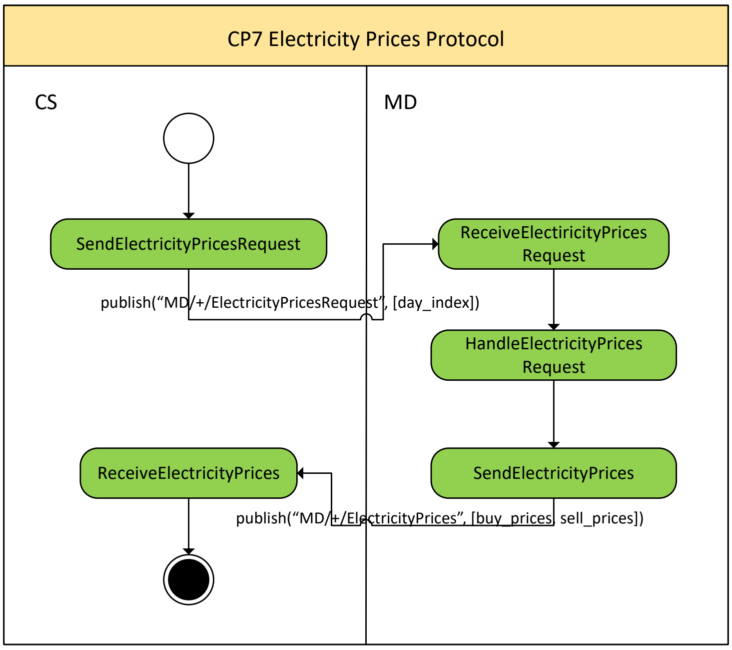


Figure 5.8: The model of the Electricity Prices Protocol (CP7)

5.1.8 Charging Station Update Schedule Protocol (CP8)

The Charging Station Update Schedule Protocol is the protocol that connects the Charging Station agent and the Service Providers that are the Electricity Imbalance and the Mechanism Design agents. The Charging Station agent regularly informs the Service Providers about the amount of energy it needs to get from the Grid and the amount it can provide back each timestep.

The protocol begins with the Charging Station entering the *SendUpdatedChargingSchedule* state and publishing the *schedule* of charging and discharging. Follows the *ReceiveRequestUpdatedStationSchedule* and the *HandleUpdateStationSchedule* where the Service Providers save the updated information. The Electricity Imbalance uses the schedule to calculate the new imbalance and the Mechanism Design calculate the payments that must be done and may be required by some pricing algorithms that the charging stations' production and consumption is required. Then, the Service Provider enters the *SendUpdateScheduleOutcome* state and publishes to the topic *"EI/+/UpdateScheduleOutcome"* for the Electricity Imbalance and *"MD/+/UpdateScheduleOutcome"* for the Mechanism Design, the *outcome* of the update that can be "Successful update" or "Failed update. Reason" with the reason for example can be that the Charging Station agent is not register in the Service Provider. The Charging Station receives the result, enters the *ReceiveUpdateScheduleOutcome* state, and the protocol is finished.

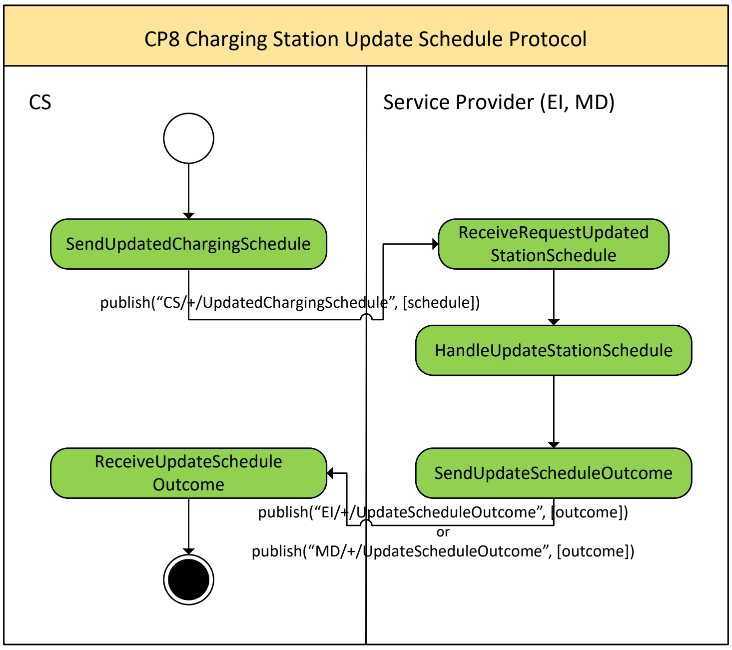


Figure 5.9: The model of the Charging Station Update Schedule Protocol (CP8)

5.1.9 *Producer Consumer Registration Protocol (CP9)*

This protocol connects the Service Users, the Electric Producer,and the Electric Consumer, with the service providers, Electricity Imbalance and Mechanism Design agents.

The Service User initiate the protocol and enters the *SendRegistrationRequest*. In case of the Service User being producer, it will publish on topic *"EP/+/RegisterElectricityProducer"* the *EP_info* message that contains information about type of the producer (e.g solar panel, wind turbines). Else if it is a consumer, it will send on topic *"EC/+/RegisterElectricityConsumer"* the *EC_info* message that specify the category of consumer it belongs (e.g. residential, commercial, industrial). The Service Provider receives the message, enters the *ReceiveRegistrationRequest* state, and stores the information of the Service User in the *HandleRegistrationRequest* state. Then, it enters the *SendRegistrationOutcome* state publishes the *outcome* of the registration on topic *"EI/+/RegistrationOutcome"* for the Electricity Imbalance agent and on *"MD/+/RegistrationOutcome"* for the Mechanism Design agent. The Service User receives the message, enters the *ReceiveRegistrationOutcome* state, and terminates the protocol.

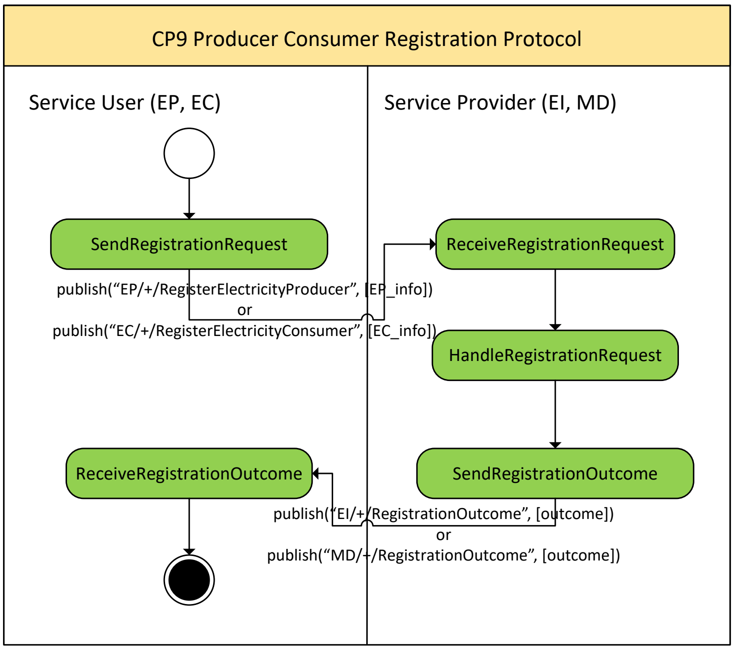


Figure 5.10: The model of the Producer Consumer Registration Protocol (CP9)

5.1.10 Update Expected Production/Consumption Protocol (CP10)

This protocol is responsible for providing connection between Service Users (Electricity Producer, Electricity Consumer agents) and Service Providers (Electricity Imbalance and Mechanism Design agents) to exchange information about their updated expectations of production and consumption. Each time a Service User charges its expected energy profile, informs through this protocol the interested Service Providers.

The Service User starts the protocol, enters the *SendUpdateExpectedEnergyProfile* state, and if it is a producer sends the *expected_production* on topic *"EP/+/UpdateExpectedProduction"* or if it is a consumer sends the *expected_consumption* on topic *"EC/+/UpdateExpectedConsumption"*. The Service Provider receives the message, enters the state *ReceiveUpdateExpectedProfileRequest*. Then, it processes and stores the updated information in the state *HandleUpdateExpectedProfileRequest* and enters the *SendUpdateProfileOutcome* and sends the *outcome* of the update in the topic *"EI/+/UpdateProfileOutcome"* if it is the energy imbalance agent or in *"MD/+/UpdateProfileOutcome"* if it is the Mechanism Design agent. The Service User receives the outcome, enters the *ReceiveUpdateProfileOutcome* state, and finishes the protocol.

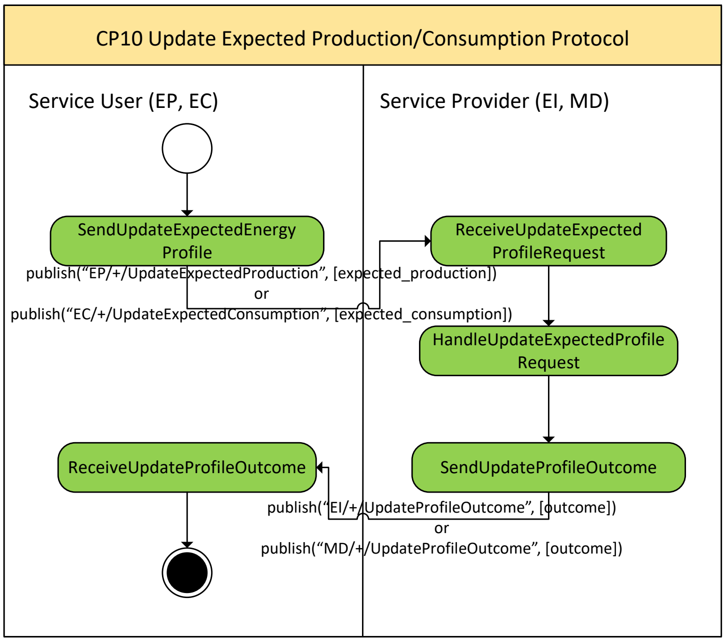


Figure 5.11: The model of the Update Expected Production/Consumption Protocol (CP10)

5.1.11 Update Energy Profile Confidence Protocol (CP11)

In this protocol, service users inform Service Providers about their confidence in the last sent energy profile. The role of Service User can be played from an Electricity Producer, an Electricity Consumer or a Charging Station agent and the role of Service Provider from Electricity Imbalance and Mechanism Design agents.

The Service User starts the protocol and enters the *SendUpdateEnergyProfileConfidence*. The *confidence* sends on topic *"EP/+/UpdateConfidence"*, *"EC/+/UpdateConfidence"*, and *"CS/+/UpdateConfidence"* from Electricity Producer, Electricity Consumer, and Charging Station agents, respectively. The Service Provider receives the message and enters the state *ReceiveUpdateConfidenceRequest*. Manage and store the confidence received in *HandleConfidenceUpdateRequest* and follows the *SendConfidenceUpdateOutcome* state where the Electricity Imbalance and the Mechanism Design publishes *outcome* on topics *"EI/+/UpdateConfidenceOutcome"* and *"MD/+/UpdateConfidenceOutcome"*, respectively. The Service User receives the message, enters *ReceiveUpdateConfidenceOutcome*, and completes the protocol.

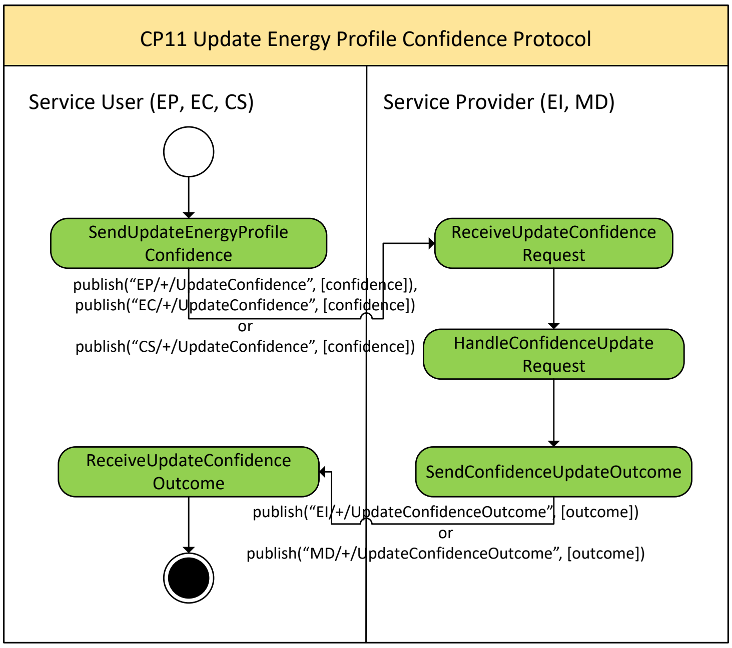


Figure 5.12: The model of the Update Energy Profile Confidence Protocol (CP11)

5.1.12 Update Station Availability Protocol (CP12)

This protocol connects the Charghing Station and the Station Recommender agents. After a change in charging slot availability, for example, a slot is reserved, the Charging Station communicates with the Station Recommender agent to inform it about the new availability schedule. This protocol supports the Station Recommender to recommend an already reserved slot, so we avoid the reservation conflicts.

The Charging Station initiates the protocol, enters the state *SendUpdatedStationAvailability*, and publishes on topic *"CS/+/UpdatedStationAvailability"* the *recommendation* generated by the Station Recommender and contains details of arrival and departure time, and the charging slot is reserved. The Station Recommender receive the update message and enters the *ReceiveRequestUpdateAvailability* state. Follows the *HandleStationAvailabilityUpdate* state where the update on charging slot takes place. Then enters the state *SendUpdateAvailabilityOutcome* and publishes on *"CS/+/UpdateAvailabilityOutcome"* the *outcome* of the update. The Charging Station agent receives the outcome message, enters the *ReceiveUpdateAvailabilityOutcome* state, and terminates the protocol.

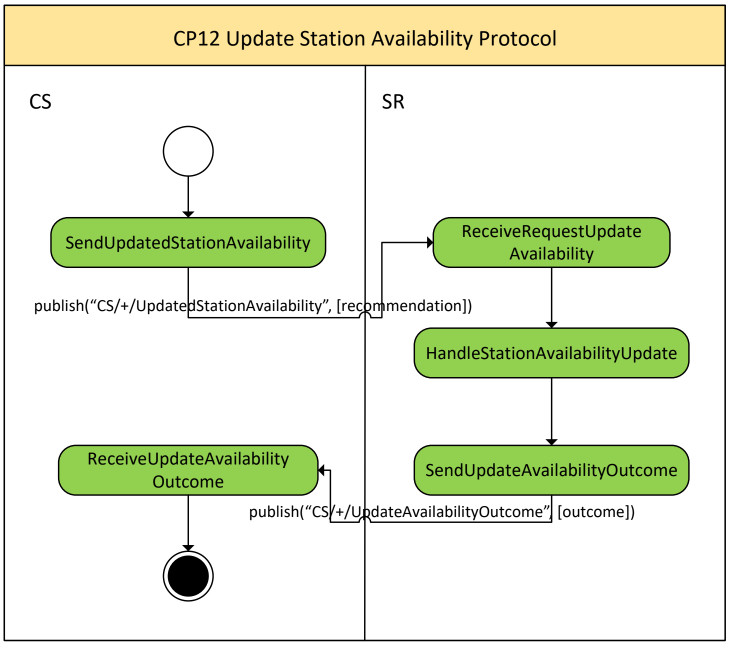


Figure 5.13: The model of the Update Station Availability Protocol (CP12)

5.2 Agent's Behavior

This section describes the internal actions of each stakeholder. There is a state diagram for each agent that shows its operation, behavior and a detailed description of it.

The figures are used to show the flow of action for a agent. START-states show the beginning of the execution (circle). The yellow states show the operations an agent can perform and the blue numbered boxes are for showing the separated tasks an agent can perform inside the *Main* operation. The grey diamonds a decision or branching point. Lines representing different decisions emerge from different points of it. Basic states (shown with green color) are where agent activities are executed and END-states show where execution ends (black dots within a circle).

Transitions from one state to the other occur *(i)* when the activity of the source state finishes and there is no event on the arrow, or *(ii)* when the event on the arrow takes place.

5.2.1 Electric Vehicle Agent

The intra-agent control model for the Electric Vehicle agent is shown in Figure 5.14.

Note that, for simplicity of representation, the protocol roles that the agent realizes are shown as basic states. These can be expanded to the relevant states in the respective protocols. For example, the *CP1 Charging Recommendation Protocol* state must be replaced by the *EV* state of Figure 5.2.

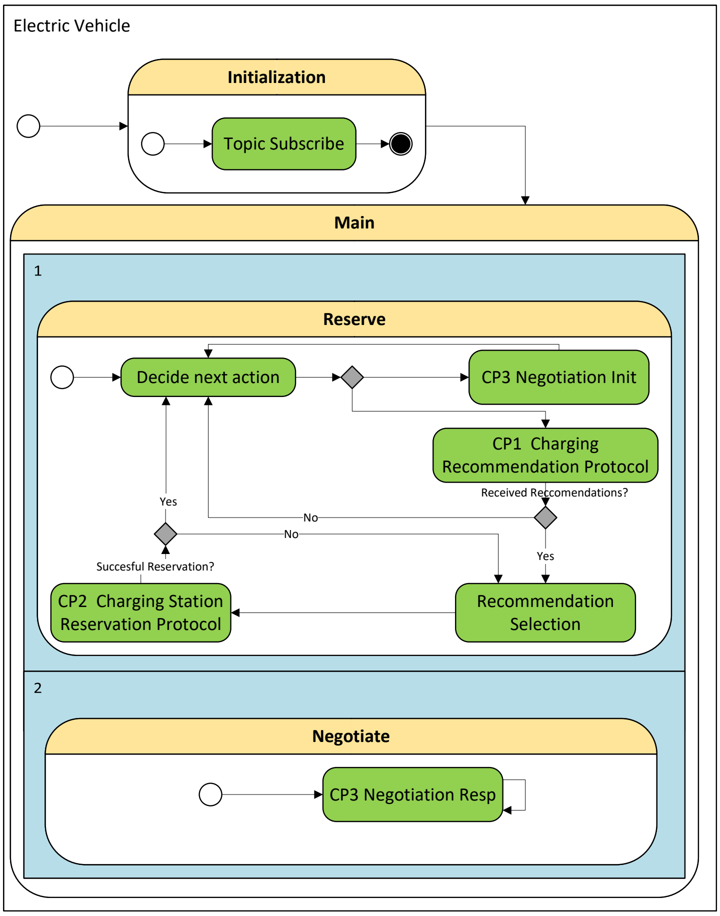


Figure 5.14: The intra-agent model of the Electric Vehicle agent.

At the beginning of its operation, the agent subscribes to the topics that must be connected and performs initialization activities (enters the *Initialization* state) . Then, the agent enters both the *Negotiate* and *Reserve* orthogonal components (Fig.5.14). In the *Reserve* component it makes a transition to the *DecideNextAction* basic state. In this state, the agent makes decisions regarding the charging of the Electric Vehicle (preferences and preferred location).

If the agent suggests making a new recommendation request, it checks if there are recommendations from Station Recommender Agent that meet its needs. If there are not it must calculate again its preferences and send them to the Station Recommender until there is a recommendation or it performs another action. Whenever the agent decides to arrange a forthcoming charging, it enters the *CP1 Charging Recommendation Protocol* state, then the *Recommendation Selection* state (to select the best offer), and, finally the *CP2 Charging Station Reservation Protocol* to reserve the selected slot. In case the reservation is not completed, the Electric Vehicle can go back to the *Recommendation*

Selection state and select an other Charging Station. If charging slot is successfully reserved, it returns to the *Decide Next Action* 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 (Electric Vehehicle or Charging Station) the roles it defines are that of the initiator (*Init*) and of the responder (*Resp*). As the reader can see, the Electric Vehicle can act either as an initiator (entering the *CP3 Negotiation:Init* state) or as responder (entering the *CP3 Negotiation:Resp* state).

5.2.2 Charging Station Agent

The internal behavior of the Charging Station agent is shown in Figure 5.15. Begins with the *Initialization* state, where takes place the *Topic Subscribe* state that the agent connects to the MQTT topics it is interested in and can have access. Afterwards, it starts the *CP4 Charging Station Registration* to inform the Electricity Imbalancer, the Mechanism Design and the Station Recommender agents about its permanent specification and that it starts to operate. When the protocol finishes, the initialization process ends and follows the operation *Main*, which contains four separate functions that are numbered and displayed in light blue boxes.

The *Charging Reservation* process starts with the *ReceiveChargingReservationRequest* and enters *HandleChargingReservationRequest* state. The Charging Station activates the *CP5 Authenticate Recommendation Protocol*. If authentication fails, the protocol ends; otherwise, if it authenticates successfully, it enters the *ScheduleAndMakeReservation* where it performs the schedule according to the selected scheduling algorithm and reserves the Charging Slot. If charging scheduling and reservation are completed successfully, the *CP8 Charging Station Update Schedule Protocol* starts to inform the Electricity Imbalancer, the Mechanism Design agents about the energy consumption per time step. Then, it starts *CP12 Update Station Availability Protocol* to inform the station recommender agent about the availability of the new charging slot and finishes the state *HandleChargingReservationRequest*. It enters the *SendReservationOutcome* state where the schedule and the reservation will be sent to the interested Electric Vehicle otherwise sends the reason of failure and terminates the process.

The secondly displayed (we remind the reader that all processes can be operated separately and parallel) process starts when the Charging Station agent needs to change a reservation. It initiates *CP3 Negotiation Protocol* and after its completion makes two checks. If there is a change on the charging schedule, then it starts the *CP8 Charging Station Update Schedule Protocol* to inform the Mechanism Design and the Electricity Imbalance. If there is a change on the arrival or departure time the Station Recommender must be informed by *CP12 Update Station Availability Protocol* and finishes the process.

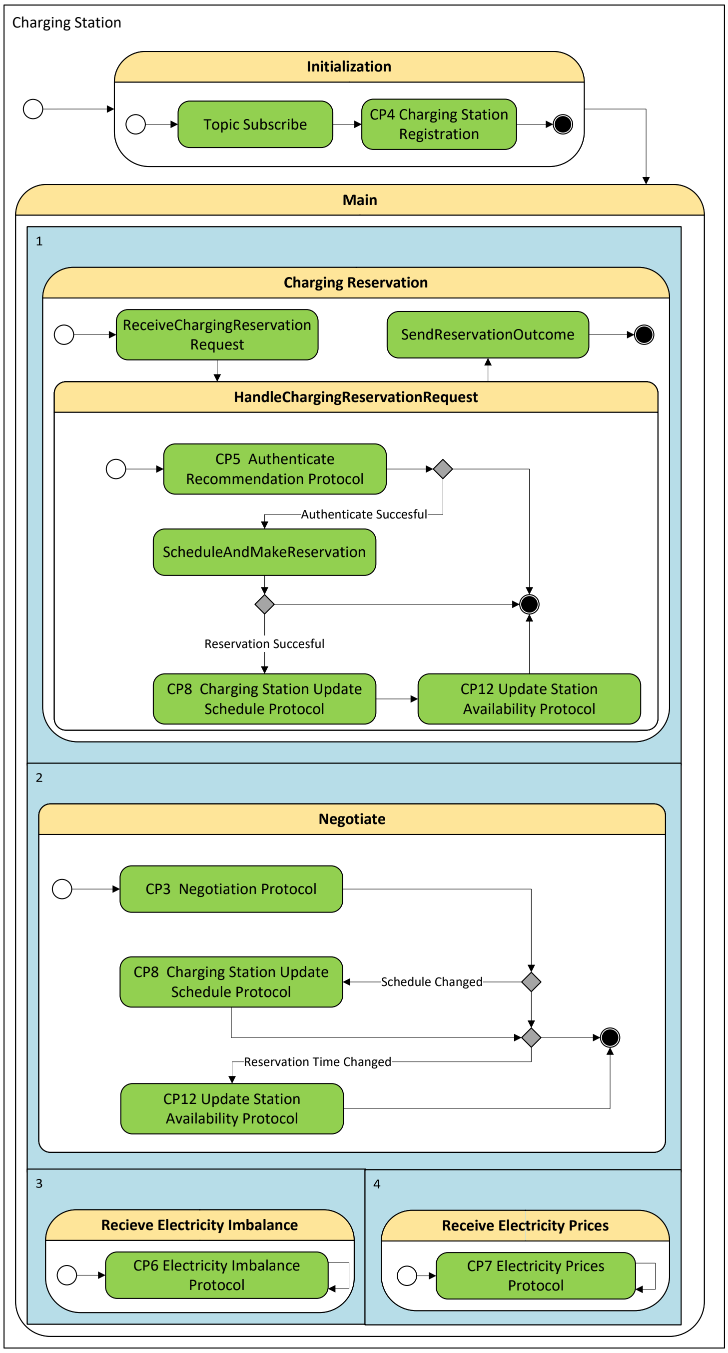


Figure 5.15: The intra-agent model of the Charging Station.

The third and fourth processes start when the Charging Station receives electricity imbalance and prices, from the Electricity Imbalancer and the Mechanism Design agents, and uses the *CP6 Electricity Imbalance Protocol* and *CP7 Electricity Prices Protocol* to store the newly received data.

5.2.3 Station Recommender Agent

The intra-agent functionality of the Station Recommender agent is shown in Figure 5.16. In the *Initialization* state the *Topic Subscribe* process takes place to make sure that the agent will receive the messages that needs to be informed.

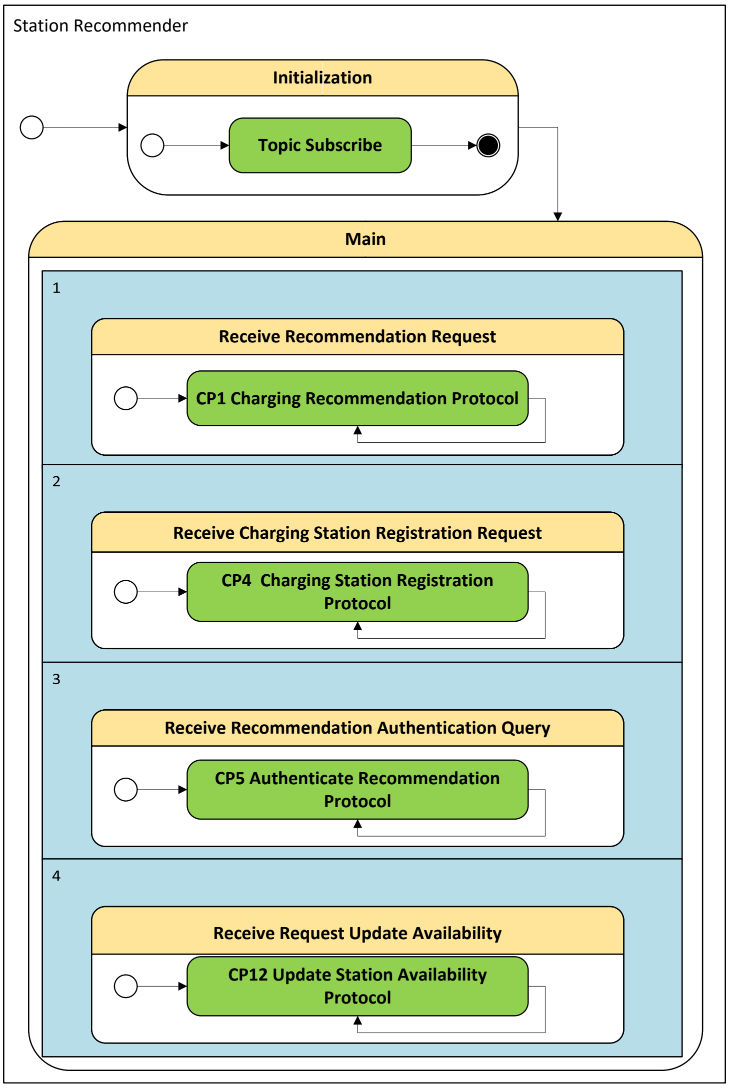


Figure 5.16: The intra-agent model of the Station Recommender.

Then it enters the *Main* state where the protocols will be enabled on the arrival of the corresponding message. If the Station Recommender receives a recommendation request, it will take into account the preferences of the Electric Vehicle. It will use *CP1 Charging Recommendation Protocol* to send the most suitable Charging Stations for Electric Vehicles that match their preferences. The Station Recommender receives information about the new newly opened Charging Station and uses *CP4 Charging Station Registration Protocol* to record and possibly add them to future recommendations. It, also, provides authentication services for the recommendations that has sent. When a recommendation needs to be verified that it is genuine the *CP5 Authenticate Recommendation Protocol* activates. The Station Recommender must be informed about the availability of the Charging Stations to generate correct recommendations and do not recommend charging slots that are already reserved in that specific time period. So on the arrival of a availability update message the *CP12 Update Station Availability Protocol* activates and stores the update.

5.2.4 Electricity Imbalance Agent

The Electricity Imbalance agent behaves according to Figure 5.17. It starts with *Initialization* where the agent subscribes to the MQTT topic that must receive the published messages on them.

On the first two operations we see the registration request of the Charging Station, the Electricity Producer and Consumer that activates the *CP4 Charging Station Registration Protocol* and the *CP9 Producer Consumer Registration Protocol*. The registration process is necessary in our system to make sure we have knowledge of the maximum production, consumption, and details, attributes about the prosumers. The next two operations describe the update of the energy consumption or generation for the Charging Station in *CP8 Charging Station Update Schedule Protocol* and for Electricity Producers and Consumers it uses the *CP10 Update Expected Production/ Consumption Protocol*

The fifth displayed operation informs the Electricity Imbalance agent about the confidence of the Charging Station, the Electricity Producers and Consumers on the amount of energy they are going to generate or consume. The last operation is the most significant for the agent and it is used to inform the interested agents about the total generation and consumption of the energy in the smart grid. After it receives an electricity imbalance request, it activates the *CP6 Electricity Imbalance Protocol* where the calculation of the imbalance and the publication of the requested message takes place.

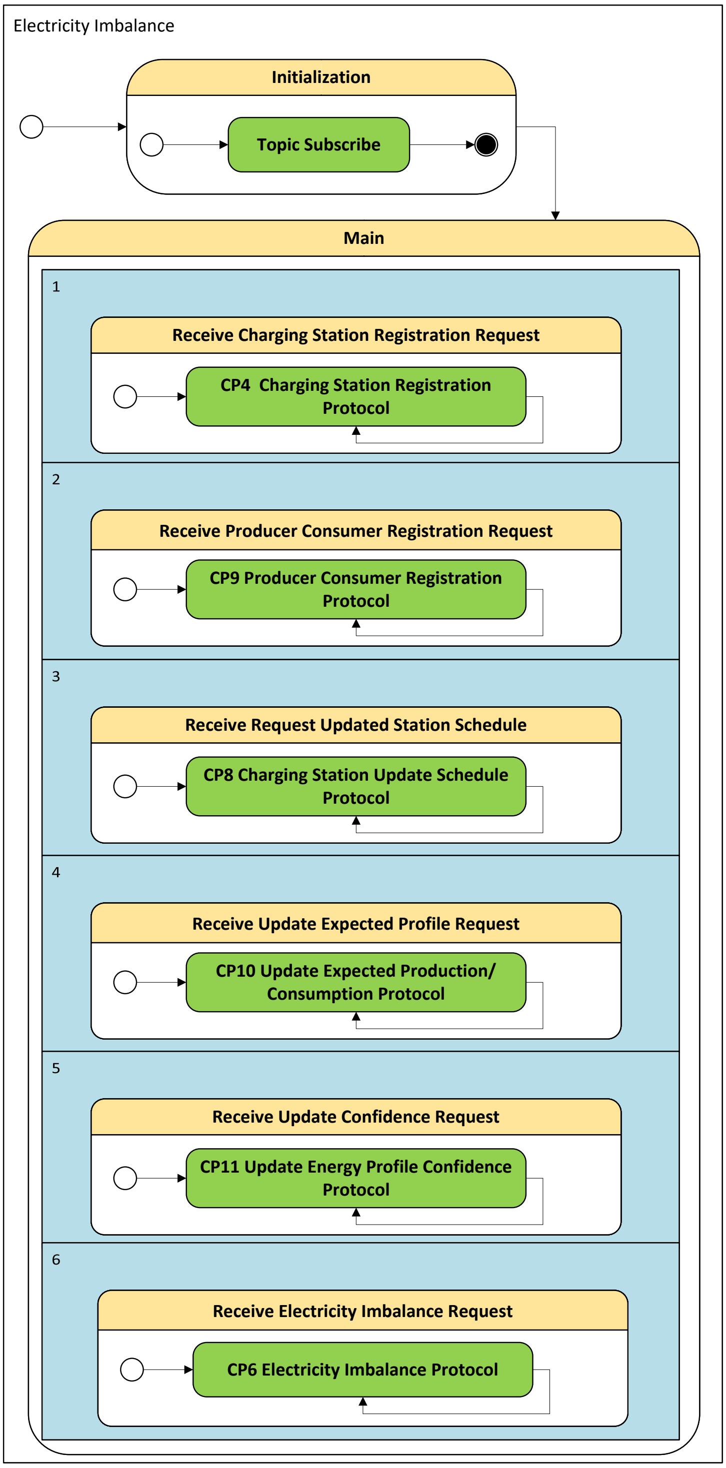


Figure 5.17: The intra-agent model of the Electricity Imbalance.

5.2.5 *Mechanism Design Agent*

Figure 5.18 displays the intra agent behavior of the Mechanism Design agent. It begins with the *Initialization* process that contains the subscription on MQTT topics it needs to operate. Then, it enters the *Main* state where the agent responds to the incoming messages.

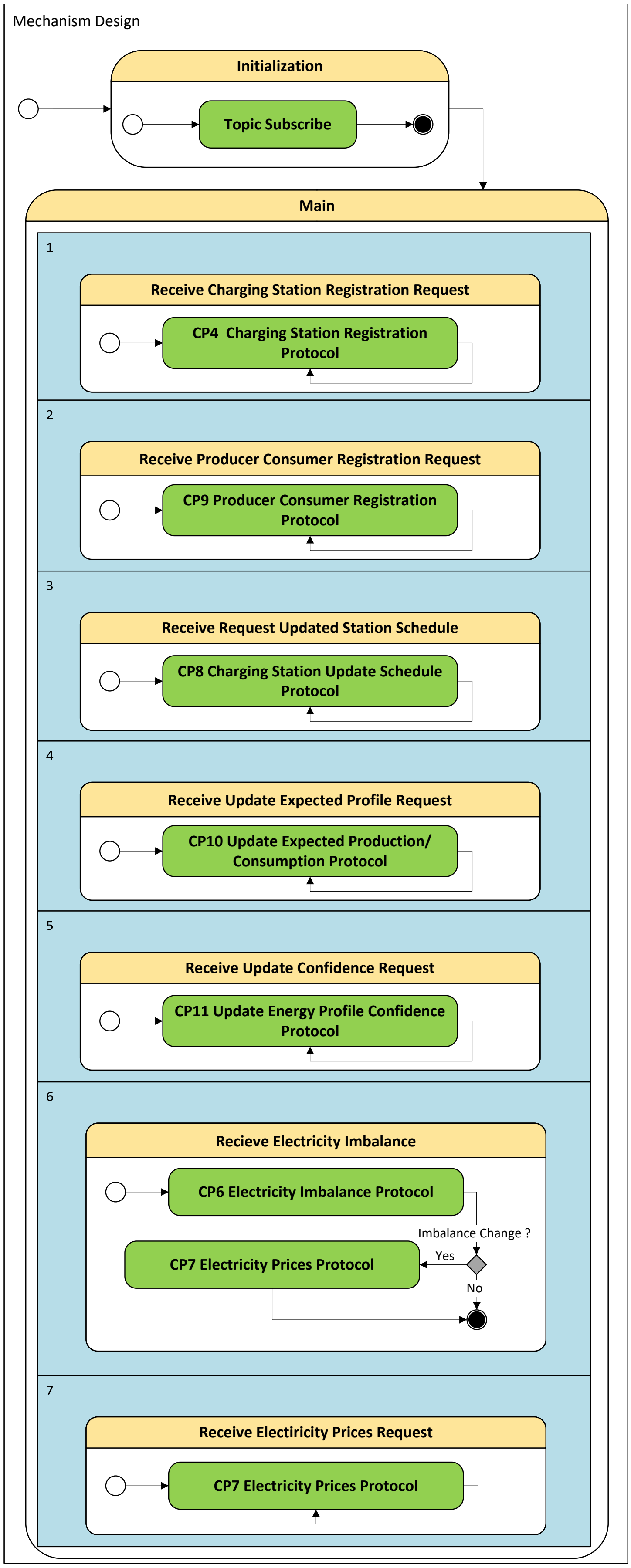


Figure 5.18: The intra-agent model of the Mechanism Design.

It receives registration request from Charging Stations, Electricity Producers, and Consumers, enables the *CP4 Charging Station Registration Protocol* and the *CP9 Producer Consumer Registration Protocol* to store the necessary information about their capabilities. The *CP8 Charging Station Update Schedule Protocol* and the *CP10 Update Expected Production/ Consumption Protocol* are used to communicate with the Charging Stations, Electricity Producers and Consumers about their expected production and future energy needs. Also, the three agents mentioned send how confident they are about their prediction and the Mechanism Design uses the *CP11 Update Energy Profile Confidence Protocol*. When the agent receives an Electricity Imbalance message, it activates the *CP6 Electricity Imbalance Protocol*. If the new imbalance changes significantly and affects the prices, then it activates *CP7 Electricity Prices Protocol* to inform the interested agents about the new prices. The *CP7 Electricity Prices Protocol* is used if an electricity price request is received to send the most recent calculated buying and selling prices for an energy unit (usually 1 kWh).

5.2.6 Electricity Producer Agent

Figure 5.19 describes the internal agent model of the implemented Electricity Producer. It starts with the *Initialization* where take place the *Topic Subscription* and the *CP9 Producer Consumer Registration Protocol*. The first process is needed to make sure the agent can receive the incoming messages, and the protocol informs the interested service providers about the type (e.g. photovoltaic panels, wind turbines), the rated power, and the location. The *Main* process follows where contains the *Periodic Energy Update* that activates periodically (for example, every day). It begins with *Calculate Production* where the agent can make an estimate of future production taking into account date, time, weather forecast and previously recorded and historical data and uses *CP10 Update Expected Production/Consumption Protocol* to publish it. After, it compares the expected production amount with similar past dates. If those values are close, then we have a good estimate of the expected production. The Electricity Producer agent uses the *CP11 Update Energy Profile Confidence Protocol* to send the confidence to the Mechanism Design and the Electricity Imbalance agent.

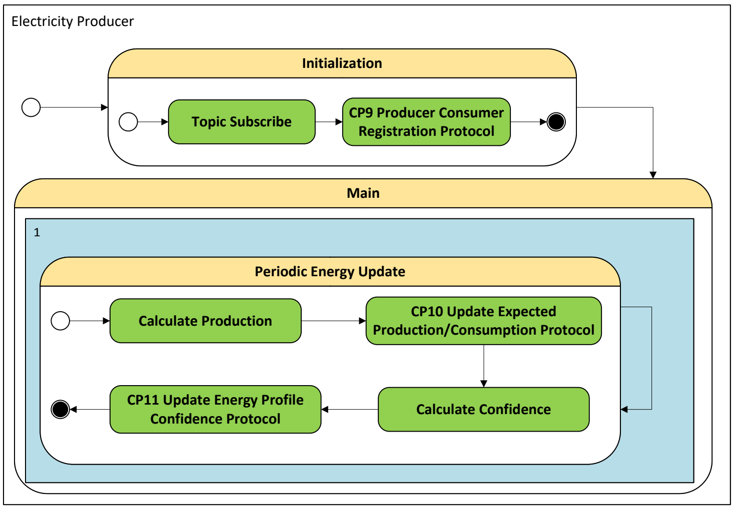


Figure 5.19: The intra-agent model of the Electricity Producer.

5.2.7 Electricity Consumer Agent

The internal behavior of the Electricity Consumer agent is shown in Figure 5.15. The agent starts with *Initialization* where the subscription is made on MQTT Topics and *CP9 Producer Consumer Registration Protocol*. CP9 is used to inform the Mechanism Design and Electricity Imbalance agents about the type of the consumer (residential, commercial, industrial).

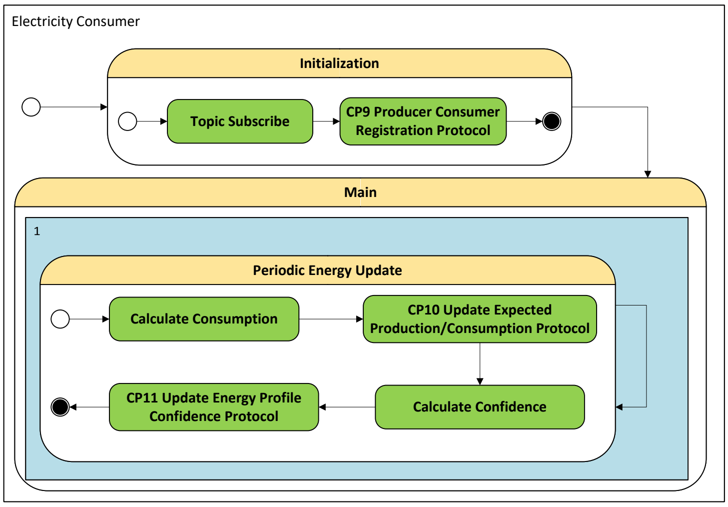


Figure 5.20: The intra-agent model of the Electricity Consumer.

Then, the *Main* process follows that contains the *Periodic Energy Update* that is a repeated process and activates after a specific time period, for example every day. It starts with the *Calculate Consumption* which takes into account the consumption of previous days and from previous years if this information is available. It publishes the consumption using the *CP10 Update Expected Production/Consumption Protocol*. After, the Electricity Consumer agent calculates the confidence from previous collected data and publish it with the use of *CP11 Update Energy Profile Confidence Protocol*.

5.3 Implemented Agent Strategies

For the purposes of evaluation via simulation, we need to test different methods and compare their effects on the system in simulation mode. To this end, we implemented two pricing algorithms used by the Mechanism Design agent to observe how they contribute to grid stability, i.e. to reducing the energy surplus and deficit peaks. We also implemented three scheduling approaches that determine when and how much energy is exchanged between Charging Station and Electric Vehicle agents.

5.3.1 Price Calculation Algorithms for the Mechanism Design Agent

A) *NRGCoin pricing algorithm:* This pricing mechanism is inspired by the one in [27], and aims at incentivizing producers and consumers to balance supply and demand. Let the aggregate supply and demand 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 to buy and sell. 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:* According to this mechanism proposed in [26], we estimate the evaluation of energy with respect to the cost of the Electric Vehicle 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 $d_t^{i'}$, the demand of Electric Vehicle agent i during the interval t. The mechanism can adjust prices to motivate agents to charge their electric vehicles when there is an energy surplus on the grid. The buying prices for the intervals $t \in \{1,...,T\}$ are given by:

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

5.3.2 Charging Scheduling Algorithms for the Electric Vehicle Agent

A) *First slot:* In this case, EVs charge their battery in the first available time interval, without taking into account if prices are better or worse.

B) *Lowest Prices:* In this approach, electric vehicles are trying to reduce their charging costs by choosing to charge during time periods when the energy prices are the lowest possible.

C) *V2G:* In this case, electric vehicles can discharge their batteries when prices are high to provide load to the rest of the grid, and then charge them back when prices are lower, nevertheless within the periods that electric vehicles are connected to a charger. For this purpose and inspired by [3] and [19], we used linear programming to minimize an objective function representing charging costs in the presence of constraints regarding the electric vehicle owner preferences and Electric Vehicle's charging specifications. The optimization function is the following:

$$\min \sum_t^T C_t^{G2V} + C_t^{deg} - I_t^{V2G} \tag{5.1}$$

subject to:

$$C_t^{G2V} = d_t^{G2V} * P_{max}^{G2V} * p_t^{buy} * dt \tag{5.2}$$

$$C_t^{deg} = f^{deg} * d_t^{V2G} * P_{max}^{V2G} * dt \tag{5.3}$$

$$I_t^{V2G} = d_t^{V2G} * P_{max}^{V2G} * p_t^{sell} * dt \tag{5.4}$$

$$d_t^{G2V} + d_t^{V2G} \leq 1, \;\; d_t^{G2V}, d_t^{V2G} \in [0,1] \tag{5.5}$$

$$\sum_t^T d_t^{G2V} * P_{max}^{G2V} * dt - d_t^{V2G} * P_{max}^{V2G} * dt = E_{need} \tag{5.6}$$

$$\sum_t^k d_t^{G2V} * P_{max}^{G2V} * dt - d_t^{V2G} * P_{max}^{V2G} * dt + E_{init} \leq c_{max}, k \in [1,T] \tag{5.7}$$

$$\sum_t^k (d_t^{G2V} * P_{max}^{G2V} * dt - d_t^{V2G} * P_{max}^{V2G} * dt) + E_{init} \geq c_{min}, k \in [1,T] \tag{5.8}$$

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. The d_t^{G2V} and d_t^{V2G} are decision variables for G2V and V2G in our optimization problem and can take values between 0 and 1, where intermediate values are assigned when it is optimal to charge or discharge at a fraction of the maximum 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 the degradation factor based on the method presented in [25], which is used to evaluate the degradation cost C_t^{deg} and dt is the duration of each time interval.

The constraints in expressions (5.5) - (5.8) must be satisfied during Electric Vehicle scheduling optimization. In (5.5) it is guaranteed that an electric vehicle will charge, discharge or stay idle in each time interval. The constraint (5.6) states that at the end of the charging session the electric vehicle's battery must be charged at the desired capacity E_{need} that the owner has set as the target. In constraints (5.7) and (5.8) we limit the allowable range of 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 until the end of each time interval, plus the initial amount of energy already stored in the battery.

Chapter 6

Experiments

In this section, we present the results of the simulations that illustrate the applicability of our proposed architecture. Specifically, we show four use cases of our system. The use cases provide comparative evaluations of the implemented strategies discussed in the previous section, or measure scalability-related aspects of the system. In addition, we provide a discussion of the requirements for the real-world deployment of our system.

All agent implementations are in Python and the duration of each simulation is 10 days. The datasets that we used are based on a collection of real data from a number of publicly available online resources. Specifically, consumption and production data originate from the ENTSOE platform, and Electric Vehicle data from the MyElectricAvenue project. Then, we used a data generator [60] that takes as input the datasets, employs a variety of models using this data as training sets; and thus generates new synthetic data, not identical to the input, but adhering to the same principles, and relationships.

The synchronization problem in the simulation mode is solved with the use of an extra input topic for all the agents. The initialization process for each agent is different and the synchronized start of a large number of agents is nearly impossible. So, we allowed the agents to complete their initialization process and wait until the message *"start"* is published on the topic *timer* externally. The start message arrives to all the agents with a very good accuracy, they synchronize their clocks and begin their *Main* state simultaneously.

The simulations were performed on a PC with an AMD Ryzen 5 1500X @ 3.5GHz processor and 8GB of RAM.

6.1 Charging Schedule Algorithms Comparison

In the first use case we compare the different Electric Vehicle charging scheduling methods, using the *NRGCoin* pricing mechanism described in Subsection 5.3.2. We remind the reader that these methods charge *(i)* during the first slots that the Electric Vehicle connects to a charger, *(ii)* during intervals with the lowest price for consumption, and *(iii)* with V2G capability, where the Electric Vehicle can also sell back to the grid some of the stored energy and recharge later, provided that the price difference between the discharge and recharge intervals generates profit. Figure 6.1 shows the average cumulative Electric Vehicle costs for the entire planning horizon. As we can see, the highest cost for the Electric Vehicle is given by the *first slot* method, which is expected as in this case the Electric Vehicle agent chooses to charge immediately without considering the energy price. By adopting the *Lowest Prices* method, the total cost of Electric Vehicle charging drops by approximately 33% at the end of the time horizon. Finally, by allowing *V2G* operations, the charging costs drop even more, 15% lower than those of *Lowest Prices*, and by 43% compared to the *first slot* method.

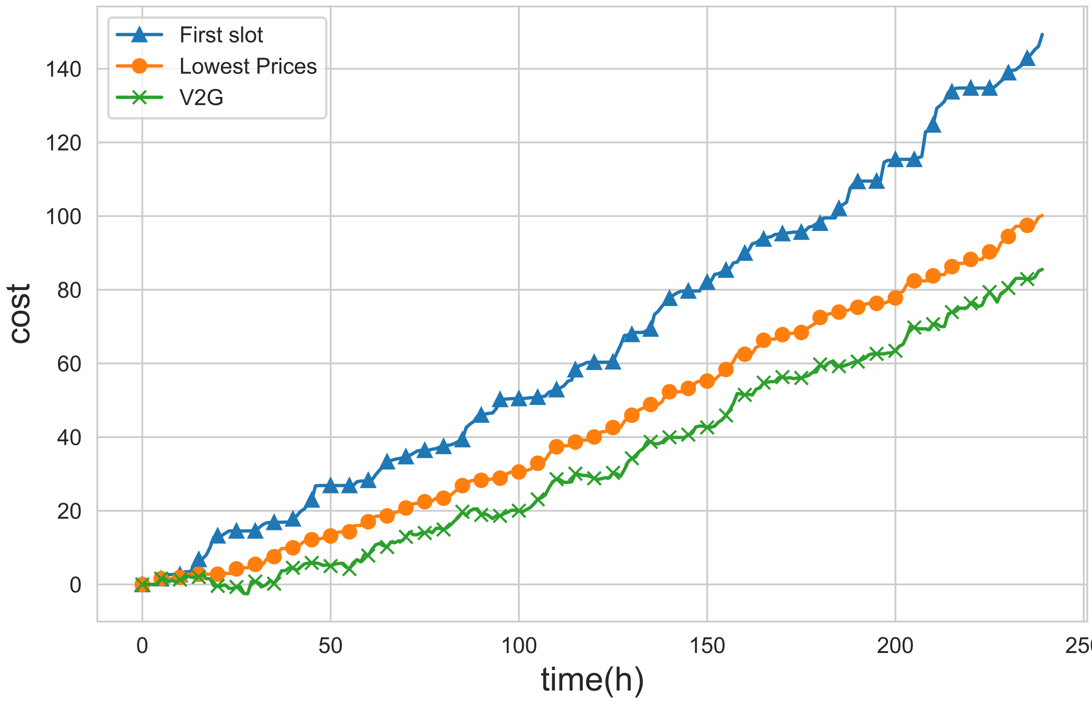


Figure 6.1: Average Cumulative Cost per EV using different charging scheduling methods.

Next, we account for the impact of the different charging scheduling methods on the aggregate energy imbalance. As a baseline, we consider grid imbalance without the Electric Vehicle demand. We calculate 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). Table 6.1 shows the significant impact of Electric Vehicles strategy on the energy imbalance. When using the *first slot* method, electric vehicles negatively affect the system, increasing the total imbalance and adding more than double the energy that must be produced to meet the demand in the grid. In parallel, the amount of energy wasted decreases, as Electric Vehicles consume energy that otherwise would not be consumed. In the case of the method *Lowest Prices*, the imbalance tangibly drops, and the available energy that is utilized and does not get wasted, increases by half (specifically, by 45.6%. The imports are also increased, due to the additional demand of the 100 Electric Vehicles and their occasional need to charge their batteries to continue their trip without caring about high energy prices and the energy shortage of the grid. An even better picture is obtained when *V2G* comes into play, with even lower imbalance (higher imbalance reduction, reaching 37.3%); less energy wasted (waste reduced by 49.1%; while imports are increased by only a very small rate (specifically, by 2.5%). Moreover, it achieves a larger reduction in the *Mean Absolute Percentage Error (MAPE)*, than the other two methods. MAPE measures the difference of the induced imbalance from a totally flat curve with a value of zero, which resembles the perfect match between supply and demand. This is clearly visible when plotting the imbalance across the time horizon for each method, as we do in Figure 6.2. In fact, it should be noted that *V2G* induces smaller peaks in the imbalance between demand and supply than the rest of the methods.

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%

Table 6.1: Energy differences on charging scheduling methods compared to baseline where there are no EVs. The MAPE of the original imbalance curve is 63.9%.

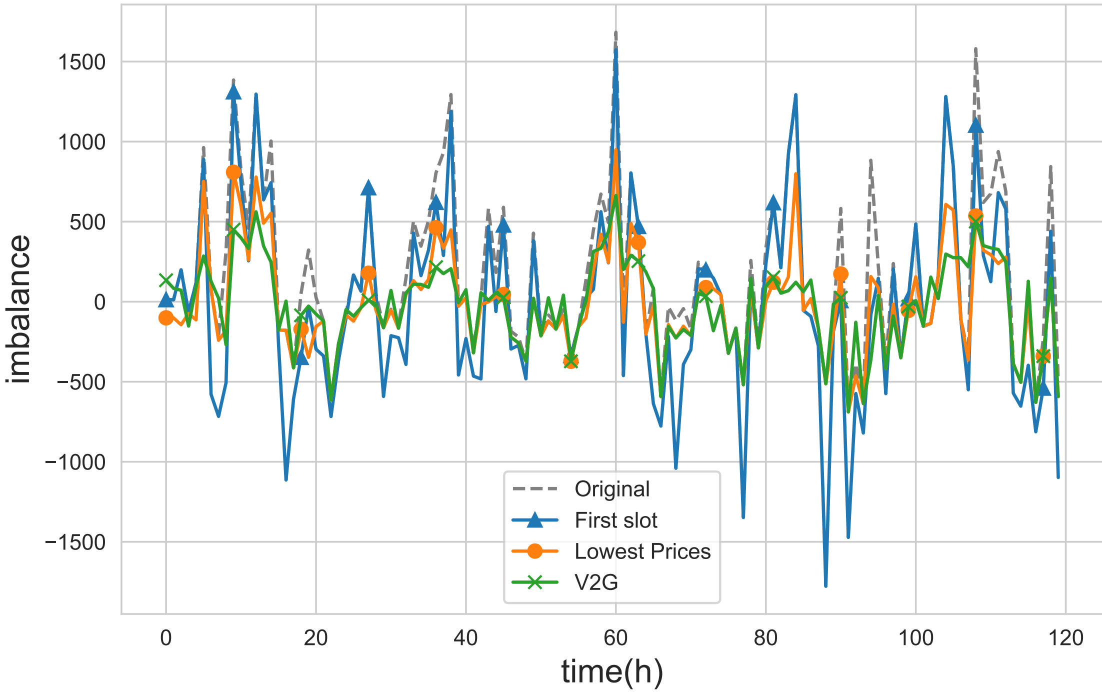


Figure 6.2: Imbalance using different charging scheduling methods.

6.2 Mechanism Design Algorithms Comparison

The second use case compares different pricing algorithms for the MD agent, in particular the *NRGCoin* pricing and the *adaptive pricing* presented in Subsection 5.3.1. Both methods aim to balance demand and supply, by setting higher prices for consumption during problematic intervals of negative imbalance, and lower for those with positive. The charging of Electric Vehicles for this use case is performed according to the *Lowest Prices* scheduling approach.

Considering that Electric Vehicle agents are rational and aim to reduce their expenses, the application of the two pricing algorithms results to demand being shifted to utilize the generated energy more effectively, thus leading to smaller peaks in the imbalance curve. Figure 6.3 shows that the algorithms have a similar effect on the stability of the grid. In Table 6.2, we can observe a similar behavior in reducing wasted energy and a slight outperformance of *NRGCoin* in imported energy and MAPE reduction.

Method	imbalance	wasted	imported	MAPE
NRGCoin	−31.4%	−45.6%	+16.4%	−44.5%
Adaptive Pricing	−31.3%	−45.6%	+17.1%	−42.7%

Table 6.2: Energy differences on pricing algorithms compared to baseline where there are no EVs. The MAPE of the original imbalance curve is 63.9%.

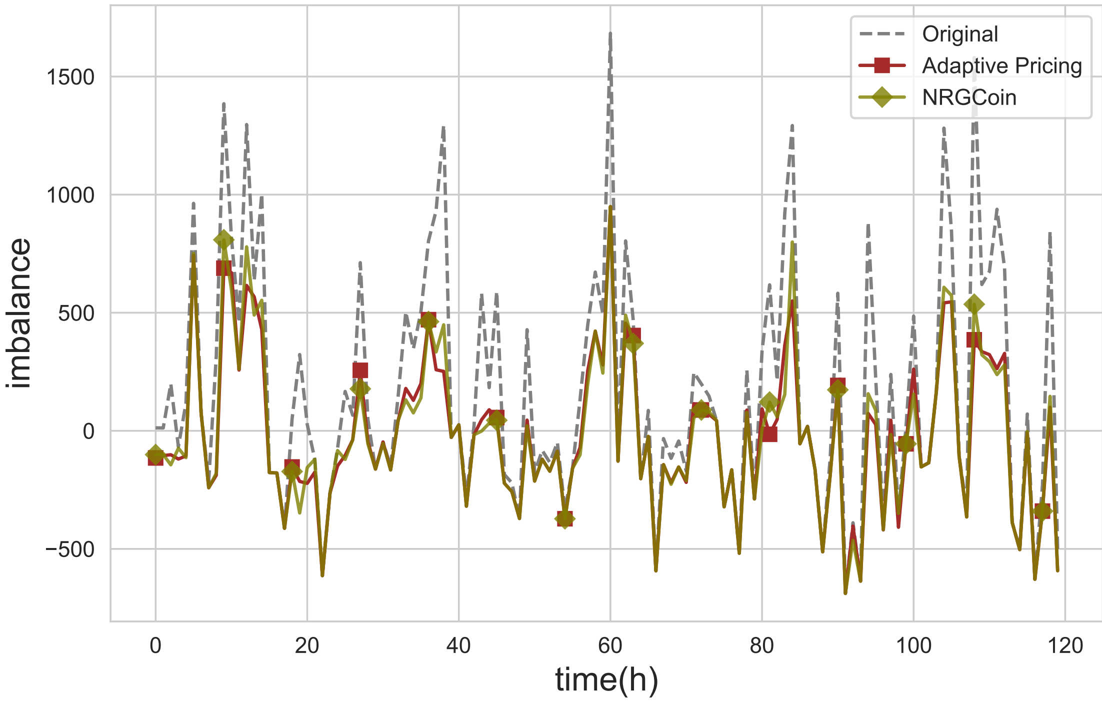


Figure 6.3: Adaptive pricing and NRGCoin mechanisms.

6.3 Charging Schedule Algorithms Comparison With Additional Charging Duration

In the third use case, we measure the total cumulative cost of Electric Vehicles, when increasing the duration of connection to chargers by 24 hours compared to the original data, by following the three different charging scheduling methods of the first use case. The results of Figure 6.4 show that by increasing the duration of connection, the methods *Lowest Price* and *V2G* manage to gradually reduce battery charging costs. This happens since the longer an Electric Vehicle is connected to a charger, the higher probability it has to find the most advantageous intervals to buy energy at from the grid— and also to sell it back to the grid in the case of V2G. As anticipated, again, the *V2G* method leads to lower charging costs than the other two, and the difference (mirroring this *V2G*'s advantage) increases as the duration of connection to a charger gets longer.

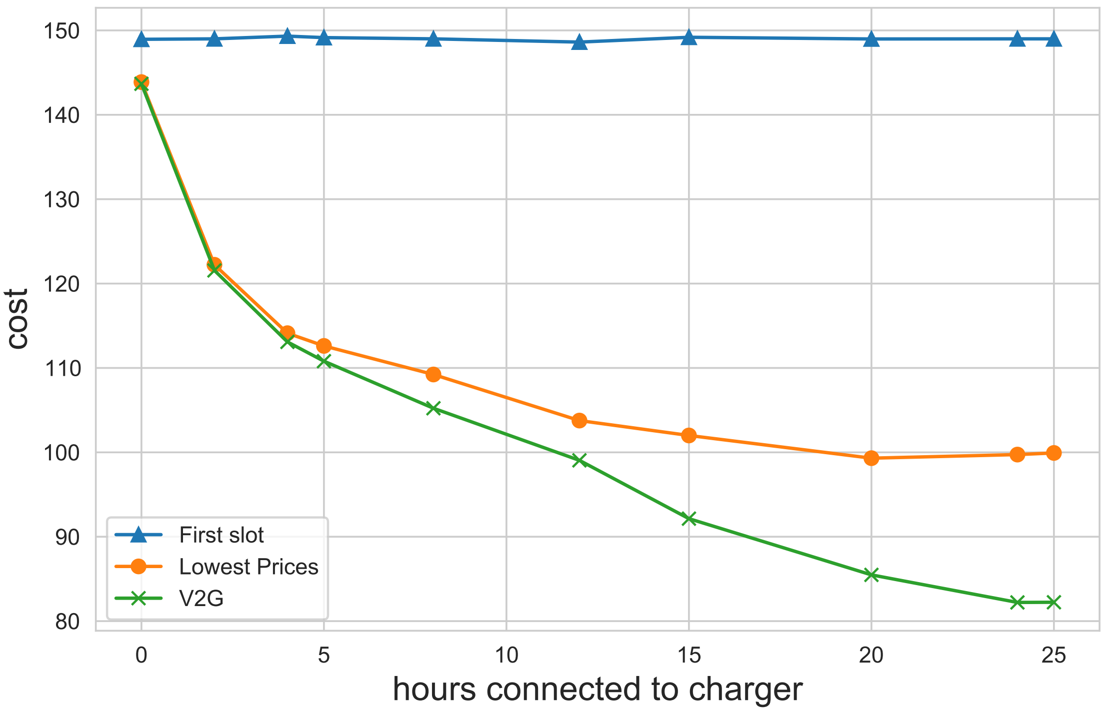


Figure 6.4: Cost comparison with varying amount of time that EVs are connected to a charging slot.

6.4 System Scalability With Respect To EVs Number

In the fourth use case, we count the total number of messages exchanged required for the scheduling of charging using our proposed cooperation protocols as the Electric Vehicle population increases. Figure 6.5 shows a linear increasing trend in the number of messages exchanged over a 10 day period.

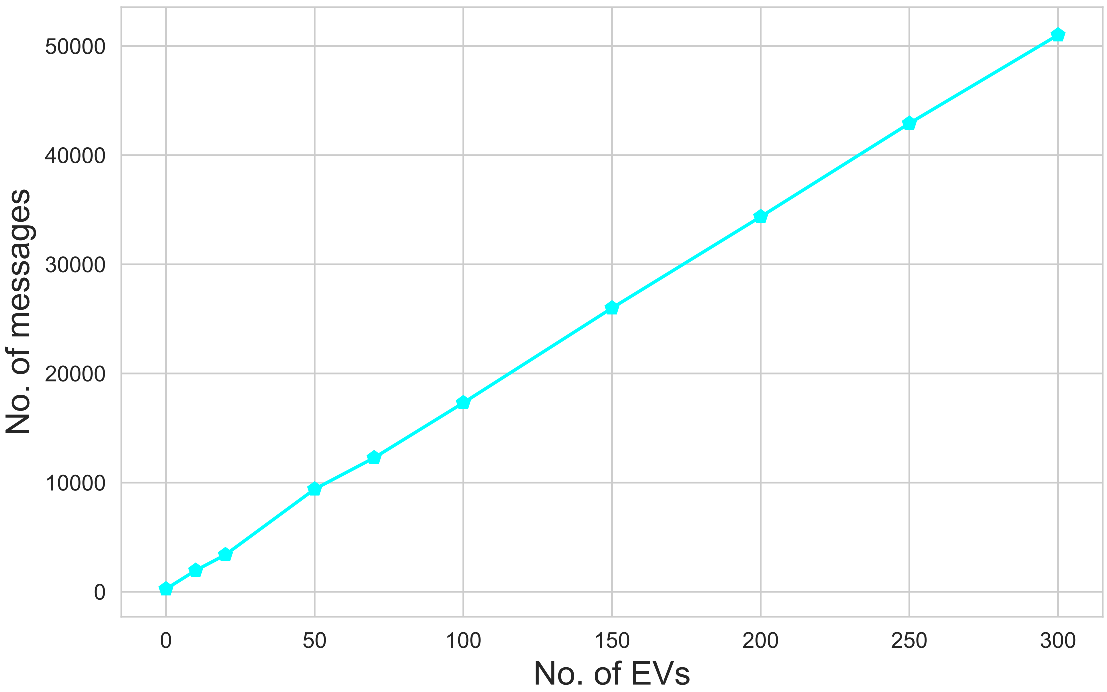


Figure 6.5: Message count for 10 days vs number of EVs

Chapter 7

Conclusions And Future Work

7.1 Conclusions

In this thesis, we presented a novel architecture for the V2G/G2V energy transfer problem domain, and provided implementations of agents as flexible microservices that are interconnected by an IoT platform. Our approach can be used for the exploration of various agent strategies in simulation mode, but is also readily deployable and can support real world trials.

The use of SYNAISTHISI IoT-based implementation offers direct applicability in real-world settings enabling digital twins models for Smart Grid ecosystems. It provides scalability to adapt the increased usage that may occur.

We also addressed the needs for openness, and the coverage of diverse business models via the definition of a number of key agent types, and the development of open protocols. These can be made available to any interested party, which can subsequently build their own agents given their expertise and business cases. This is demonstrated via presenting realistic use case scenarios.

We implemented and tested algorithms that schedule the charging process and pricing mechanisms that reflect the scarcity and the abundance of electric energy. We tested the cost efficiency of the scheduling algorithms when the charging session period is increasing and the linear growth of message count when the population of Electric Vehicles is raising.

7.2 Future Work

In terms of future works, there is much to be done in terms of populating the agents' components with actual machine learning that could applied on Electric Vehicle Agent to learn the preferences of the owner and foresee the next charging schedule. It can used also in Electricity Producer and Electricity Consumer Agents, with the knowledge, for example the weather forecast and the date, they can achieve accurate estimations for the future production and consumption.

The decision-making process could upgrade in Electric Vehicle Agent to allow a more complex Charging Station selection. The negotiation modules in the Electric Vehicle and the Charging Station will gain benefits against more naive agents.

The Station Recommender Agent can collect information about the users and recommend the stations that the owner and other owner with the similar preferences tend to select. It can also evolve to a service like accommodation or airplane ticket recommendation platforms that serve large amount of uses daily.

Moreover, it would be interesting to use our system in the real-world, first as part of a pilot study. This would allow us to test the perceived openness and the usability of the system, and to identify potential extensions, as well as important business models.

Appendix A

Topics And Messages In Our Implementation

In this Appendix, we provide detailed information about each topic and the structure of each message.

1. EP/+/RegisterElectricityProducer:

   Plus sign is replaced with Electricity Producer's ID (e.g. EP/EP100/RegisterElectricityProducer). The message contains a list with PowerSlot for the current day production (e.g. ['"dateTime": "2020-01-01T00:00:00", "kwh": 228.0', '"dateTime": "2020-01-01T01:00:00", "kwh": 543.0', ... , '"dateTime": "2020-01-01T23:00:00", "kwh": 241.0']).

2. EC/+/RegisterElectricityConsumer:

   Plus sign is replaced with Electricity Consumer's ID (e.g. EC/EC100/RegisterElectricityConsumer). The message contains a list with PowerSlot for the current day consumption (e.g. ['"dateTime": "2020-01-01T00:00:00", "kwh": 940.0', '"dateTime": "2020-01-01T01:00:00", "kwh": 472.0', ... , '"dateTime": "2020-01-01T23:00:00", "kwh": 491.0']).

3. CS/+/RegisterChargingStation:

   Plus sign is replaced with Charging Station's ID (e.g. CS/CS10001/RegisterChargingStation). The message contains a json string with Charging Station's attributes. (e.g. "id": "CS10001", "stationNetwork": "net1", "location": "latitude": 1.0, "longitude": 0.0, "chargingSlots": ["ratedPower": 120.0, "connectorType": "3 phase-DC", "category": "level 3", "slotID": 0, "netProfitPerKwh": 0.03, "reservations": [], "ratedPower": 22.0, "connectorType": "3 phase-32A per phase", "category": "level 2", "slotID": 1, "netProfitPerKwh": 0.02, "reservations": []], "reg": , "reservation_id": 100000, "schedules": , "linked_evs": [], "imbalance_plus": [], "imbalance_minus": [])

4. EP/+/UpdateExpectedProduction:

Plus sign is replaced with Electricity Producer's ID (e.g. EP/EP100/UpdateExpectedProduction). The message contains a list with PowerSlot for the next day expected production (e.g. ['"dateTime": "2020-01-02T00:00:00", "kwh": 144.0', '"dateTime": "2020-01-02T01:00:00", "kwh": 1282.0', ... , '"dateTime": "2020-01-02T23:00:00", "kwh": 818.0']).

5. EC/+/UpdateExpectedConsumption:

   Plus sign is replaced with Electricity Consumer's ID (e.g. EC/EC100/UpdateExpectedConsumption). The message contains a list with PowerSlot for the next day expected consumption (e.g. ['"dateTime": "2020-01-02T00:00:00", "kwh": 504.8', '"dateTime": "2020-01-02T01:00:00", "kwh": 515.3', ... , '"dateTime": "2020-01-02T23:00:00", "kwh": 657.5']).

6. EP/+/UpdateConfidence:

   Plus sign is replaced with Electricity Producer's ID (e.g. EP/EP100/UpdateConfidence). The message contains a list with ConfidenceSlot for the next day expected production confidence (e.g. ['"dateTime": "2020-01-02T00:00:00", "confidence": "mean": 427.63, "std": 57.59', '"dateTime": "2020-01-02T01:00:00", "confidence": "mean": 436.61, "std": 53.71', ... , '"dateTime": "2020-01-02T023:00:00", "confidence": "mean": 524.29, "std": 66.73']).

7. EC/+/UpdateConfidence:

   Plus sign is replaced with Electricity Consumer's ID (e.g. EC/EC100/UpdateConfidence). The message contains a list with ConfidenceSlot for the next day expected consumption confidence (e.g. ['"dateTime": "2020-01-02T00:00:00", "confidence": "mean": 460.96, "std": 65.80', '"dateTime": "2020-01-02T01:00:00", "confidence": "mean": 444.42, "std": 63.24', ... , '"dateTime": "2020-01-02T23:00:00", "confidence": "mean": 522.72, "std": 70.48']).

8. CS/+/UpdateConfidence:

   Plus sign is replaced with Charging Station's ID (e.g. CS/CS10001/UpdateConfidence). The message contains a list with ConfidenceSlot for the next day expected consumption confidence (e.g. ['"dateTime": "2020-01-02T00:00:00", "confidence": "mean": 578.83, "std": 78.45', '"dateTime": "2020-01-02T01:00:00", "confidence": "mean": 634.41, "std": 79.53', ... , '"dateTime": "2020-01-02T23:00:00", "confidence": "mean": 645.37, "std": 83.50']).

9. CS/+/UpdatedChargingSchedule:

   Plus sign is replaced with Charging Station's ID (e.g. CS/CS10001/UpdatedChargingSchedule). The message contains a list with PowerSlot for the charging schedule (e.g. ['"dateTime": "2020-01-01T07:00:00", "kwh": 0.0', '"dateTime": "2020-01-01T08:00:00", "kwh": 0.0', '"dateTime": "2020-01-01T09:00:00", "kwh": 0.0', '"dateTime": "2020-01-01T10:00:00", "kwh": 0.0', '"dateTime": "2020-01-01T11:00:00", "kwh": 27.23']).

10. CS/+/UpdatedStationAvailability:

    Plus sign is replaced with Charging Station's ID (e.g. CS/CS10001/UpdatedStationAvailability). The message contains the Recommendation that reserved (e.g. '"id": "rec3003", "arrival": "2020-01-01T07:00:00", "departure": "2020-01-01T12:00:00", "stationNetwork": "net1", "connectorType": "3 phase-60A per phase", "stationID": "CS10001", "issueDate": "2020-01-01T06:00:00", "slotID": 40, "pricePerKWH": 0.2, "totalKWHs": 27.23, "chargingPower": 43.0, "rank": 4.5, "location": "latitude": 4.0, "longitude": 0.0, "evid": "EV10012"').

11. MD/+/ElectricityPricesRequest:

    Plus sign is replaced with Charging Station's ID (e.g. MD/CS10001/ElectricityPricesRequest). The message contains a string with the number of the day it requests (e.g. "-1" a day before, "0" for current day, "1" for the next day etc.).

12. EI/+/ElectricityImbalanceRequest:

    Plus sign is replaced with Mechanism Design or Charging Station's ID (e.g. EI/CS10001/ElectricityImbalanceRequest). The message contains a string with the number of the day it requests (e.g. "-1" a day before, "0" for current day, "1" for the next day etc.).

13. EV/+/RequestChargingRecommendations:

    Plus sign is replaced with Electric Vehicle's ID (e.g. EV/EV1000/RequestChargingRecommendations). The message contains a list with the EVPreferences and location of the EV (e.g. ('"arrivalConfidence": "mean": 0.037, "std": 0.19, "socConfidence": "mean": 0.78, "std": 0.24, "departureConfidence": "mean": 0.05, "std": 0.23, "desiredSOC": 1.0, "stationNetwork": "net1", "arrival": "2020-01-01T05:00:00", "departure": "2020-01-01T06:00:00", "chargingType": "ratedPower": 120.0, "connectorType": "3 phase-DC", "category": "normal", "totalKWHs": 40.0, "selling_price": 0.2', '"latitude": 2.0, "longitude": 0.0')).

14. CS/+/ReserveChargingSlot:

    Plus sign is replaced with the Charging Station's ID (e.g. CS/CS10001/ReserveChargingSlot). The message contains a list with the Charging Recommendation, the Battery information and the EVPreferences (e.g. [ChargingRecommendation(id='rec3000', arrival=datetime.datetime(2020, 1, 1, 5, 0), departure=datetime.datetime(2020, 1, 1, 6, 0), stationNetwork='net1', connectorType='3 phase-DC', stationID='CS10001', issueDate=datetime.datetime(2020, 1, 1, 4, 0), slotID=0, pricePerKWH=0.2, totalKWHs=40.0, chargingPower=120.0, rank=4.5, location=Location(latitude=4.0, longitude=0.0), evid='EV10012'), Battery(capacity=79.2, soh=0.95, soc=0.5, chargingEfficiency=0.95, consumption=23.4, chargingTypes=[ChargingType(ratedPower=3.0, connectorType='Regular electricity socket 13A', category='level 1'), ... , ChargingType(ratedPower=120.0, connectorType='3 phase-DC', category='level 3')]), EVPreferences(arrivalConfidence=Confidence(mean=0.037, std=0.19), socConfidence=Confidence(mean=0.78, std=0.24), departureConfidence=Confidence(mean=0.05, std=0.23), desiredSOC=1.0, stationNetwork='net1', arrival=datetime.datetime(2020, 1, 1, 5, 0), departure=datetime.datetime(2020, 1, 1, 6, 0), chargingType=ChargingType (ratedPower=120.0, connectorType='3 phase-DC', category='normal'), totalKWHs=40.0, selling_price=0.2)]).

15. CS/+/AuthenticateRecommendation:

    Plus sign is replaced with the Charging Station's ID (e.g. CS/CS10001/AuthenticateRecommendation). The message contains the Recommendation that the Charging Station that needs to verify (e.g. "id": "rec3000", "arrival": "2020-01-01T05:00:00", "departure": "2020-01-01T06:00:00", "stationNetwork": "net1", "connectorType": "3 phase-DC", "stationID": "CS10001", "issueDate": "2020-01-01T04:00:00", "slotID": 0, "pricePerKWH": 0.2, "totalKWHs": 40.0, "chargingPower": 120.0, "rank": 4.5, "location": "latitude": 4.0, "longitude": 0.0, "evid": "EV10012").

16. EI/+/RegistrationOutcome:

Plus sign is replaced with the Electricity Producer, Electricity Consumer or Charging Station's ID (e.g. EI/EP100/RegistrationOutcome). The message contains a string that is "SUCCESS" if the registration completed successfully or "FAIL" if the registration failed.

17. MD/+/RegistrationOutcome:

    Plus sign is replaced with the Electricity Producer, Electricity Consumer or Charging Station's ID (e.g. MD/EP100/RegistrationOutcome). The message contains a string that is "SUCCESS" if the registration completed successfully or "FAIL" if the registration failed.

18. SR/+/RegistrationOutcome:

    Plus sign is replaced with the Charging Station's ID (e.g. SR/CS10001/RegistrationOutcome). The message contains a string that is "SUCCESS" if the registration completed successfully or "FAIL" if the registration failed.

19. EI/+/UpdateProfileOutcome:

    Plus sign is replaced with the Electricity Producer or Electricity Consumer's ID (e.g.EI/EP100/UpdateProfileOutcome). The message contains a string that is "SUCCESS UPDATE" if the update completed successfully or "FAIL UPDATE" if the update failed.

20. MD/+/UpdateProfileOutcome:

    Plus sign is replaced with the Electricity Producer or Electricity Consumer's ID (e.g.MD/EP100/UpdateProfileOutcome). The message contains a string that is "SUCCESS UPDATE" if the update completed successfully or "FAIL UPDATE" if the update failed.

21. EI/+/UpdateConfidenceOutcome:

    Plus sign is replaced with the Electricity Producer, Electricity Consumer or Charging Station's ID (e.g.EI/EP100/UpdateConfidenceOutcome). The message contains a string that is "SUCCESS UPDATE CONFIDENCE" if the confidence update completed successfully or "FAIL UPDATE CONFIDENCE" if the confidence update failed.

22. MD/+/UpdateConfidenceOutcome:

Plus sign is replaced with the Electricity Producer, Electricity Consumer or Charging Station's ID (e.g.MD/EP100/UpdateConfidenceOutcome). The message contains a string that is "SUCCESS UPDATE CONFIDENCE" if the confidence update completed successfully or "FAIL UPDATE CONFIDENCE" if the confidence update failed.

23. EI/+/UpdateScheduleOutcome:

    Plus sign is replaced with the Charging Station's ID (e.g.EI/CS10001/UpdateScheduleOutcome). The message contains a string that is "SUCCESS SCHEDULE UPDATE" if the schedule update completed successfully or "FAIL SCHEDULE UPDATE" if the schedule update failed.

24. MD/+/UpdateScheduleOutcome:

    Plus sign is replaced with the Charging Station's ID (e.g.MD/CS10001/UpdateScheduleOutcome). The message contains a string that is "SUCCESS SCHEDULE UPDATE" if the schedule update completed successfully or "FAIL SCHEDULE UPDATE" if the schedule update failed.

25. CS/+/UpdateAvailabilityOutcome:

    Plus sign is replaced with the Charging Station's ID (e.g.CS/CS10001/UpdateAvailabilityOutcome). The message contains a string that is "SUCCESS SCHEDULE UPDATE" if the schedule update completed successfully or "FAIL SCHEDULE UPDATE" if the schedule update failed.

26. MD/ElectricityPrices or MD/+/ElectricityPrices:

    Plus sign is replaced with the Charging Station's ID (e.g.MD/CS10001/ElectricityPrices) in case there is a request or the Mechanism Design publishes periodically to the topic MD/ElectricityPrices to inform all the subscribed agents. The message contains two lists of TimePrices. The first is the cost for buying of a kWh of energy each timestep in a day and the second is for the selling price (e.g. ['"dateTime": "2020-01-02T00:00:00", "price": 0.50573', '"dateTime": "2020-01-02T01:00:00", "price": 0.18636', ... , '"dateTime": "2020-01-02T23:00:00", "price": 0.28965'], ['"dateTime": "2020-01-02T00:00:00", "price": 0.22', '"dateTime": "2020-01-02T01:00:00", "price": 0.12186', ... , '"dateTime": "2020-01-02T23:00:00", "price": 0.28843']).

27. EI/ElectricityImbalance or EI/+ElectricityImbalance:

    Plus sign is replaced with the Mechanism Design or Charging Station's ID (e.g.EI/CS10001/ElectricityImbalance) in case there is a request or the Electricity Imbalance publishes to the topic EI/ElectricityImbalance periodically to inform all the subscribed agents. The message contains two lists of PowerSlots. The first is the production of energy each timestep for a day and the second list is the conumption (e.g. ['"dateTime": "2020-01-02T00:00:00", "kwh": 144.0', '"dateTime": "2020-01-02T01:00:00", "kwh": 1282.0', ... , '"dateTime": "2020-01-02T23:00:00", "kwh": 818.0'], ['"dateTime": "2020-01-02T00:00:00", "kwh": 504.8', '"dateTime": "2020-01-02T01:00:00", "kwh": 515.3', ... , '"dateTime": "2020-01-02T23:00:00", "kwh": 657.5'])

28. EV/+/ChargingRecommendations:

    Plus sign is replaced with Electric Vehicle's ID (e.g. EV/EV1000/ChargingRecommendations). The message contains a list of Recommendations (e.g. ['"id": "rec3000", "arrival": "2020-01-01T05:00:00", "departure": "2020-01-01T06:00:00", "stationNetwork": "net1", "connectorType": "3 phase-DC", "stationID": "CS10001", "issueDate": "2020-01-01T04:00:00", "slotID": 0, "pricePerKWH": 0.2, "totalKWHs": 40.0, "chargingPower": 120.0, "rank": 4.5, "location": "latitude": 7.0, "longitude": 0.0, "evid": "EV10012"', '"id": "rec3001", "arrival": "2020-01-01T05:00:00", "departure": "2020-01-01T06:00:00", "stationNetwork": "net1", "connectorType": "3 phase-DC", "stationID": "CS10002", "issueDate": "2020-01-01T04:00:00", "slotID": 0, "pricePerKWH": 0.2, "totalKWHs": 40.0, "chargingPower": 120.0, "rank": 4.5, "location": "latitude": 8.0, "longitude": 0.0, "evid": "EV10012"'])

29. EV/+/ReservationOutcome:

    Plus sign is replaced with Electric Vehicle's ID (e.g. EV/EV1000/ReservationOutcome). The message contains the reserved Recommendation, a list of PowerSlots for the charging schedule, a list of TimePrices for buying prices, a list of TimePrices for selling prices and a boolean value to indicate if the reservation completed successfully (e.g. ('"id": "rec3006", "arrival": "2020-01-01T07:00:00", "departure": "2020-01-01T12:00:00", "stationNetwork": "net1", "connectorType": "3 phase-60A per phase", "stationID": "CS10001", "issueDate": "2020-01-01T06:00:00", "slotID": 40, "pricePerKWH": 0.2, "totalKWHs": 27.23, "chargingPower": 43.0,

"rank": 4.5, "location": "latitude": 7.0, "longitude": 0.0, "evid": "EV10012"', '['"dateTime": "2020-01-01T07:00:00", "kwh": 0.0', '"dateTime": "2020-01-01T08:00:00", "kwh": 0.0', '"dateTime": "2020-01-01T09:00:00", "kwh": 0.0', '"dateTime": "2020-01-01T10:00:00", "kwh": 0.0', '"dateTime": "2020-01-01T11:00:00", "kwh": 27.23']', '['"dateTime": "2020-01-01T07:00:00", "price": 0.34306', '"dateTime": "2020-01-01T08:00:00", "price": 0.39411', '"dateTime": "2020-01-01T09:00:00", "price": 0.26966', '"dateTime": "2020-01-01T10:00:00", "price": 0.28723', '"dateTime": "2020-01-01T11:00:00", "price": 0.17283']', '['"dateTime": "2020-01-01T07:00:00", "price": 0.2978', '"dateTime": "2020-01-01T08:00:00", "price": 0.27685', '"dateTime": "2020-01-01T09:00:00", "price": 0.269', '"dateTime": "2020-01-01T10:00:00", "price": 0.28663', '"dateTime": "2020-01-01T11:00:00", "price": 0.109']', True))

30. CS/+/AuthenticateRecommendationOutcome:

    Plus sign is replaced with the Charging Station's ID (e.g. CS/CS10001/AuthenticateRecommendationOutcome). The message contains a Recommendation and boolean value that indicates if the authentication of Recommendation completed successfully (e.g. '('"id": "rec3000", "arrival": "2020-01-01T05:00:00", "departure": "2020-01-01T06:00:00", "stationNetwork": "net1", "connectorType": "3 phase-DC", "stationID": "CS10001", "issueDate": "2020-01-01T04:00:00", "slotID": 0, "pricePerKWH": 0.2, "totalKWHs": 40.0, "chargingPower": 120.0, "rank": 4.5, "location": "latitude": 4.0, "longitude": 0.0, "evid": "EV10012"', True)')

31. CS/+/EV/+/EV_start/start_negotiation:

    The first plus sign is replaced with the Charging Station's ID and the second with the Electric Vehicle's ID (e.g. 'CS/CS10001/EV/EV10001/EV_start/start_negotiation'). The message contains a NegotiationObject (e.g. '"price": 0.99, "arrival": "2020-01-01T07:00:00", "departure": "2020-01-01T12:00:00", "soc": 0.5').

32. CS/+/EV/+/EV_start/accept_negotiation:

    The first plus sign is replaced with the Charging Station's ID and the second with the Electric Vehicle's ID (e.g. 'CS/CS10001/EV/EV10001/EV_start/accept_negotiation'). The message contains the outcome of the negotiation "ACCEPT" or "REJECT".

33. CS/+/EV/+/CS_start/start_negotiation:

The first plus sign is replaced with the Charging Station's ID and the second with the Electric Vehicle's ID (e.g. 'CS/CS10001/EV/EV10001/CS_start/start_negotiation'). The message contains a NegotiationObject (e.g. '"price": 0.99, "arrival": "2020-01-01T07:00:00", "departure": "2020-01-01T12:00:00", "soc": 0.5').

34. CS/+/EV/+/CS_start/accept_negotiation:

    The first plus sign is replaced with the Charging Station's ID and the second with the Electric Vehicle's ID (e.g. 'CS/CS10001/EV/EV10001/CS_start/accept_negotiation'). The message contains the outcome of the negotiation "ACCEPT" or "REJECT".

35. timer:

    The topic timer is responsible to synchronize all the agents for the simulation time. It contains the message "start" and when the agents receive it, start their first day of simulation,

Appendix B

Ontology

In this appendix, we provide information about each concept and attributes that used in our implementation.

EVPreferences : This concept, contains a set of parameters which capture the current desires and needs of the Electric Vehicle driver concerning the charging of his vehicle. These parameters include the expected arrival and departure to and from the charging station expressed as a YYYY-MM-DDTHH:MI:Sec formatted String, the desired SOC at the departure time, the preferred network of charging stations, the preferred charging type from those which the vehicle supports and the confidence to his preferences.

ChargingType : Electric Vehicles can be equipped with one or more inlets which allow different maximum power inputs for their battery to charge. The charging power determines the actual charging time of the battery. Given these, this concept contains a field that denotes the maximum rated power for charging and discharging the battery, the inlet type, and a label to categorize the type charging (e.g. Level 1,2,3).

Battery : This concept describes the battery which is equipped to an Electric Vehicle. It contains the capacity of the battery, which is the maximum energy it can store; the current State of Charge (SOC) of the battery, which is the amount of energy it has currently stored; the State of Health (SOH), which represents the condition of the battery compared to its ideal conditions expressed as percentage; the charging efficiency, which denotes the amount of power that the battery receives compared to the nominal value of the input power; the energy consumption (kwh/100km) and a list of charging types.

ChargingRecommendation : This concept represents a charging recommendation that is made to an Electric Vehicle given its type (i.e. battery state, charging preferences). It contains information about the recommended arrival and departure to and from the charging station, details about the station itself (e.g, station id, charging slot id, location), as well as, the price for charging at a particular station during the period mentioned before, the amount of energy that the vehicle will receive and the rate of charging. Finally, the recommendation system assigns to each recommendation made a rank, that is, the likelihood that the Electric Vehicle will "like" a particular recommendation.

ElectricVehicle : The Electric Vehicle concept contains all these parameters which describe a real-world Electric Vehicle. These parameters include the battery of the vehicle, the charging preferences of the driver, its current location, as well as the model and an identifier to uniquely recognize it.

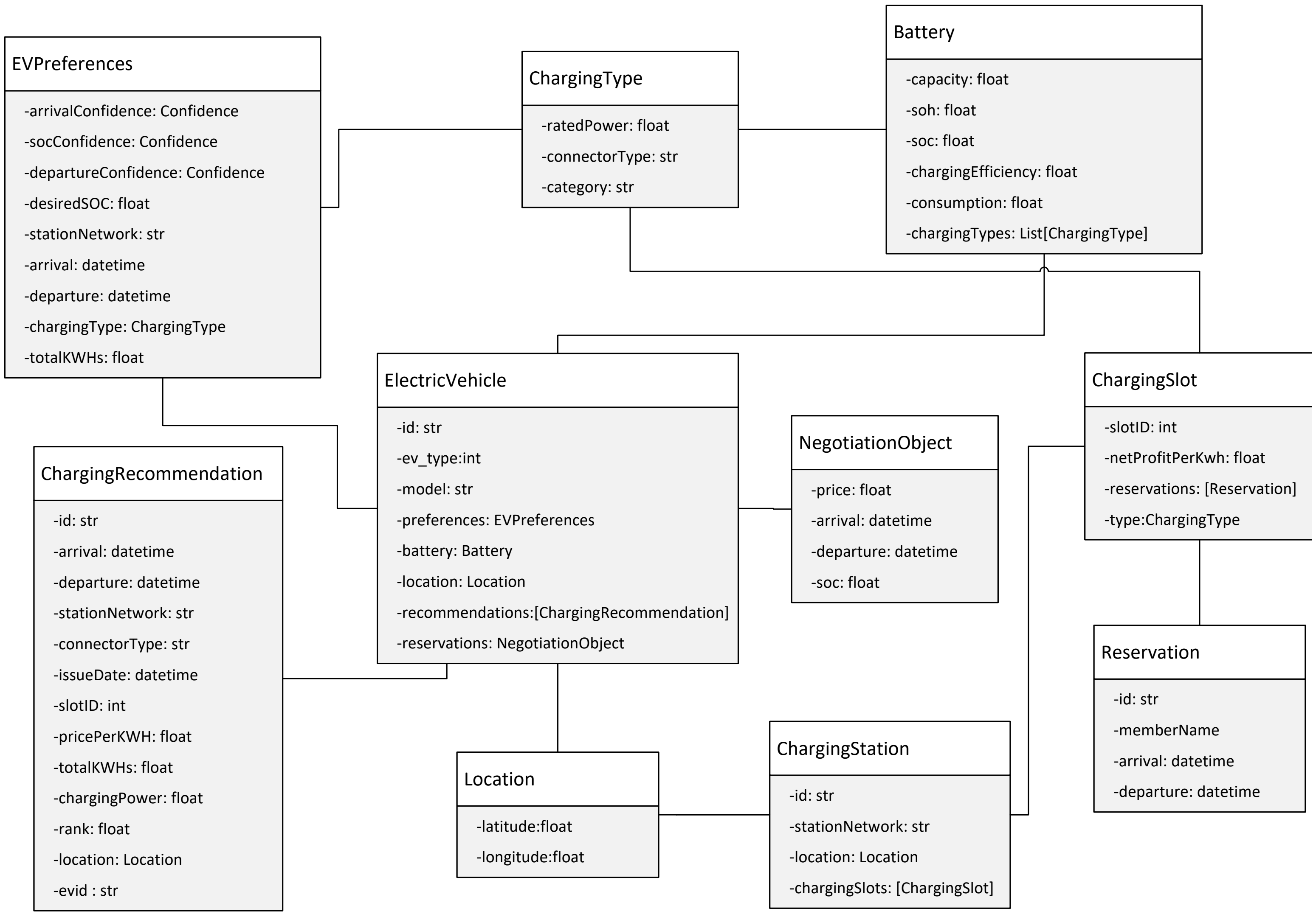
NegotiationObject : In our system, we allow negotiations between Electric Vehicles and charging stations. The negotiation concerns a set of variables that are available for negotiation and are denoted using this object. These variables include the arrival and the departure to and from a particular charging station, the amount of energy that the Electric Vehicle is going to receive, as well as the price charged per energy unit (kWhs).

ChargingSlot : A charging station can contain one or more charging slots. These slots are the ports where the Electric Vehicles connect and derive power to charge their batteries. Each slot is coupled with a parking slot that we assume its in front or near it. At a given charging station, different charging slots can have different payment tariffs for each electricity unit consumed.

Location : This concept contains a set of geographic coordinates (latitude and longitude) that uniquely determine the position of an object in the world.

ChargingStation : This concept describes an Electric Vehicle charging station. A charging station contains the identifier of the station, its location, the network of stations that she belongs, as well as the charging slots available in this particular charging station.

Reservation : This concept contains information about a reservation made to charging slot of a charging station. Its parameters are the period of the reservation expressed as a set of date-time values (YYYY-MM-DDTHH:MI:Sec), as well as a unique reservation identifier.


EVPreferences
-arrivalConfidence: Confidence
-socConfidence: Confidence
-departureConfidence: Confidence
-desiredSOC: float
-stationNetwork: str
-arrival: datetime
-departure: datetime
-chargingType: ChargingType
-totalKWHs: float
ChargingType
-ratedPower: float
-connectorType: str
-category: str
Battery
-capacity: float
-soh: float
-soc: float
-chargingEfficiency: float
-consumption: float
-chargingTypes: List[ChargingType]
ElectricVehicle
-id: str
-ev_type:int
-model: str
-preferences: EVPreferences
-battery: Battery
-location: Location
-recommendations:[ChargingRecommendation]
-reservations: NegotiationObject
ChargingSlot
-slotID: int
-netProfitPerKwh: float
-reservations: [Reservation]
-type:ChargingType
NegotiationObject
-price: float
-arrival: datetime
-departure: datetime
-soc: float
ChargingRecommendation
-id: str
-arrival: datetime
-departure: datetime
-stationNetwork: str
-connectorType: str
-issueDate: datetime
-slotID: int
-pricePerKWH: float
-totalKWHs: float
-chargingPower: float
-rank: float
-location: Location
-evid : str
Reservation
-id: str
-memberName
-arrival: datetime
-departure: datetime
Location
-latitude:float
-longitude:float
ChargingStation
-id: str
-stationNetwork: str
-location: Location
-chargingSlots: [ChargingSlot]

ElectricityProducer : This concept contains all the information which describe an Electricity Producer. It includes the identifier of the producer, the units of measurement of production and its type (e.g. PV Panel, Wind Turbine, VPP).

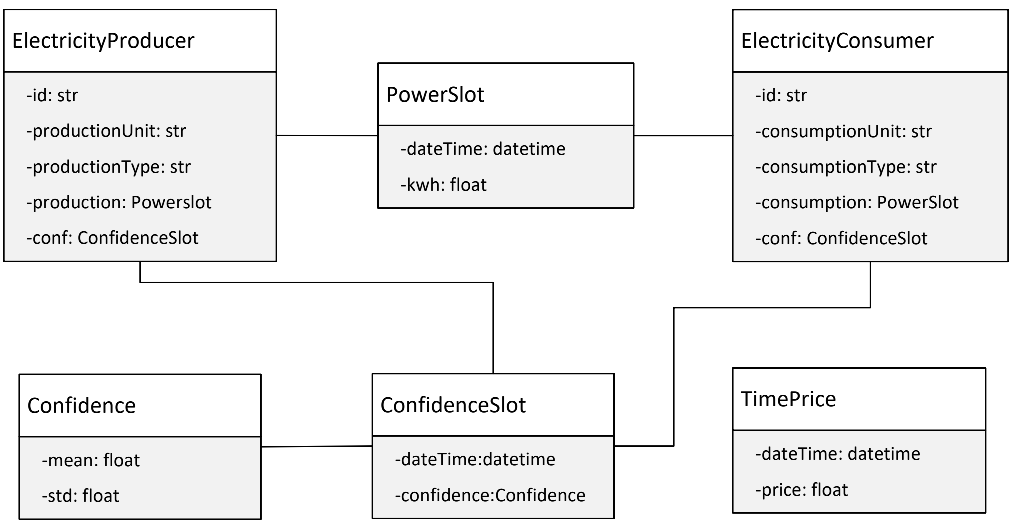
PowerSlot : This concept contains information about the amount of energy which is expected to be available at a specific time interval. It contains a parameter about the amount of energy (kWh) available and a data-time parameter with format YYYY-MM-DDTHH:MI:Sec.

ElectricityConsumer : This concept contains all the information which describe an Electricity Consumer. It includes the identifier of the consumer, the units of measurement consumption (e.g. kWh) and its type (e.g. Residential, Industrial).

Confidence : This is a concept that contains information about the confidence of an entity regarding its expected goals. The confidence is expressed using two values; the mean and standard deviation and indicate the deviation from the reported expected energy profile.

ConfidenceSlot : This concept associates a Confidence object to a specific date and time. It is used to declare our confidence over a specific time interval. Its parameters are a Confidence concept as described above, and a date-time value with format YYYY-MM-DDTHH:MI:Sec.

TimePrice : This concept contains information about the price of each unit of energy available at a specific time interval. Its parameters are a price, and a date-time value with format YYYY-MM-DDTHH:MI:Sec.



References

[1] S. Karapostolakis, E. S. Rigas, N. Bassiliades, and S. D. Ramchurn. "EVLib: A Library for the Management of the Electric Vehicles in the Smart Grid". In: *Proceedings of the 9th Hellenic Conference on Artificial Intelligence, SETN 2016, Thessaloniki, Greece, May 18-20, 2016*. Ed. by N. Bassiliades, A. Bikakis, D. Vrakas, I. P. Vlahavas, and G. A. Vouros. ACM, 2016, 13:1–13:4.

[2] E. S. Rigas, S. Karapostolakis, N. Bassiliades, and S. D. Ramchurn. "EVLibSim: A tool for the simulation of electric vehicles' charging stations using the EVLib library". In: *Simulation Modelling Practice and Theory* 87 (2018), pp. 99–119.

[3] E. L. Karfopoulos and N. D. Hatziargyriou. "A Multi-Agent System for Controlled Charging of a Large Population of Electric Vehicles". In: *IEEE Transactions on Power Systems* 28.2 (May 2013), pp. 1196–1204. ISSN: 0885-8950.

[4] J. Jordán, J. Palanca, E. del Val, V. Julian, and V. Botti. "MASEV: A MAS for the Analysis of Electric Vehicle Charging Stations Location". In: *Advances in Practical Applications of Agents, Multi-Agent Systems, and Complexity: The PAAMS Collection*. Ed. by Yves Demazeau, Bo An, Javier Bajo, and Antonio Fernández-Caballero. Cham: Springer International Publishing, 2018, pp. 326–330. ISBN: 978-3-319-94580-4.

[5] P. Papadopoulos, N. Jenkins, L. M. Cipcigan, I. Grau, and E. Zabala. "Coordination of the Charging of Electric Vehicles Using a Multi-Agent System". In: *IEEE Transactions on Smart Grid* 4.4 (Dec. 2013), pp. 1802–1809. ISSN: 1949-3053.

[6] Nikolaos Spanoudakis and Pavlos Moraitis. "Engineering Ambient Intelligence Systems Using Agent Technology". In: *IEEE Intelligent Systems* 30.3 (2015), pp. 60–67. DOI: 10.1109/MIS.2015.3.

[7] P. Asmus. "Microgrids, Virtual Power Plants and Our Distributed Energy Future". In: *The Electricity Journal* 23.10 (2010), pp. 72–82. ISSN: 1040-6190.

[8] W. Ketter, J. Collins, and P. Reddy. "Power TAC: A competitive economic simulation of the smart grid". In: *Energy Economics* 39 (2013), pp. 262–270. ISSN: 0140-9883.

[9] M. Wooldridge and N. R. Jennings. "Intelligent Agents: Theory and Practice". In: *Knowledge Engineering Review* 10 (1995), pp. 115–152.

[10] S. Kamboj, W. Kempton, and K. S. Decker. "Deploying power grid-integrated electric vehicles as a multi-agent system". In: *10th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2011), Taipei, Taiwan, May 2-6, 2011, Volume 1-3*. 2011, pp. 13–20.

[11] S. D. Ramchurn, P. Vytelingum, A. Rogers, and N. R. Jennings. "Putting the 'smarts' into the smart grid: a grand challenge for artificial intelligence". In: *Commun. ACM* 55.4 (2012), pp. 86–97.

[12] M. J. Burke and J. C. Stephens. "Energy democracy: Goals and policy instruments for sociotechnical transitions". In: *Energy Research & Social Science* 33 (2017). Policy mixes for energy transitions, pp. 35–48. ISSN: 2214-6296.

[13] E. Espe, V. Potdar, and E. Chang. "Prosumer Communities and Relationships in Smart Grids: A Literature Review, Evolution and Future Directions". In: *Energies* 11.10 (2018), p. 2528.

[14] T. D. Huynh, N. R. Jennings, and N. R. Shadbolt. "An integrated trust and reputation model for open multi-agent systems". In: *Autonomous Agents and Multi-Agent Systems* 13.2 (2006), pp. 119–154.

[15] N. Nisan. "Introduction to Mechanism Design (for Computer Scientists)". In: *Algorithmic Game Theory*. Ed. by N. Nisan, T. Roughgarden, E. Tardos, and V. Vazirani. Cambridge University Press, 2007, pp. 209–242.

[16] D. Harel and A. Naamad. "The STATEMATE Semantics of Statecharts". In: *ACM Transactions on Software Engineering and Methodology (TOSEM)* 5.4 (1996), p. 293. ISSN: 1049-331.

[17] Nikolaos Spanoudakis and Pavlos Moraitis. "An Agent Modeling Language Implementing Protocols through Capabilities". In: *2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*. Vol. 2. IEEE Computer Society. 2008, pp. 578–582. DOI: 10.1109/WIIAT.2008.343.

[18] C. Sheppard, A. Jenn, and USDOE. *Grid-Integrated Electric Mobility Model (GEM) v1.0*. Jan. 2021. DOI: 10.11578/dc.20210217.4. URL: https://www.osti.gov//servlets/purl/1765949.

[19] J.-T. Liao, H.-W. Huang, H.-T. Yang, and D. Li. “Decentralized V2G/G2V Scheduling of EV Charging Stations by Considering the Conversion Efficiency of Bidirectional Chargers”. In: *Energies* 14.4 (2021). ISSN: 1996-1073. DOI: 10.3390/en14040962. URL: https://www.mdpi.com/1996-1073/14/4/962.

[20] N. Hossein Motlagh, M. Mohammadrezaei, J. Hunt, and B. Zakeri. “Internet of Things (IoT) and the energy sector”. In: *Energies* 13.2 (2020), p. 494.

[21] M. Noura, M. Atiquzzaman, and M. Gaedke. “Interoperability in internet of things: Taxonomies and open challenges”. In: *Mobile Networks and Applications* 24.3 (2019), pp. 796–809.

[22] C. Akasiadis, V. Pitsilis, and C. D. Spyropoulos. “A Multi-Protocol IoT Platform Based on Open-Source Frameworks”. In: *Sensors* 19.19 (2019). ISSN: 1424-8220. DOI: 10.3390/s19194217. URL: https://www.mdpi.com/1424-8220/19/19/4217.

[23] M. Zhou, J. Yan, and D. Feng. “Digital twin framework and its application to power grid online analysis”. In: *CSEE Journal of Power and Energy Systems* 5.3 (2019), pp. 391–398. DOI: 10.17775/CSEEJPES.2018.01460.

[24] C. Brosinsky, D. Westermann, and R. Krebs. “Recent and prospective developments in power system control centers: Adapting the digital twin technology for application in power system control centers”. In: *2018 IEEE International Energy Conference (ENERGYCON)*. 2018, pp. 1–6.

[25] Willett Kempton, Jasna Tomic, Steven Letendre, Alec Brooks, and Timothy Lipman. “Vehicle-to-grid power: battery, hybrid, and fuel cell vehicles as resources for distributed electric power in California”. In: *UC Davis Research Reports* (2001).

[26] Konstantina Valogianni, Wolfgang Ketter, John Collins, and Dmitry Zhdanov. “Sustainable electric vehicle charging using adaptive pricing”. In: *Production and Operations Management* 29.6 (2020), pp. 1550–1572.

[27] Mihail Mihaylov, Sergio Jurado, Narcís Avellana, Kristof Van Moffaert, Ildefons Magrans de Abril, and Ann Nowé. “NRGcoin: Virtual currency for trading of renewable energy in smart grids”. In: *11th International conference on the European energy market (EEM14)*. IEEE. 2014, pp. 1–6.

[28] Zachary Lee, Daniel Johansson, and Steven H. Low. “ACN-Sim: An Open-Source Simulator for Data-Driven Electric Vehicle Charging Research”. In: *Proceedings of the Tenth ACM International Conference on Future Energy Systems (e-Energy ’19)*. Phoenix, AZ, USA: Association for Computing Machinery, 2019, pp. 411–412. ISBN: 9781450366717. DOI: 10.1145/3307772.3330170. URL: https://doi.org/10.1145/3307772.3330170.

[29] Timothy Lipman, Duncan Callaway, Therese Peffer, and Alexandra von Meier. “Open-Source, Open-Architecture Software Platform for Plug-In Electric Vehicle Smart Charging in California”. In: (2020).

[30] Hossein Shahinzadeh, Jalal Moradi, Gevork B Gharehpetian, Hamed Nafisi, and Mehrdad Abedi. “IoT architecture for smart grids”. In: *2019 International Conference on Protection and Automation of Power System (IPAPS)*. IEEE. 2019, pp. 22–30.

[31] Sebastian Käbisch,Daniel Peintner,Jörg Heuer,Jens Schmutzler, Sven Gröning,Markus Lauterbach. *The OpenV2G Project*. Online; accessed 11 February 2022. URL: http://openv2g.sourceforge.net/.

[32] ElaadNL. *EV Related protocol study v1.1*. Online; accessed 11 February 2022. 2016. URL: https://www.elaad.nl.

[33] Stefan Meisenbacher, Karl Schwenk, Johannes Galenzowski, Simon Waczowicz, Ralf Mikut, and Veit Hagenmeyer. “Smart charging of electric vehicles with cloud-based optimization and a lightweight user interface: A real-world application in the energy lab 2.0: Poster”. In: *Proceedings of the Twelfth ACM International Conference on Future Energy Systems (e-Energy ’21)*. 2021, pp. 284–285.

[34] Claudio Savaglio, Maria Ganzha, Marcin Paprzycki, Costin Bădică, Mirjana Ivanović, and Giancarlo Fortino. “Agent-based Internet of Things: State-of-the-art and research challenges”. In: *Future Generation Computer Systems* 102 (2020), pp. 1038–1053. ISSN: 0167-739X. DOI: https://doi.org/10.1016/j.future.2019.09.016.

[35] Franco Zambonelli. “Key Abstractions for IoT-Oriented Software Engineering”. In: *IEEE Software* 34.1 (2017), pp. 38–45. DOI: 10.1109/MS.2017.3.

[36] Karl Georg Høyer. “The history of alternative fuels in transportation: The case of electric and hybrid cars”. In: *Utilities Policy* 16.2 (2008). Sustainable Energy and Transportation Systems, pp. 63–71. ISSN: 0957-1787. DOI: https://doi.org/10.1016/j.jup.2007.11.001. URL: https://www.sciencedirect.com/science/article/pii/S0957178707000768.

[37] Hannah Ritchie and Max Roser. “Energy”. In: *Our World in Data* (2020). https://ourworldindata.org/energy.

[38] Rupesh Kumar, Ajay Jha, Akhil Damodaran, Deepak Bangwal, and Ashish Dwivedi. “Addressing the challenges to electric vehicle adoption via sharing economy: an Indian perspective”. In: *Management of Environmental Quality: An International Journal* (2020).

[39] Hannah Ritchie and Max Roser. “CO2 and Greenhouse Gas Emissions”. In: *Our World in Data* (2020). https://ourworldindata.org/co2-and-other-greenhouse-gas-emissions.

[40] U.S. Department of Energy. *What Is the Smart Grid?* 2013. URL: https://www.smartgrid.gov/the_smart_grid/.

[41] Dipa Soni and Ashwin Makwana. “A survey on mqtt: a protocol of internet of things (iot)”. In: *International Conference On Telecommunication, Power Analysis And Computing Techniques (ICTPACT-2017)*. Vol. 20. 2017, pp. 173–177.

[42] Stephen Ornes. “Core Concept: The Internet of Things and the explosion of interconnectivity”. In: *Proceedings of the National Academy of Sciences* 113.40 (2016), pp. 11059–11060.

[43] Thorsten Kramp, Rob van Kranenburg, and Sebastian Lange. “Introduction to the Internet of Things”. In: *Enabling Things to Talk*. Springer, Berlin, Heidelberg, 2013, pp. 1–10.

[44] Knud Lasse Lueth. “IoT basics: Getting started with the Internet of Things”. In: *White paper* (2015), pp. 1–9.

[45] Amanda Zantal-Wiener. *What is the internet of things? (and why you should care – a lot)*. Dec. 2019. URL: https://blog.hubspot.com/marketing/internet-of-things-for-marketers.

[46] *Making sense of IOT (internet of things) - the IOT Business Guide*. Apr. 2022. URL: https://www.i-scoop.eu/internet-of-things-iot/.

[47] Sandro Nižetić, Petar Šolić, Diego López-de-Ipiña González-de-Artaza, and Luigi Patrono. "Internet of Things (IoT): Opportunities, issues and challenges towards a smart and sustainable future". In: *Journal of Cleaner Production* 274 (2020), p. 122877. ISSN: 0959-6526. DOI: https://doi.org/10.1016/j.jclepro.2020.122877. URL: https://www.sciencedirect.com/science/article/pii/S095965262032922X.

[48] Giorgos Sfikas, Charilaos Akasiadis, and Evaggelos Spyrou. "Creating a Smart Room using an IoT approach". In: *Proceedings of the Workshop on AI and IoT (AI-IoT), 9th Hellenic Conference on Artificial Intelligence, Thessaloniki, Greece*. 2016, pp. 18–20.

[49] Charilaos Akasiadis, Vassilis Pitsilis, and Constantine D Spyropoulos. "A multi-protocol IoT platform based on open-source frameworks". In: *Sensors* 19.19 (2019), p. 4217.

[50] Yating Liu. "BotSpine-A Generic Simple Development Platform of Smartphones and Sensors or Robotics". PhD thesis. University of Saskatchewan, 2017.

[51] AVSystem. *What is an internet of things platform?* Mar. 2020. URL: https://www.avsystem.com/blog/what-is-internet-of-things-platform/.

[52] Carlos Rodríguez-Domínguez, Kawtar Benghazi, Manuel Noguera, Jose Luis Garrido, María Luisa Rodríguez, and Tomás Ruiz-López. "A Communication Model to Integrate the Request-Response and the Publish-Subscribe Paradigms into Ubiquitous Systems". In: *Sensors* 12.6 (2012), pp. 7648–7668. ISSN: 1424-8220. DOI: 10.3390/s120607648. URL: https://www.mdpi.com/1424-8220/12/6/7648.

[53] Muneer Bani Yassein, Mohammed Q Shatnawi, Shadi Aljwarneh, and Razan Al-Hatmi. "Internet of Things: Survey and open issues of MQTT protocol". In: *2017 international conference on engineering & MIS (ICEMIS)*. Ieee. 2017, pp. 1–6.

[54] Om Prakash Mahela, Mahdi Khosravy, Neeraj Gupta, Baseem Khan, Hassan Haes Alhelou, Rajendra Mahla, Nilesh Patel, and Pierluigi Siano. "Comprehensive overview of multi-agent systems for controlling smart grids". In: *CSEE Journal of Power and Energy Systems* (2020).

[55] Xinghuo Yu and Yusheng Xue. "Smart Grids: A Cyber–Physical Systems Perspective". In: *Proceedings of the IEEE* 104.5 (2016), pp. 1058–1070. DOI: 10.1109/JPROC.2015.2503119.

[56] Anam Kalair, Naeem Abas, Muhammad Shoaib Saleem, Ali Raza Kalair, and Nasrullah Khan. "Role of energy storage systems in energy transition from fossil fuels to renewables". In: *Energy Storage* 3.1 (2021), e135.

[57] A. Haines, R.S. Kovats, D. Campbell-Lendrum, and C. Corvalan. "Climate change and human health: Impacts, vulnerability and public health". In: *Public Health* 120.7 (2006), pp. 585–596. ISSN: 0033-3506. DOI: https://doi.org/10.1016/j.puhe.2006.01.002. URL: https://www.sciencedirect.com/science/article/pii/S0033350606000059.

[58] Omar Ellabban, Haitham Abu-Rub, and Frede Blaabjerg. "Renewable energy resources: Current status, future prospects and their enabling technology". In: *Renewable and Sustainable Energy Reviews* 39 (2014), pp. 748–764. ISSN: 1364-0321. DOI: https://doi.org/10.1016/j.rser.2014.07.113. URL: https://www.sciencedirect.com/science/article/pii/S1364032114005656.

[59] Ramteen Sioshansi and Paul Denholm. "Emissions Impacts and Benefits of Plug-In Hybrid Electric Vehicles and Vehicle-to-Grid Services". In: *Environmental Science & Technology* 43.4 (2009). PMID: 19320180, pp. 1199–1204. DOI: 10.1021/es802324j. eprint: https://doi.org/10.1021/es802324j. URL: https://doi.org/10.1021/es802324j.

[60] Emmanouil S. Rigas Georgios Charalambidis Charilaos Akasiadis and Georgios Chalkiadakis. "A Realistic Dataset Generator for Smart Grid Ecosystems with Electric Vehicles". In: *ACM e-Energy'22*. 2022.

[61] Robin Roche, Fabrice Lauri, Benjamin Blunier, Abdellatif Miraoui, and Abderrafiâa Koukam. "Multi-agent technology for power system control". In: *Power Electronics for Renewable and Distributed Energy Systems*. Springer, 2013, pp. 567–609.

[62] Georgios Chalkiadakis, Valentin Robu, Ramachandra Kota, Alex Rogers, and Nick Jennings. "Cooperatives of distributed energy resources for efficient virtual power plants". In: (2011).

[63] Georgios Chalkiadakis, Charilaos Akasiadis, Nikolaos Savvakis, Theocharis Tsoutsos, Thomas Hoppe, and Frans Coenen. "Providing a scientific arm to renewable energy cooperatives". In: *The role of exergy in energy and the environment*. Springer, 2018, pp. 717–731.

[64] Filippos Christianos and Georgios Chalkiadakis. "Efficient multi-criteria coalition formation using hypergraphs (with application to the v2g problem)". In: *Multi-Agent Systems and Agreement Technologies*. Springer, 2016, pp. 92–108.

[65] Nikolaos Spanoudakis, Charilaos Akasiadis, Georgios Kechagias, and Georgios Chalkiadakis. "An open MAS services architecture for the V2G/G2V problem". In: *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*. 2019, pp. 2198–2200.

[66] Charilaos Akasiadis, Georgios Iatrakis, and Georgios Spanoudakis Nikolaos Chalkiadakis. "An Open MAS/IoT-based Architecture for Large-scale V2G/G2V". In: *Proceedings of 20th International Conference on Practical Applications of Agents and Multi-Agent Systems (PAAMS'22)*. 2022.

[67] Henrik Lund and Willett Kempton. "Integration of renewable energy into the transport and electricity sectors through V2G". In: *Energy Policy* 36.9 (2008), pp. 3578–3587. ISSN: 0301-4215. DOI: https://doi.org/10.1016/j.enpol.2008.06.007. URL: https://www.sciencedirect.com/science/article/pii/S0301421508002838.

[68] Charilaos Akasiadis. "Multiagent demand-side management for real-world energy cooperatives". PhD thesis. Πολυτεχνείο Κρήτης. Σχολή Ηλεκτρονικών Μηχανικών και Μηχανικών Υπολογιστών, 2017.

[69] Nikolaos I Spanoudakis. "Engineering Multi-agent Systems with Statecharts". In: *SN Computer Science* 2.4 (2021), pp. 1–21.

[70] Nikolaos I. Spanoudakis and Pavlos Moraitis. "The ASEME methodology". In: *International Journal of Agent-Oriented Software Engineering* 7.2 (2022), pp. 79–107. DOI: 10.1504/ijaose.2022.122600.