



**POLITECNICO
MILANO 1863**

**SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE**

EXECUTIVE SUMMARY OF THE THESIS

ISO15118-20 - DESIGN AND IMPLEMENTATION OF BIDIRECTIONAL POWER TRANSFER FOR V2X APPLICATIONS

LAUREA MAGISTRALE IN AUTOMATION AND CONTROL ENGINEERING - INGEGNERIA DELL'AUTOMAZIONE
E DEL CONTROLLO

Author: GIOVANNI AZZARÀ

Advisor: PROF. MARCO MAURI

Co-advisor: ENG. GABRIELE MARCHEGANI

Academic year: 2021-2022

1. Introduction

In a world that has been fighting for years against the greenhouse effect, the ozone hole and the massive production of toxic gases from the combustion of oil, the development of green technologies is fundamental. One of these technologies on which in recent years is focusing a lot is the EV (Electric Vehicle). Unfortunately, with the advent and spread of EVs, not only it will increase the energy demand that the network will have to meet but will decrease one of the fundamental values that until now has allowed to intervene promptly on the grid stability since it introduces a delay in the system: the network inertia [4]. The current standard that regulates the operation of EV recharges in a context of V2G (Vehicle-to-Grid), the ISO15118, will be updated shortly normalizing a useful concept to lower the reaction time of the network to possible contingency: the BPT (Bidirectional Power Transfer). The BPT is a system that will allow an electric vehicle connected to the network to interrupt its charging cycle and to reverse the current flow to favour the network; this exchange will be necessary only in cases when the grid is in deficit and requires producers a greater

power. The purpose of this thesis will be to create an EV simulator that is BPT enabled and has a graphical interface for managing parameters in real time during a charging session; this simulator will be useful to S.&h., the company that hosted my internship, to test the new generation of EVSE (EV Supply Equipment) stations that will project in the near future. To do that, it will be modified a simulation software of EV and EVSE, the RISEV2G, complying the constraints imposed by the ISO15118-2 standard, to recognize and manage any requests of BPT; the modified software will be deployed on two micro-controllers that will have to communicate with PLC modules, as required by the standard, which, in turn, will communicate with each other as one will simulate the EV and the other the EVSE. The document will summarize this standard and lower level standard, the IEC61851, it will show the components and explain the choices taken and, finally, will show a practical case of application of the V2G: a V2L Wallbox by S.&h..

2. Electrical grid infrastructure

Energy production chain starts with an electric generation plant like thermal, hydroelectric and so on.



Figure 1: Hydroelectric plant.

Regardless to the power source, when a turbine is rotated it generates an amount of current proportional to its velocity rotation [3] thanks to the alternator. Once the motion energy is transformed into electric, it is time to carry the current to the end user. To do that, there are some entities which intervene to handle this transmission having care of the infrastructures all over the country. This entity, the TSOs (Transfer System Organizations), not only takes care of the systems, but also plays the role of network regulator: if a congestion is detected before causing a blackout, Terna, the Italian TSO, intervene with different regulation method. Traditional method used so far are the FRC, aFRR, mFRR and RR, but Terna is introducing another method called "Fast Reserve" [1]. While the first four work together, one consequently to the others, the last one works in parallel with FRC as an extra help. TSOs distribute energy all over the country but to reach the end user another figure must be presented, the DSO. While TSOs work only with HV (High Voltage) AC current in order to make it easy to transport, DSOs aim is to lower this current to medium or low voltages. Users getting energy may be single person or companies; in this case study, energy reaches the EVSEs (Electric Vehicle Supply Equipments) stations to let EVs (Electric Vehicles) user recharge their cars. During energy production, there is a very important factor that

introduces in the grid a certain delay: network inertia. Inertia is a property of the grid related to the rotational motion of generators: they are connected one to each other through the grid, so when a malfunctioning causes the arrest of one generator, all the others start to slow down because the energy demand remain constant and they have not received the command to improve production yet; thanks to the inertia, generators will not stop immediately and will continue to supply energy for enough time to intervene. Until now everything worked fine, but with the introduction of RES (Renewable Energy Sources) the value of inertia is more and more decreasing due to the high reactivity of DRES (Distributed RES) infrastructures: these systems usually do not have rotary parts so do not introduce inertia in the grid. EVs will help the network to be more reactive: creating aggregations of charging EVs allowed in every moment to supply their stored energy back to the grid, it will be available a big amount of energy that can instantly be induced into the electrical grid to restore the right frequency (50Hz in Europe, 60Hz in North America and Japan). These charging stations might provide a huge quantity of energy to be sold on the MSD ("Mercato dei Servizi per il Dispacciamento", the Italian market for ancillary services) but this is a difficult goal for a single EVSE station; the solution are the UVAMs ("Unità Virtuali Abilitate Miste", translated in "Virtual Aggregated Unit for Charging"), organization of small producers, managed by the "Aggregator", which cooperate to generate the energy target to sell on the MSD.

3. V2G

So far it has been described how electricity is generated and carried to the EVs, now it is time to better explain the concept of bidirectionality that has been introduced in the previous chapter. V2G [2] is the acronym for Vehicle-2-Grid and describe an infrastructure where electric vehicles charge their battery till the grid requires energy. When this happen, EVSEs or EVs, depending on both technologies, command to their inverters to switch the current direction letting it flow from the EV to the grid. This is the updated version of previous method developed to help the electric network to restore its frequency; the older architecture, indeed, did not foresee a

return of energy to the network, on the contrary, initially there was not even a control on that. Initially, the only way not to incur in UFLS (Underfrequency Load Shedding, the disconnection of a portion of the customer load) was to incentivise the population to consume less energy during peak periods and to delay consumption to off-peak bands.

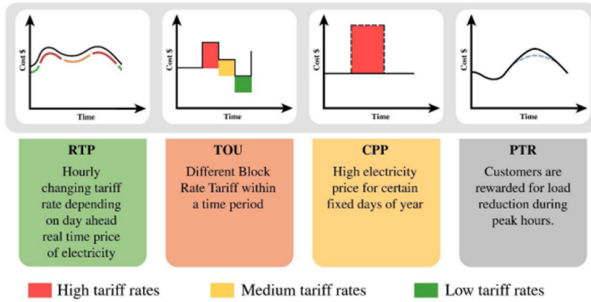


Figure 2: Uncontrolled method to take under control grid stability.

Despite all the incentives, however, this method could not give certainties as it was totally subjugated to the will of users. With the introduction of V1G, the dynamic charging of vehicles, the trend has reversed: the vehicle received only what the network could provide. This means that when an EV was connected to an EVSE, for hypothesis a $22kW$ one, it got the maximum power of $22kW$ since the grid did not detect a congestion; at that point Terna reduces automatically the maximum capacity of current that can be absorbed by the EVSE station where the EV is charging. At this point the EV is being charging with an energy of, for example, $7kWh$; if another vehicle is connected to the same EVSE station the $7kWh$ previously supplied will then be split, supplying $3,5kWh$ both. This solution is for sure very helpful for the grid, but it is only useful to reduce consumes; with the birth of V2G the EV owner becomes power supplier and is remunerated for that. There exist different kind of EVs and different kind of EVSEs, but the working principle is the same: the AC current coming from the grid must be rectified in DC and supplied to battery, as shown in Fig.(3)

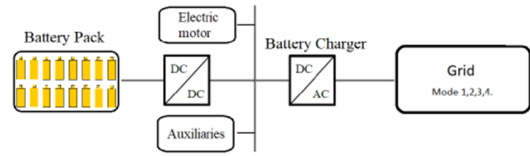


Figure 3: General schema of an EV powertrain for a charging battery.

Depending on the position of the inverters, whether it is mounted on the EV or the EVSE we have different types of one and the other. If inverters are inside the vehicle, the EVSE will provide only AC current and it could be a Mode 1, Mode 2 or Mode 3 and the EV will charge slowly; otherwise, if the inverters are inside the EVSE, it will be a Mode 4 and it could be designed to be very powerful: there exist EVSEs that work at $400kW$ that can charge an EV in just $10mins$. For what concerns EVSE, there are two main types, the column (usually Modes 3 and 4) and the Wallbox (usually Mode 3) and they all implement safety functions [8] such as *circuit breakers*; there are a lot of protocols involved in such big project for both network (TCP/IP, IPV6, UDP, TLS and so on) and serial (HomePlug Green PHY, SPI, CAN BUS ecc.) communication. There are different types of cable connectors (Type1, Type2, CSS, CHAdeMO) and different types of EV (BEV, HEV, PHEV, MHEV FCEV); also for the batteries there exist different typologies made on their chemical composition: Lead.acid, Ni-Cd, Ni-Mh and the most used in this field, Li-ion. All batteries are subject to aging [7] that increases with wear. You might wonder if using a battery inside a V2G infrastructure does not degrade it too quickly. To answer this question, a study was carried out by the Politecnico di Milano, the result of which is shown below.

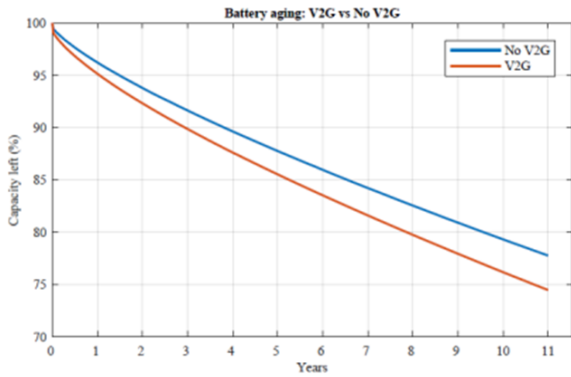


Figure 4: Global aging.

As is evident from Fig.(4), the continuous charge and discharge cycle provided by the V2G deteriorates most batteries, but with a minimum offset; the advantages of this technology are greater than a gap of 4% in 11 years.

4. ISO15118

There are rules that must be followed to create and be able to sell EVSEs; before the birth of V2G, IEC61851 was the standard constructors relied on. It defined an analogical communication between EV and EVSE based on a PWM signal.

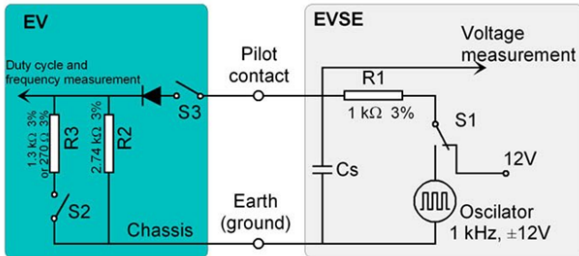


Figure 5: Electrical schema for IEC61851 implementation.

In Fig.(5) 3 switches are shown, S2 and S3 belong to EV, while S1 is mounted in the EVSE. Depending on their position and, so, on the voltage on the CP (Control Pilot) are defined the states of the standard: STATE_A (12V), vehicle not connected, STATE_B (9V), vehicle connected but not ready to charge, and STATE_C (6V), vehicle connected and ready to charge; beyond them there are other states like fault state and cooling request state. With the spread of V2G technology, instead, it is introduced a

digital communication based on software dialog between the EVCC (EV Charge Controller) and the SECC (Supply Equipment Charge Control). V2G structure is defined in the standard "ISO15118 - Vehicle-to-grid communication interface" which is divided in 9 parts; the argument of this thesis is the second one [5], "ISO15118 part 2: Network and application protocol requirement".

8.2.2 Message definition supportedAppProtocolReq and supportedAppProtocolRes
 [V2G20-175] The SECC and EVCC shall implement the message elements as defined in Figure 19.

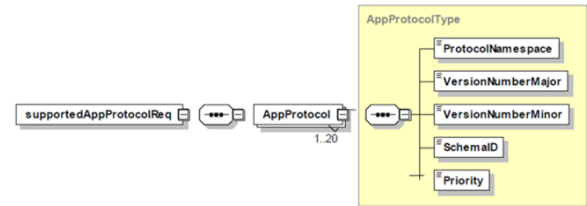


Figure 19 — Schema diagram – supportedAppProtocolReq

Figure 6: Example of how messages are defined: supportedAppProtocolReq.

In this document it is explained how to build the communication [6] between EVCC and SECC, which protocols are implemented (TCP, UDP, TLS, IPV6, EXI, V2GTP and so on), how each message is defined (Fig.(6)) and implemented (Fig.(7)) and are shown states machine both for DC and AC charging, but also for EVCC and SECC.

```
<?xml version="1.0" encoding="UTF-8"?>
<n1:supportedAppProtocolReq xsi:schemaLocation="urn:iso:std:iso:15118-20:AppProtocol:/V2G_CI_AppProtocol.xsd"
xmlns:n1="urn:iso:15118:2:2015:AppProtocol"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <AppProtocol>
    <ProtocolNamespace>urn:iso:15118:2:2015:MsgDef-<ProtocolNamespace>
    <VersionNumberMajor>3-<VersionNumberMajor>
    <VersionNumberMinor>0-<VersionNumberMinor>
    <SchemaID>10-<SchemaID>
    <Priority>1-<Priority>
  </AppProtocol>
  <AppProtocol>
    <ProtocolNamespace>urn:iso:15118:2:2013:MsgDef-<ProtocolNamespace>
    <VersionNumberMajor>2-<VersionNumberMajor>
    <VersionNumberMinor>0-<VersionNumberMinor>
    <SchemaID>20-<SchemaID>
    <Priority>2-<Priority>
  </AppProtocol>
  <AppProtocol>
    <ProtocolNamespace>urn:iso:15118:2:2010:MsgDef-<ProtocolNamespace>
    <VersionNumberMajor>1-<VersionNumberMajor>
    <VersionNumberMinor>0-<VersionNumberMinor>
    <SchemaID>30-<SchemaID>
    <Priority>3-<Priority>
  </AppProtocol>
</n1:supportedAppProtocolReq>
```

Figure 7: Example of information carried by messages: supportedAppProtocolReq.

These messages are exchanged following the corresponding state machine, depending on which charging is chosen by the user; for this project it has been focused on DC charging only, because it has less limitation than AC, as stated in the "Grid Code", the document which details the technical requirements for connecting to and using the National Electricity Transmis-

sion System (NETS). The EVCC sends request type messages, such as supportedAppProtocolReq or PreChargeReq, and the SECC responds with response type messages, such as supportedAppProtocolRes or PreChargeRes; each message carries some information that will be used by the receiver to take decisions on how to move through the states machine. In the following chapter it will be explained how the state machine has been modified to introduce another state for BPT, which function has been added and how an HMI has been built to interact with the simulation of a charging process; moreover, it will be shown how the EV simulator has been designed and configured.

5. Practical work

The aim of this thesis was to build an EV simulator which would implement BPT function to help the company S.&h. to test their V2G/V2L charging columns and Wallboxes; unfortunately this device cannot be implemented as a standalone project, indeed an EVSE simulator enabled to BPT needed to be designed too. To achieve this goal, two RaspberryPi 3B+ have been used, one for the EV (RPIEV) and one for the EVSE (RPIEVSE), in addition to two StampMini2 PLC modules and a software, RISEV2G, for EV/EVSE emulation, modified in order to implement BPT on both. The RPis, running Raspbian OS, have been equipped with an SSH and a VNC server to interact with them without need of two more monitors and peripherals, through both terminal or monitor mode. It has been also enabled the SPI interface to interact with PLC (Power Communication Carrier) modules that, in turn, talk to each other in compliance with IEC61851 standard. Since on both RPIEV and RPIEVSE will run RISEV2G that needs Java8 to work, this package has been installed on the RPis and on the PC where the code development tool Eclipse IDE had been place. Once all connections have been done and the driver to interact with the PLC modules have been added to RPi's driver list, communication has been tested and the circuit has been modified to improve the signal. At this moment the hardware was quite complete, what missed yet was an interface to send command to both RPis; in the first phase of the project it was being build a physical HMI (Human Ma-

chine Interface), with push buttons, potentiometers to regulate battery SOC (State of Charge), a LED strip to show current direction and whatever would be useful.

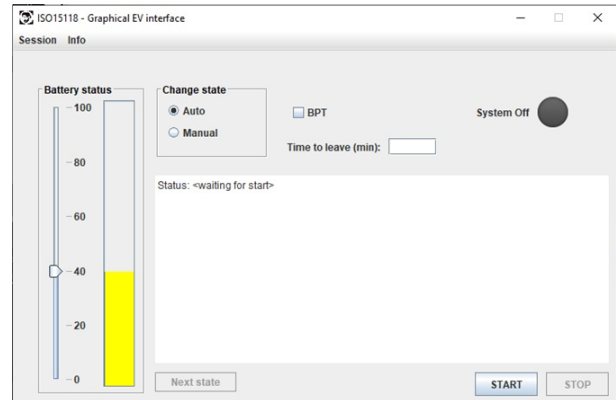


Figure 8: HMI for handling EV charging session build with an Eclipse plugin: WindowsBuilder.

Having problem of compatibility between Java8 and WiringPi (the library used to interact with hardware through GPIO) and being such system too limiting, it has been developed a graphical interface, maybe less easy to implement, but very much powerful; moreover, having a graphical UI (User Interface) opens the door to an internet management of the charging session because it is possible to exploit it to create a web app, very convenient in order to dismiss screens and peripherals otherwise connected to the device.

Now that hardware is ready, the software modification (ModRISE) needs to take place to add BPT as RISEV2G function: an extra state has been added to the state machine of both EVCC and SECC, called respectively WaitForBptRes and WaitForBptReq. These states have been added principally with a didactic scope to underline that particular situation during simulations, but in a future development it could be maintained the modified code bypassing this state and saving time during execution.

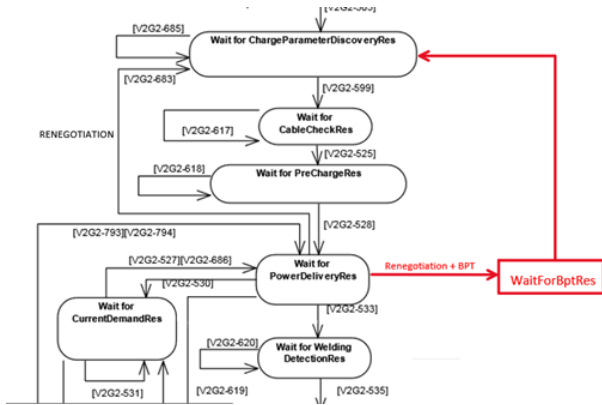


Figure 9: Modified state machine of standard ISO15118-2: EVCC side, DC charging.

In addition to new states, it has been created also the messages exchanged by these additional states: BptRes and BptReq. Each of them brings information that may not be used in this version of the program, but they have been thought as a guide for those who will work on it in the future.

What happens when the BPT state is triggered is that a "Renegotiation" of all parameters is made: in the ChargeParameterDiscovery state, all positive value of current, that were defined during program initialization, will become negative values depending on the hardware this system will be mount on and this will be interpreted by the inverter through its control logic. With ModRISE it is possible to emulate a normal charging session, but also enable or disable BPT in any time through the HMI; in same way it is possible to emulate battery SOC and see how the program reacts to this changing value beyond setting the TimeToLeave parameter which tells the software how much time is left before the EV leaves the station. These parameters are interpreted from the logic of the ModRISE that has been implemented with this project and decisions are taken: if the EVCC understands that EV is discharging, through an hysteresis cycle it sets a limit to 20% under which the SOC will not drop; if instead it understands that the vehicle becomes enable to BPT but it is charging, before enabling a BPT session the SOC must reach at least 80%. The time parameter is important too because even if the veicle is enabled to BPT and its SOC value allows it, if the vehicle has less than 60mins it will not let the vehicle to discharge in favor of the grid.

6. Conclusions

A simpler implementation of this V2G project is the V2L Wallbox by S.&h., a device based on IEC61851 which allows to recharge any load compatible with EV battery capacity.

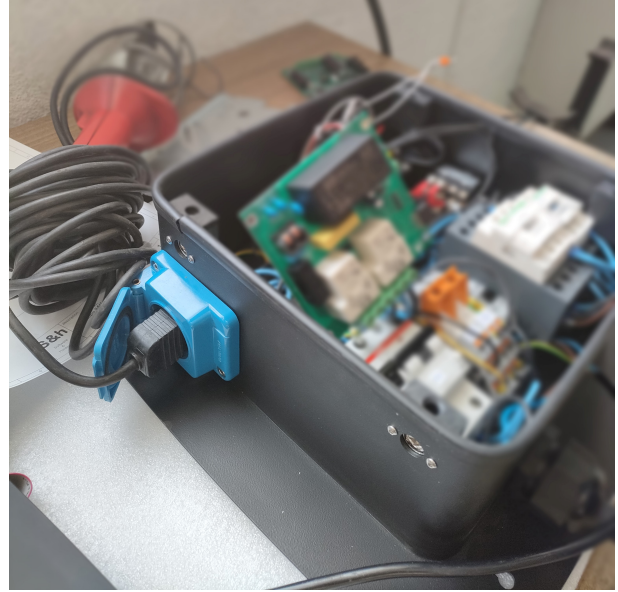


Figure 10: Wallbox by S.&h.: here it can be seen a load (a 60W lamp) directly connected to the device.

To test this product it has been used an EV, still not on the market at the time of the thesis, which was enabled to V2L and, consequently, to BPT. It is here that this project result useful: the company needed an EV that allowed bidirectionality, but they are rare by now and reserve them even for half a day it is quite complicated; having a simulator at own disposal every time needed will be a great opportunity.

7. Acknowledgements

The author would like to thank S.&h., the company which hosted the internship, provided all needed hardware and knowledge, the head of the company, Eng. Alberto Crivellaro, and the whole staff. Thanks to Eng. Gabriele Marchegiani and Eng. Marco Mauri, for proposing this interesting opportunity. In conclusion, thanks to Eng. Attilio Borri who has been fundamental for RISEV2G comprehension and implementation.

References

- [1] Fast reserve - information pack. Technical report, Terna S.p.A.
- [2] Adil Amin, Wajahat Ullah Khan Tareen, Muhammad Usman, Haider Ali, Inam Bari, Ben Horan, Saad Mekhilef, Muhammad Asif, Saeed Ahmed, and Anzar Mahmood. A review of optimal charging strategy for electric vehicles under dynamic pricing schemes in the distribution charging network. *Sustainability*, 12(23):10160, 2020.
- [3] Maria Stefania Carmeli, Mattia Rossi, and Mauri Marco. *Macchine elettriche Modelli a regime: teoria ed esercizi*. Società Editrice Esculapio, 2020.
- [4] Paul Denholm, Trieu Mai, Richard Wallace Kenyon, Benjamin Kroposki, and Mark O'Malley. Inertia and the power grid: A guide without the spin. Technical report, National Renewable Energy Lab.(NREL), Golden, CO (United States), 2020.
- [5] ISO/IEC. Iso/iec dis 15118-2: Road vehicles - vehicle to grid communication interface – part 2: Network and application protocol requirements, 2012.
- [6] Sergejus Martinenas, C Træholt, PB Andersen, and M Marinelli. Enabling technologies for smart grid integration and interoperability of electric vehicles. *Technical University of Denmark, Department of Electrical Engineering*, 2017.
- [7] M Mauri, F Castelli-Dezza, MS Carmeli, M Scarforghiero, and G Marchegiani. Electro-thermal aging model of li-ion batteries for vehicle-to-grid services. In *2019 AEIT International Conference of Electrical and Electronic Technologies for Automotive (AEIT AUTOMOTIVE)*, pages 1–6. IEEE, 2019.
- [8] Jean-François REY. Safety measures for electric vehicle charging. Technical report, University of Zurich, Department of Informatics, 01.



POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

ISO15118-20 – DESIGN AND IMPLEMENTATION OF BIDIRECTIONAL POWER TRANSFER FOR V2X APPLICATIONS

Tesi di Laurea Magistrale in
Ingegneria dell'Automazione e del Controllo

Author: **Giovanni Azzarà**

Student ID: 953924

Advisor: Prof. Marco Mauri

Co-advisors: Eng. Gabriele Marchegiani, Eng. Alberto Crivellaro, Eng.
Marco Aversa

Academic Year: 2021-22

Abstract

In a world that has been fighting for years against the greenhouse effect, the ozone hole and the massive production of toxic gases from the combustion of oil, the development of green technologies is fundamental. One of these technologies on which in recent years is focusing a lot is the EV (Electric Vehicle). Unfortunately, with the advent and spread of EVs, not only it will increase the energy demand that the network will have to meet but will decrease one of the fundamental values that until now has allowed to intervene promptly on the grid stability since it introduces a delay in the system: the network inertia. The current standard that regulates the operation of EV recharges in a context of V2G (Vehicle-to-Grid), the ISO15118, will be updated shortly normalizing a useful concept to lower the reaction time of the network to possible contingency: the BPT (Bidirectional Power Transfer). The BPT is a system that will allow an electric vehicle connected to the network to interrupt its charging cycle and to reverse the current flow to favour the network; this exchange will be necessary only in cases the grid is in deficit and requires producers a greater power. The purpose of this thesis will be to create an EV simulator that is BPT enabled and has a graphical interface for managing parameters in real time during a charging session; this simulator will be useful to S.&h., the company that hosted my internship, to test the new generation of EVSE (EV Supply Equipment) stations that will project in the near future. To do that, it will be modified a simulation software of EV and EVSE, the RISEV2G, complying the constraints imposed by the ISO15118-2 standard, to recognize and manage any requests of BPT; the modified software will be deployed on two micro-controllers that will have to communicate with PLC modules, as required by the standard, which, in turn, will communicate with each other as one will simulate the EV and the other the EVSE. The document will summarize this standard and lower level standard, the IEC61851, it will show the components and explain the choices taken and, finally, will show a practical case of application of the V2G: a V2L Wallbox by S.&h..

Keywords: V2X, V2G, Bidirectionality, BPT, Battery, Electric grid, ISO15118, IEC61851

Abstract in lingua italiana

In un mondo da anni in lotta contro l'effetto serra, il buco nell'ozono e la massiccia produzione di gas tossici derivati dalla combustione del petrolio, lo sviluppo di tecnologie green risulta fondamentale. Una di queste tecnologie su cui in questi anni si sta puntando molto è l'auto elettrica. Con l'avvento e la diffusione degli EV (Electric Vehicle, veicoli elettrici), non solo aumenterà la richiesta di energia che la rete dovrà soddisfare, ma diminuirà uno dei valori fondamentali che fino ad oggi ha consentito di intervenire tempestivamente sulla stabilità poiché introduce un ritardo nel sistema: l'inerzia di rete. L'attuale norma che regola il funzionamento delle ricariche EV in un contesto di V2G (Vehicle-to-Grid), la ISO15118, verrà aggiornata a breve normalizzando un concetto utile per abbassare i tempi di reazione della rete ad eventuali contingenze: il BPT (Bidirectional Power Transfer). Il BPT sarà un sistema che consentirà ad un veicolo elettrico collegato alla rete di interrompere il suo ciclo di ricarica e di invertire il flusso di corrente per favorire la rete; questo scambio sarà necessario solamente nei casi in cui la rete elettrica sarà in deficit e richiederà ai produttori una maggiore potenza. Lo scopo di questa tesi sarà quello di creare un simulatore di veicolo elettrico che sia abilitato al BPT e che abbia un'interfaccia grafica per la gestione dei parametri in tempo reale durante una sessione di carica; questo simulatore sarà utile ad S.&h., la società che ha ospitato il mio tirocinio, per testare la nuova generazione di stazioni di EVSE (EV Supply Equipment) che progetteranno nel prossimo futuro. Per fare ciò verrà modificato un software di simulazione di EV ed EVSE, RISEV2G, rispettando i vincoli imposti dalla norma ISO15118-2, affinché riconosca eventuali richieste di BPT e le gestisca; il software modificato verrà caricato su due microcontrollori che dovranno comunicare con dei moduli PLC, come previsto dalla norma, che a loro volta comunicheranno tra loro in quanto uno simulerà l'EV e l'altro l'EVSE. Nel documento verrà riassunta questa norma e quella di più basso livello, la IEC61851, verranno mostrati i componenti e spiegate le scelte prese e, infine, verrà mostrato un caso pratico di applicazione del V2G: una wallbox V2L.

Parole chiave: V2X, V2G, Bidirezionalità, BPT, Batteria, Rete elettrica, ISO15118, IEC61851

Contents

Abstract	i
Abstract in lingua italiana	iii
Contents	v
Introduction	1
1 Network services and MSD	5
1.1 Energy production chain	5
1.2 Inertia and frequency in grid stability	8
1.3 MSD and Terna	11
1.4 UVAM	14
2 Smart charging	17
3 V2G: main entities of the infrastructure	21
3.1 EV	22
3.1.1 Existing technologies	23
3.1.2 Battery	25
3.2 EVSE	31
3.2.1 Charging modes and time	31
3.2.2 Type of connector	35
3.2.3 Protocols involved	37
3.2.4 Safety measures	38
4 Standards overview	45
4.1 IEC61851	45
4.2 ISO15118	47

5	Practical work	55
5.1	Hardware	55
5.2	Connection and environment	58
5.3	Communication test	62
5.4	Software	63
5.4.1	RISEV2G	64
5.4.2	OpenV2G	67
5.5	HMI	69
5.6	Software design and implementation	72
6	Conclusions and future developments	79
	Bibliography	83
A	Appendix A	87
B	Appendix B	93
	List of Figures	97
	List of Symbols	101
	Acknowledgements	103

Introduction

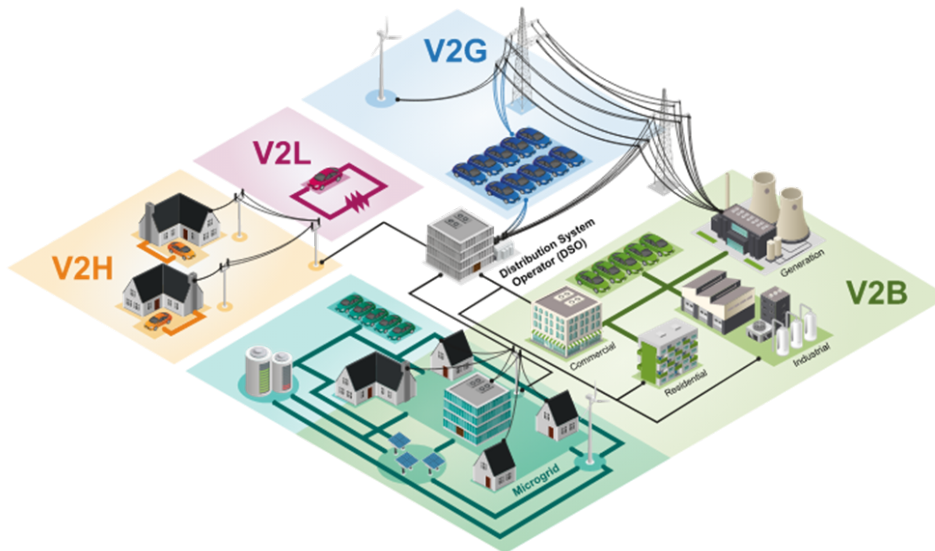
Last 20 years has been a period of strong technological growth. In 2000 was born the Bluetooth, in 2001 Wikipedia, the largest encyclopaedia in the world, and in the years between 2003 and 2005 were created Skype video chat, social networks with Facebook, YouTube videos and Google Maps. 2007 gave rise to the invention of the century, the smartphone with its innovative touchscreen, but also to the digital book; Bitcoin was born in 2009, the first prototype of autonomous driving in 2012 and then, all the communication networks that we still use nowadays like 3G, 4G and 5G.

Today it is possible to watch movies without going to the cinema thanks to streaming, save photos without printing them thanks to the clouds, teaching to machines and move within a universe made entirely of lines of code thanks to Meta, Mark Zuckerberg's Matrix. Nowadays, in 2022, it is possible to make a payment by credit card, without any credit card, with our face or fingerprint. The key word of this historical period is "Share": in these years were born concepts of car sharing, bike sharing and, in particular, *power sharing*.

In 1800 Alessandro Volta gave life to the battery, today more than two centuries after, there is a wide range of electric vehicles with different characteristics, with which it is possible to travel hundreds of kilometres at full load and without the need to refuel, and it is also possible to charge the batteries at a low cost and in an acceptable time. All improvements and the attention to that world lead to an electric car market growth: there are more and more people who choose electric instead of combustion cars and, as a result, there are more and more people who tap into the power grid with cars that have an increasing power demand. Fortunately, this development goes hand in hand with that of DER (Distributed Energy Resources), a distributed system made of small producers who are paid to cooperate providing small amount of energy each. This energy can be generated in any way, but it is easy that a small producer will dispose only of renewable energy like solar and wind energy, energy of piezoelectric materials, energy of hydroelectric plants and so on. The usage in everyday life of RES (Renewable Energy Sources) makes the total production capacity increase constantly but introduces an important problem:

the more RES are integrated in the electric grid, the more network inertia will decrease and it becomes difficult to intervene in case of contingencies. Moreover, if the growth trend of EVs (Electric Vehicles) market was to remain the current one, or increases, the energy produced would not be enough to satisfy the whole population.

The Japanese experts of the sector found a solution between 2009 and 2012, the V2X infrastructure (in the figure below). The acronym V2X stands for "Vehicle-to-X" where the "X" means *everything*; as the name might suggest, the V2X is a protocol that allows any device which implements these rules to require and obtain, if allowed, power from the electric vehicles in charge.



V2X is like a macrofamily within which we find sub-groups of environments that are addressed in detail such as, for example, the V2H (Vehicle-to-Home), the V2B (Vehicle-to-Building), the V2L (Vehicle-to-Load) and the core of this thesis, the V2G (Vehicle-to-Grid). Since this concept of sharing energy was born, it also raised the necessity to regulate these processes, so the International Organization for Standardization (ISO) wrote a document with that purpose: it was 2014 when ISO15118 was born.

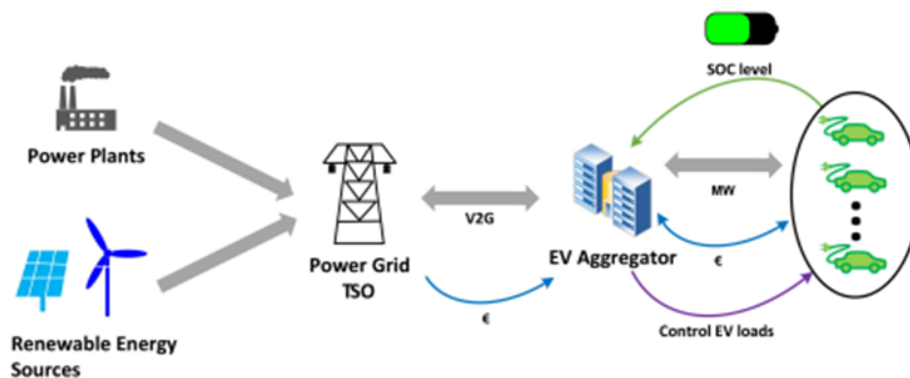
This paper is divided in parts each of which describes certain aspects of this communication: physical and data link, network and application protocol requirements, how to execute tests and so on. There are 9 parts, but what this thesis is interested in is the second one: "ISO15118-2 – Network and application protocol requirements".

To quote the standard [12]:

«This part of ISO 15118 specifies the communication between battery electric vehicles (BEV) or plug-in hybrid electric vehicles (PHEV) and the Electric Vehicle Supply Equipment. The application layer message set defined in this part of ISO 15118 is designed to support the energy transfer from an EVSE to an EV. [...] The purpose of this part of ISO 15118 is to detail the communication between an EV (BEV or a PHEV) and an EVSE. Aspects are specified to detect a vehicle in a communication network and enable an Internet Protocol (IP) based communication between EVCC and SECC.»



In the schema above, only communication (1) will be treated in this document while communication (2) is left to other standards. In the quote previously mentioned, there was talk about charging by the EVSE towards the EV, but, in the first part of the ISO15118, a very interesting concept is introduced: the Bidirectional Power Transfer (BPT). Although this is the basic concept of the V2X, there is no more than a briefly introduction to it in that official document, but to regulate its use and implementation a first version is already being drafted; it will take the name of “ISO15118-20” and it will be an update of the part 2 adding not only the concept of BPT, but also wireless and bus charging.



The arguments presented in this thesis follow as far as possible the flow energy production shown in the figure above and it will be divided as follow:

- **Chapter 1** In this section it will be explained how energy is generated, transformed and delivered to the end users, explaining the role of producers, TSOs, DSOs and

Aggregators; moreover, it will be explained the importance of inertia and frequency in grid stabilization and will be presented different way to react to contingencies. In the end, it will be shown what are the MSD and a UVAM and which is their relationship;

- **Chapter 2** In this section it will be explained how grid was regulated so far and the evolution its regulators are undergoing, showing the differences between “Uncontrolled charging”, V1G and V2G;
- **Chapter 3** In this section it will be shown what are EVs and EVSEs; for what concerns EVs it will be shown different typologies of vehicle and batteries briefly analysing their aging model. About the EVSE, they will be explained in detail showing all existing types depending on their charging mode and connectors; moreover, there will be an overview on protocol involved and security measures;
- **Chapter 4** In this section it will be briefly explained the IEC61851 standard and, more in details, ISO15118-2;
- **Chapter 5** This section will talk about the practical work made; it will be described the hardware used, the software to deploy on it, RISEV2G, and how it has been modified to implement BPT;
- **Chapter 6** This final chapter will summarize what has been done and will show both a real application (Wallbox V2L) and future developments.

Since this project is intended as an access point to this new technology, not as the whole implementation, the practical part will leave different opened ways to future developments.

1 | Network services and MSD

As stated before, let us begin this document starting from the top of the power supply daisy chain; in the following pages it will be explained how electricity is produced, sold and distributed, what is the role of an Aggregator, who is responsible for its delivery, distribution and stability maintenance and how this system is stabilized after a damage or an excessive demand for power.

1.1. Energy production chain

The electricity market consists in four distinct entities [24] that are shown in Fig.(1.1); at the beginning of the chain, energy is generated by exploiting fossil or renewable sources and sold under a free market regime. This energy produced and sold must be transported from the source to the distributor and, finally, it reaches the end user.

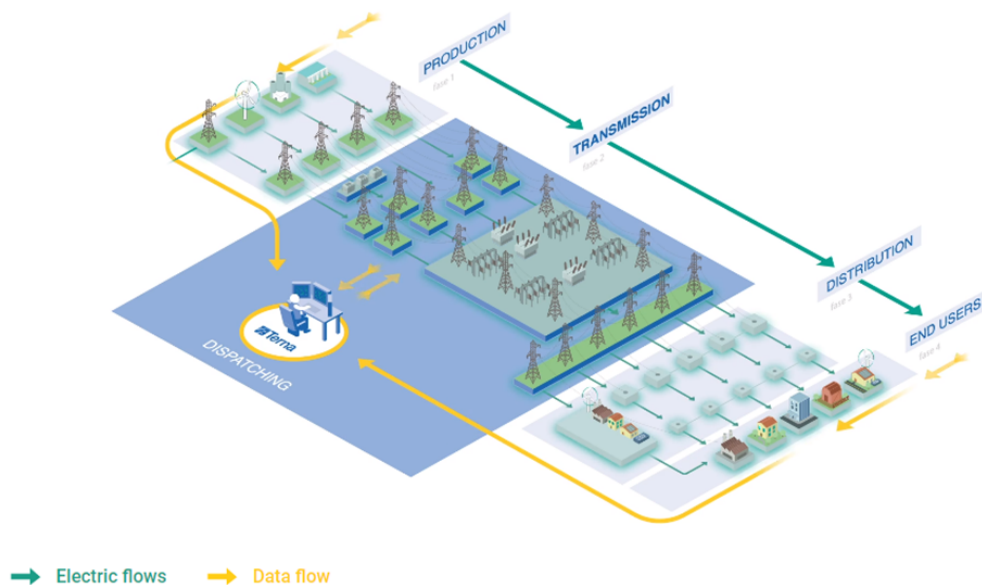


Figure 1.1: Energy production and distribution chain with focus on Terna's position.

Production [24] As the reader might know, electricity cannot be created, but transformed from primary sources. It is produced in power stations (thermoelectric, hydroelectric, wind power, geothermal solar, nuclear). Pressurized steam is the most used method and consists of pressurized water heated to extremely high temperatures (even over 600°) using the primary energy source; the heated steam hits a turbine that makes an alternator rotate. This process is common in turbo-gas, nuclear, thermodynamic solar, geothermal electricity, wind power and hydroelectric production, while only photovoltaic and the use of hydrogen in combustion cells do not involve rotating parts. In Italy this process still largely takes place using non-renewable sources (natural gas, coal and oil). To provide an idea about quantities, in December 2019, electricity demand was 25,600GWh, down from the same month of the previous year (-3.0%). In particular, there was a decrease in thermoelectric production (-12.1%), the foreign balance (-4.5%) and an increase in wind production (+28.2%) and hydroelectric production (+23.8%) compared to the same month of the previous year. In 2019, the cumulative value of electricity demand (319,597GWh) was in line (-0.6%) with respect to 2018. In December 2019, the demand for electricity was met for 49% of production from non-renewable energy sources, 38% from renewable energy sources and the remaining share from the foreign balance. In 2019, the demand for electricity was 319,597GWh and was met at 53% by production from Non-renewable Energy Sources, 35% from renewable Energy Sources and the remaining share from the foreign balance. Most of the electricity that is imported through the 25 interconnections with foreign countries (about 80%) comes from France and Switzerland. This role of producer, in Italy, is played by companies like "Enel" (15,8%), "Eni" (8,0%), "Edison" (7,0%), "A2A" (6,0%), EPH (5,3%) and Iren (3,6%), which produce their energy and generally relay on other companies to the transmission phase; the only TSO in Italy is Terna, which holds the monopoly.

Transmission [24] A TSO (Transmission System Operator) is an entity operating independently by other electricity market players and are responsible for energy transmission: essentially, they connect the producer to the distributor carrying high voltage through the territory. As just said, in Italy the only TSO is Terna which owns 74442Km of power lines and 431 transformer and switching stations over whole Italy. The main roles of a TSO are: monitoring of electricity flows, arrangements for managing the coordinated operation of all the elements of the system, programming of grid unavailability and forecasting the national electricity requirement and comparing it consistently with production programmes resulting from the free energy market. The task of transmitting electricity in Italy is undertaken by Terna through a complex and structured system made up of various elements, the most important of which are: the EHV (extra-high voltage), transformers that receive

and transform the energy produced by the national electrical power stations (or by the border stations for imported energy); the EHV and HV (high voltage) electricity lines that transport energy; and, lastly, the transforming stations that transform HV to low and medium voltage. Moreover, since electricity cannot be stored, the quantity required by families, businesses and transport must be produced when necessary; a TSO is also tasked with managing the energy transmission so that supply and demand are always balanced, continuously and safely. This energy produced is then delivered to distribution companies, which carry electricity to houses and factories through sales companies. This complex system of machines needs to be coordinated and managed by dispatching specialists.

Distribution [7, 24] A DSO (Distribution System Organization) deals with energy distribution, sometimes being also the owner of the infrastructures for its transport, operating at low (LV), medium (MV) and, in some cases, high voltage levels. These high and extreme high voltages are carried to the outskirts of large cities or industrial zones to be lowered and distributed to all end-users. The DSOs, as for TSOs, deliver current through lines made both by over-head and underground cables; traditionally, this concept of line was thought to be unidirectional and based on predictable, controllable and centralized power generation. Nowadays things are changing due to an increasing in locally generated power directly connected to distribution networks: thinking about solar panels they introduce in the grid an amount of voltage which vary from solar panels mounted on the roofs to small power plants. This concept of self energy production is called DER (Distributed Energy Resources) and if energy comes from renewable sources it takes the name of DRES (Distributed Renewable Energy Sources). Since 2007, the UE has subscribed the so called “20-20-20 target” where it committed by 2020 to reduce by 20% its greenhouse gas emission (compared to 1990), produce 20% of energy consumed using RES and will have consumed 20% less energy. To achieve this result, an ever-increasing share of RES is being connected to our electricity networks, as is being connected new forms of energy demand, such as EVs. It is at this point of the power production and delivery chain that a dramatic re-thinking of the infrastructures is required; simultaneously to network extension and reinforcement, complementary IT solutions has been introduced, adding communication, sensors and automation allowing DSOs to actively manage the varying generation and demand. This combination of solutions is what is commonly referred to as “Smart Grid” and, since its birth, it is regulated by these organizations. In Italy, the distribution network management is entrusted to 135 DSOs; among them, “E-Distribuzione” covers the majority of energy demand while the rest goes to “A2A”, “ACEA”, “IREN”, “DEVAL” and “HERA” which are the most important. To have an idea

about quantities in energy distribution, according to previsions for 2023: the agriculture sector will require an amount of $6100000MWh$, the industrial sector request would be around $134900000MWh$, the tertiary sector estimated a value around $133700000MWh$ while the domestic sector may undergo an increment from $69456600MWh$ (in 2012) to $75300000MWh$.

End user The concept of end user is as very general: it is a costumer who buys energy to provide some service or to use it in his or her private life; it is wrong to think about it as a single physical person, indeed, it could be both a person and a company.

1.2. Inertia and frequency in grid stability

RES got more and more space in the market over years and it is a commendable result, but it leads to a decrement of inertia network which in turn reduces the time available to react to contingencies. To better understand this concept, it is worth to spend some words on the working principle of energy production and how, nowadays, the grid is controlled and stabilized.

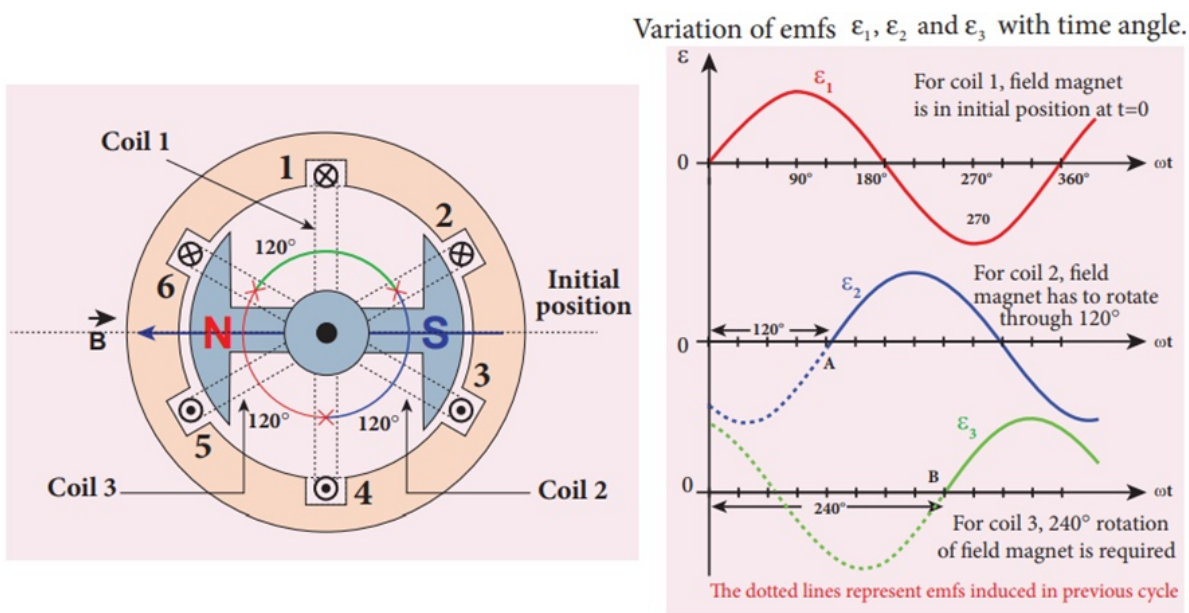


Figure 1.2: Working principle of an AC generator.

The basic idea [4], regardless of the source of energy, is to transform movement into electricity. In Fig. (1.2) it is reported the electric schema of an AC generator; the operating principle described below is common to all electrical machines, what differs is the output that can be in DC or AC depending on the construction characteristics.

Everything revolves around the principle that when a spiral of conductive material is crossed by a variable magnetic field B , an induced current begins to circulate inside it, proportionally to the magnetic flux variation. Since this flux depends proportionally on the surface of the coil exposed to B , as the exposed area increases, the absolute value of the flux increases and with it the current module; when the surface decreases, the induced current also decreases. In Fig.(1.2) are shown 3 coils (coil 1, coil 2 and coil 3), out of phase of 120° one from the others, which constitute the stationary part of the machine, the stator, and an internal magnet that will be set in motion generally by a turbine, the rotor. The rotor, which can be a magnet or an electromagnet, creates a magnetic field that crosses it from South (S) to North (N); in the situation shown in the figure, taking into account only coils 1 and 2, the magnetic field of the rotor is perpendicular to coil 1 and this means that the flux through it is maximum, but at the same time it also passes through the other coils that will be less exposed because of the geometry. By moving the rotor *clockwise*, the magnetic field will rotate along with it and the portion of flux that passes through Turn 1 will begin to decrease along with the current; simultaneously, however, the flow through Coil 2 will increase causing an increase in current at its core, while the current module will decrease to zero and then increase. These continuous flux variation affect all coils at the same time, each of which will be subject in every instant of time to a variation of flow that will be equal to that of the coil before, but out of phase in the time of $+120^\circ$. The result of all this process will be three AC voltages with the same phase shift.



Figure 1.3: Power generator in a hydroelectric plant.

To make the rotor move, it is possible to exploit, for examples, rivers kinetic energy or the pressurized water inside pipes; for each energy source there are proper processes and architectures, but, although it is very interesting, that is not the real goal of this document. For simplicity, it will be considered a hydroelectric plant focusing on the importance of having or not a dynamic part inside the system. In a hydroelectric system [8], a flow of water is conveyed into charge basins through adduction and diversion channels and is directed through penstocks toward hydroelectric turbines. The rotating electric generator (called alternator) is directly connected to the turbine and turns the mechanical energy into electrical energy. If, for any reason, no more water flows in tunnels there will be no power to transfer to the turbine, so it tends to stop. Before it happens, there is a period during which the turbine is still rotating even if no active force is acting on it: this phenomenon is well known as inertia. Inertia is the tendency of an object in motion to remain in motion and, applying this to spinning objects, like the power generator we just saw, we run into the concept of “rotational inertia”. Inertia, together with frequency, is an important factor to consider in the power grid. It gives an indication about the kinetic energy stored in hundreds or thousands of spinning generators that are synchronized at the same frequency ($50Hz$ in Europe, $60Hz$ in North America and Japan); moreover, as for someone who is riding a bicycle, it is a useful property whether operating a power grid: in the first case, inertia gives the rider a chance to stop pedalling and coast without falling over, while, in the grid, it gives the system operator the time to respond to power plant failures (called contingencies). While inertia provides sufficient time to operate changes to restore the grid, frequency is used as an indicator of significant changes in either supply or demand and allows the TSO to take compensating actions. During normal grid operation, the supply of power from all the generators equals the demand for electricity and the frequency remains constant. But just as a vehicle slows when foot is taken off the throttle, if there is a loss of a power plant, the supply of power will drop almost instantaneously. However, the demand for electricity has not changed, so the same amount of power will be extracted from the system, slowing down the other generators. As said before, thanks to inertia there is a short period when the mechanical systems in the grid can detect the imbalance (as reflected in declining frequency) and tell power plants to speed up (or slow down). This is only one way of restoring the electrical network, however to maintain grid frequency and avoid UFLS (underfrequency load shedding, the disconnection of a portion of the customer load), grid operators use a variety of processes that can respond to events that might change frequency all based on “Energy Reserves” as FCR or Fast Reserve (they will be described in the following chapter, but to have an initial overview it can be seen Fig.(1.4) and Fig.(1.6)). What should be clear so far is that inertia is a fundamental grid property to avoid UFLS, but what has not been said yet

is that its value depends on grid size: inertia, indeed, increases proportionally with grid size (larger grids have inherently more inertia). Unfortunately, not every producer who connect its plant to the grid introduces inertia to the whole system; it should be noticed, in fact, that DRES as solar panel system are not affected by this property because during the process of transformation from solar to electrical energy, there are no rotating parts that would introduce it. This leads to a constant decrement of inertia in the whole grid. Historically, the combination of traditional inertia (from both generators and loads) plus FCR has been sufficient to address contingency events in most of the United States. But as the grid evolves with the addition of V2G and other new technologies, system planners and operators are deploying new ways to maintain stable frequency even with declining amounts of conventional inertia.

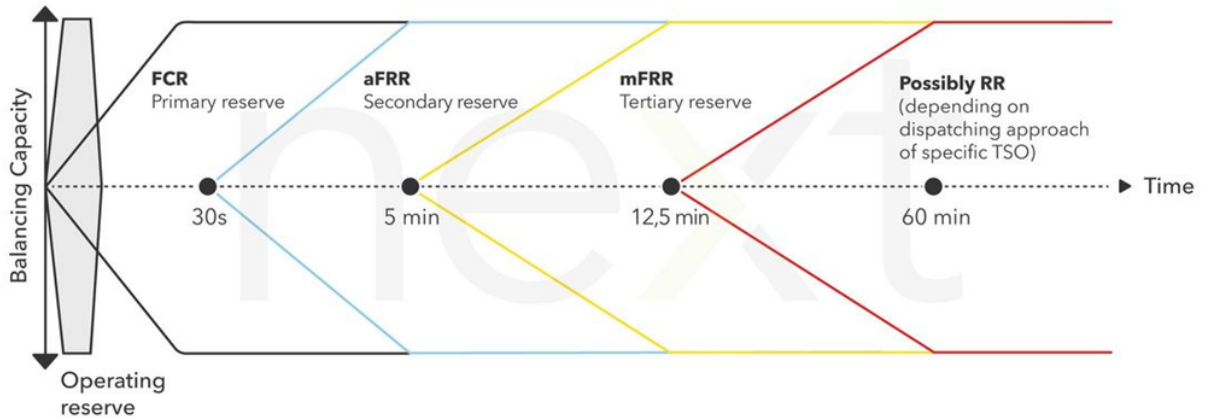


Figure 1.4: Terna reaction to out of range frequency values.

1.3. MSD and Terna

The MSD (in Italian “Mercato dei Servizi di Dispacciamento”, translated into “Ancillary Services Market”) is the Italian market that supplies the resources necessary for the management and control of the system; thanks to MSD it is possible to ensure a stable electricity supply system through the provision of power reserves. It consists of a scheduling substage (ex-ante MSD) and Balancing Market (MB): in the ex-ante MSD, Terna accepts energy demand bids and supply offers in order to relieve residual contingencies and to create reserve margins, while in the MB, Terna accepts energy demand bids and supply offers in order to provide its service of secondary control and to balance energy injections and withdrawals into/from the grid in real time. The TSO uses these resources to balance any losses in the energy grid. The trading of power reserves takes place in four modes: the FCR (Frequency Controlled Reserve), the aFRR (Automatic Frequency

Restoration Reserve), the mFRR (manual Frequency Restoration Reserve) and the RR (Replacement Reserve).

- The FCR mode, also known as PFR (“Primary Frequency Reserve”) or “Primary Control Reserve”, is the most lucrative, but also the fastest to react to frequency changing and, due to this, the most required on the technical level: the complete activation of this service must be available within 30s and it has 15mins to restore stability, but just after half a minute the second countermeasure, the aFRR, comes into play. FCR detects changes in frequency and automatically, without action from the system operator, adjusts generators energy supply to maintain frequency within the desired range (49.5Hz and 50.5Hz, also shown in Fig.(1.5)). After a contingency event, FCR acts to increase power from the remaining generators and (temporarily) replace energy from the failed generator; to ensure the grid stability in the Continental Europe synchronous area, an amount of 3000 MW positive and negative primary control reserve is constantly ready for activation. This amount represents the cumulative power of the two largest power plants in the grid area. Only a generator that can increase output (and sustain that output for a period) can provide PFR;



Figure 1.5: Snapshot of the frequency plot taken online in the moment in which this part was written.

- The aFRR, in Italy still known as "Secondary Reserve ", intervenes if FCR does not bring any benefit to the grid in the first 30s. It is slightly slower and must provide the reserve within 5mins, but if the grid imbalance persists after 12.5minutes, the mFRR supports or replaces aFRR gradually. In order to ensure aFRR, the installation with power reserves shall be connected to a remote operating unit located in

the TSO control station to exchange commands and data. At present, only conventional power stations provide a market, and there is no such market yet; actually, they are working on a project called “Platform for the International Coordination of Automated Frequency Restoration and Stable System Operation” (PICASSO [9]) which involves TSOs from 30 European countries, including Italy, with the purpose to design, implement and operate a unique aFRR-Platform, but also to enhance economic and technical efficiency and integrate the European aFRR markets;

- The mFRR, also known as “Tertiary Reserve”, intervenes when the other upstream balancing services (FCR or aFRR) were not sufficient to resolve deviations in the power grid. This resource must be fully deployable after $12.5mins$ and has a minimum duration period of $5mins$. What European countries have in common is that this procedure is the first countermeasure that actually requires an operator to start the process, but its implementation is different for each of them.
- The RR, known as "other services", is the last option to contrast a frequency variation not only regulates the frequency, but also handles any contingency. Also in this case, the power provided shall be fully available $15mins$ after the activation control, which shall not, however, be transmitted directly and automatically to the system control.

In the last two years Terna proposed a new reaction method to contingency events, the “Fast Reserve” [1]; this new energy reserve, acting together with FCR, works faster (as can be seen in Fig.(1.6)) than the primary reserve and comes into play 1s after the detection of unusual frequency fluctuation. Fast Reserve is a service provided by the so called FRUs (Fast Reserve Unit) which can be composed both of stand-alone and aggregated devices. The FRU guarantees a minimum of 5MW up to a maximum of 25MW, an activation time of 1s, a minimum up time of 30s, a release time of $5mins$ and an energy capacity sufficient to provide the contracted capacity for at least $15mins$. Whoever can participate to Fast Reserve, a dispatching user (UdP), a device owner or an aggregator: the provider must issue a guarantee equal to 5000 €/MW by 10 days before the auction and one to ensure the availability of the FRU according to the following formula:

$$Guarantee[€] = 0.25 * ContractedCapacity[MW] * ContractedPrice[€/MW]$$

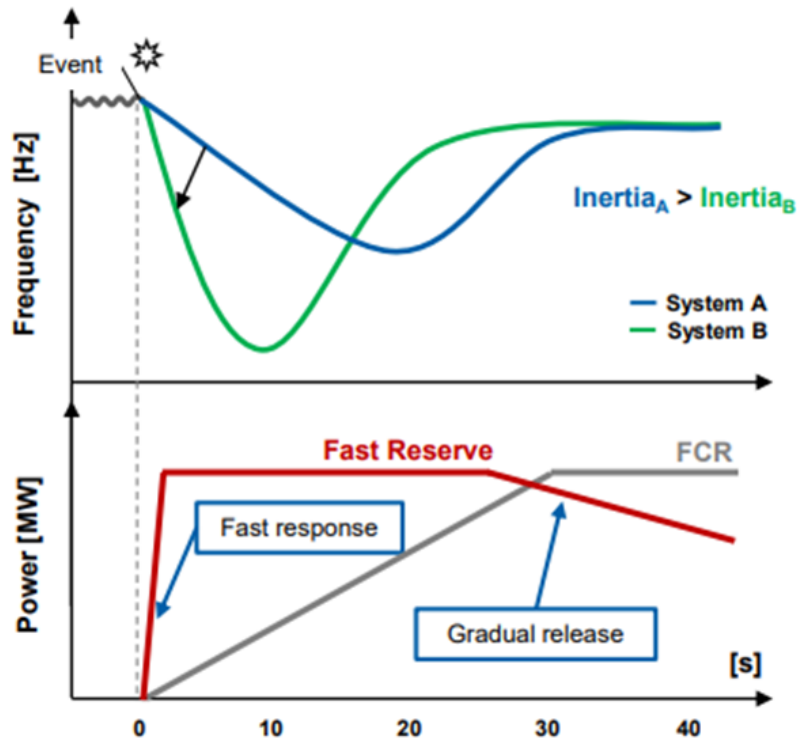


Figure 1.6: Comparison between the Fast Reserve and FCR resource.

1.4. UVAM

In Italy, to have access to the MSD you must be a big energy supplier or consumer (over 10MVA); if a small producer, as a private citizen, desires to sell his or her energy it is possible, but this person must be part of a group of small producers like him or her: an UVAM. The UVAM [18] (“Unità Virtuale Abilitate Miste”, translated “Virtually Aggregated Mixed Units”) is an entity which collects small amount of energy, regardless for the sources, from small producers to get the possibility of entering the market. Basically, it is an aggregation of peripheral units of production, consumption, and accumulation. Thanks to the decentralised approach, energy stored by this entity can be used as power reserve when network stability problems arise. In order to obtain the maximum possible regional distribution, TERNA has therefore established 15 aggregation zones within which the participants in the UVAM project must offer at least 1 MW of positive or negative reserve power.



Figure 1.7: Charging station with 7 Mode 4 EVSEs.

Another limitation to be accepted in a UVAM is that the installed capacity of the electricity production and storage facilities shall not exceed 10 MW, beyond the fact that the units consuming electricity shall be equipped with a load management system and/or take emergency measures; of course, participating units must not be already inside the MSD. It is in this scenario that EVs and V2G come into play. With V2G introduction, each EV turns into a producer since it stores an important quantity of energy that can be supplied to the grid in every moment; unfortunately, according to what has been said before about prerequisites, a single EV cannot be considered an energy producer and cannot be integrated in an UVAM, but in case of an EV Charging Station as the one shown in Fig.(1.7) the energy stored will be enough the station owner to join the group. If a UVAM integrate one or more charging station it is called UVAR (“Unità Virtuale di Aggregazione di Ricarica”, translated “Virtual Aggregated Unit for Charging”). Just to have a better understanding about quantities involved, considering an electric vehicle with an average capacity of $71kW$, computed taking as extremes a Tesla Model S with a capacity of $100kW$ and a Fiat 500e with a capacity of $42kW$, it would be needed 15 vehicles fully charged to satisfy the constrain of a minimum supply of 1MW. What is described here and in the whole document, the role of an EV user in the power production chain, has to be intended as a service that each EV owner subscribes with the owner of UVAM: the “Aggregator”; these entities are responsible for the relationship with the TSO, and they take responsibility for accomplish to their duty as part in the MSD.



Figure 1.8: Italian UVAM owned by Terna.

In Italy, for example, Ego is an aggregator, one of the first participants in an UVAM project, which can provide more than 100MW of modular power; most of the energy produced comes from: a cogeneration plant that supplies the district heating service of Morbegno (± 14 MW), a plant managed by the hydroelectric company Boschetto, another co-generator that feeds the production cycle of an industrial site and waste treatment plants available to disconnect loads, then to move consumption, in certain periods and/or hours of the day to meet the needs of Terna. Obviously behind such an organization there is an economic compensation for those who join the UVAM project, who earn three times: from the sale of reserve power on the MSD, for the consideration of capacity and the connection to the digital future of Italian energy supply. Reserve power providers shall receive the price in euro per *MWh* for their activation on the MSD. In case of activation for supply of positive reserve power, they receive the consideration directly, otherwise costs must be incurred which will be compensated by the profit made on the market the day before and the cost savings for fuel (in the case of cogeneration plants fired by oil, coal or gas). In addition, UVAM participants receive capacity fees from an auction procedure with an initial base of 52,500€ for each MW/year made available in the 15:00 - 22:00 time slot from Monday to Friday.

2 | Smart charging

As just shown in previous chapter, the owner of an electric charging station that meets the requirements can join a UAVR, interacting with the TSO to help manage the power in the grid. The problem here is that being owner of an EVSE station is not enough: indeed, it is necessary that this station is able not only to charge vehicles, but, most important, to discharge it. To do so, there are software, hardware and economical constraints that must be satisfied; all these rules go under the name of “Smart Charging”. Smart charging can be defined as a new concept that allows you to change the charging profile of an EV based on network and market conditions, taking autonomous decisions thanks to the logic control. The main functioning is very simple: each vehicle in the charging station can be recharged by the EVSE which provide energy based on EV specifics and on grid availability. Till the electrical network works fine, it does not require any intervention and every EVs keep charging normally; but when Terna detects an anomaly in the network frequency, it asks to the MSD for energy reserves. If an EVSE station is part of the UVAR it is obligated to provide the amount of energy it has been subscribed in the contract between Terna and the UVAR itself. What happen next depends on how the aggregator though the UVAR to handle this request, so, in particular, how the installed software works. This depends on software implementation, but it is also strictly connected to its architecture: while most home chargers offer basic on/off control, advanced chargers may provide more sophisticated controls which provide benefits for multiple stakeholders. Moreover, when an aggregator designs the station, it must consider which services will be provided or required to understand whether to build a unidirectional or bidirectional information and energy system. What is important to have understood once got here is that “Smart Charging” could be intended as another way of grid stabilization with the same purpose of FCR, aFRR and so on. Although there is not a common definition yet, it is possible to say that Smart Charging identifies 5 different architectures: Uncontrolled smart charging, Basic control, V1G and V2X depending on information and power direction, and on playing entities; what follows will be a general description on what they are, and which are the differences among them. [23]

Uncontrolled with Time-of-Use optimization [3] There are multiple uncontrolled systems thought to encourage customers to not overload the electrical grid and, just to see an example, let us just present the Time-of-Use method.

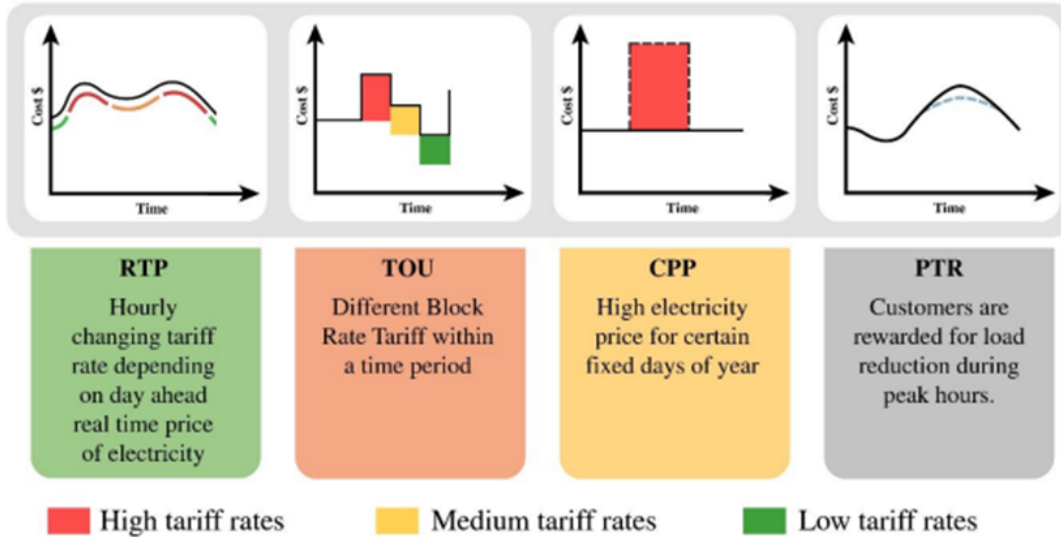


Figure 2.1: Comparison between different uncontrolled methods.

The Time-of-Use (ToU) can be defined as the time block rates of electricity and it consists on assigning a certain predefined price to each hour of the day based on historical conditions rather than current load curve; more in general, it is usually based on three time periods according to load: off peak (capacity greater than demand, low ToU cost), mid peak (capacity and demand very close, moderate ToU cost), and full peak (high electric load, high ToU cost). This way, if an EV user needs to recharge its vehicle, he, or she, will choose an off peak period, if possible; this is the simplest method to build up a smart charging system because it does not require any control from external stakeholders. Below experiment conducted by Mckinsey in the US shows that mid-night time-of-use smart charging optimization reduced peak grid demand close to 50%. (from +30% peak demand to +16%). Other uncontrolled method are shown and briefly described in Fig.(2.1).

V1G V1G is a monidirectional charging system where energy and information only flow from the TSO to the EV. In a V1G structure, the EV only receives the amount of energy that the electrical network can provide in that moment. This means that if Terna detects an imminent contingency, it communicates to the Charging Point Operator (CPO) its energy need; the CPO itself adjusts EVSE power limitation to start providing less energy. This will give to the grid the opportunity to restore its nominal frequency value without adding external energy from power reserves shown in Chapter 1.3.

V2X V2X (Vehicle-to-everything) services, including V2G, offers the same principle as V1G both in charging and discharging mode; its aim is to generate revenue from the battery asset through dynamic and bi-directional charge control. It has been already said that V2X it is a macro-group describing all those systems which take energy out of the EVs; this macro-group can be divided in sub-groups: V2H (Vehicle to House), V2L (Vehicle to Load), V2B (Vehicle to Buildings) and V2G (Vehicle to Grid). V2H and V2B do not typically directly affect grid performances because they generally interact with the environment providing energy to it and not to the grid: looking at an example, if the grid is suffering due to an exceeding request of energy, a building that is part of a V2B system can be disconnected from the grid and one or more EVs will supply energy instead of electrical network. V2H and V2B do not typically directly affect grid performances because they generally interact with the environment providing energy to it and not to the grid: looking at an example, if the grid is suffering due to an exceeding request of energy, a building that is part of a V2B system can be disconnected from the grid, but still energized by one or more EVs. In practice, EVs are used as residential back-up power supply. Unlike V1G solutions that were able on the market since years, V2X has not yet reached market deployment, only in Japan where commercial V2H solutions have been available since 2012 to handle electricity blackout.

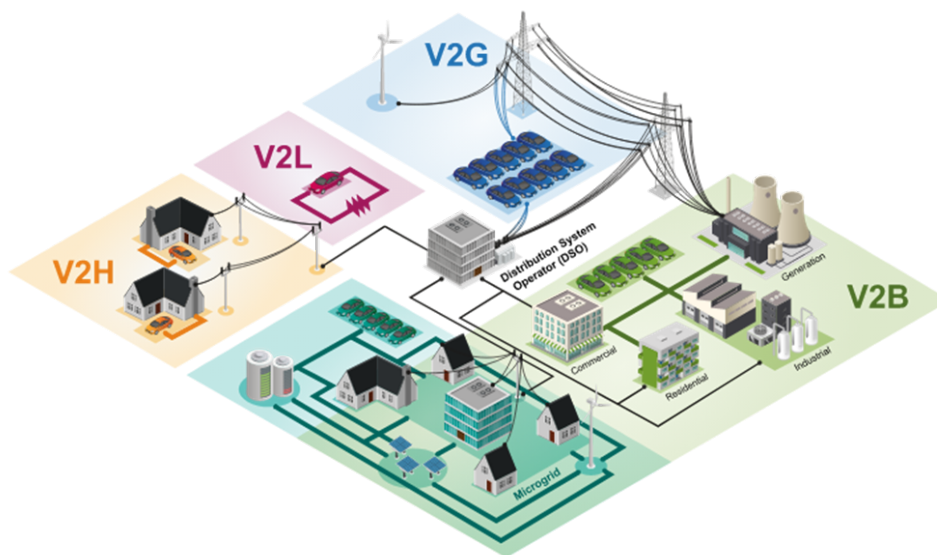


Figure 2.2: Caption of the Figure to appear in the List of Figures.

While V2B and V2H are generally not directly connected to the grid, it is different for V2G which purpose is exactly to interact with it. V2G is considered a fundamental infrastructure by grid operators, since the EV battery offers the same functionality as a stationary battery for ancillary services such as FCR, aFRR, mFRR and RR.[23]

3 | V2G: main entities of the infrastructure

In the previous chapters it has been explained where energy comes from, here it will be seen how it is used by the main actors of the V2G architecture: EVs (Electric vehicles) and EVSEs (EV Supply Equipments). Although there exists different types of both, they all rely on a simple process: charge a battery supplying current; in the case study, of course, instead of only one battery it must be charged a whole battery pack, so an important amount of energy is required. This energy is extracted directly from the grid (and controlled by the charging cable or the car itself) or through an EVSE which adjusts energy flowing from the grid and may take some decision about the charging process, depending on how that type of EVSE has been designed.

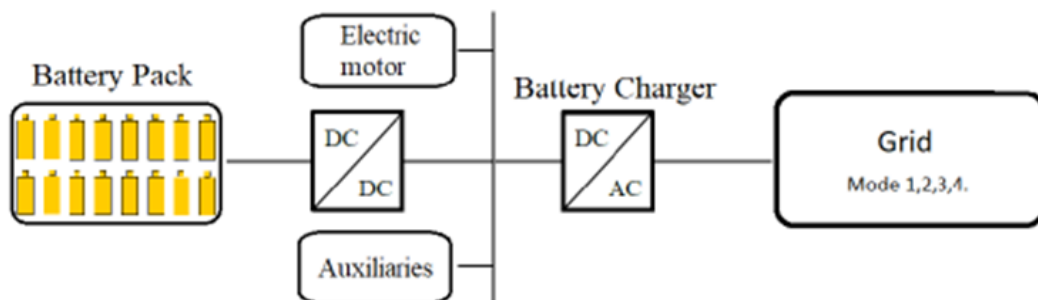


Figure 3.1: Here are shown the EV connected to the EVSE; it is not possible to define which block belong to which entity because it changes depending on the type of both.

As the reader might know, there are two types of current, AC and DC, and, as shown in Fig.(3.1), only the latter can be used to charge a battery. Unfortunately, the current we usually have at our disposal, taken from our homes or outdoors, is of the AC type. This means that somewhere in the infrastructure in Fig.(3.1), the alternating current will have to be rectified and this is done by usually two inverters: the first, the upstream, straightens the voltage making it continuous, while the second takes care of adjusting

the current to the desired value. In the schema above, there are no limits that defines where finish the EV and where starts the EVSE because there is not an actual standard limit; it depends on how each single vehicle and each single EVSE is designed based on manufacturer choices. In this chapter there will be presented the main actor of this architecture, the EV and the EVSE, showing all the existing types and explaining their functioning.

3.1. EV

[5] The reader might be familiar with the concept of combustion engine that since 1899 has been employed to build vehicles to drive on the roads, but on that date the electric vehicle was almost 30 years old: an “Electric Vehicle”, also known as EV, is a vehicle that uses one or more electric motors powered by on board batteries for propulsion. In 1800 Alessandro Volta gave life to the battery, but it took 67 years, in 1867, during the Universal Exhibition in Paris, for the first electric vehicle was presented. The evolution led to the invention of an electric tricycle in 1881 and, a few months later, to the experimentation of a bus.



Figure 3.2: First experimental electric bus.

In 1897, in London and New York, the first electric taxis began to run and, to follow, the first competitions were run, authorized and not. The evolution of these electric vehicles

took more and more foot, to the point that, in the first decade of the '900, about a third of the vehicles circulating in the United States was electric; it was such a great phenomenon that Thomas Edison himself, in 1901, studied how to improve the performance of their batteries. If the fact that this technology was already available at the beginning of the '900 seems strange, it is because the reader does not know that it in the same period, around 1901, was born one of the first hybrid cars, invented by the engineer Ferdinand Porsche, the founder of the eponymous car manufacturer. Unfortunately, in the first half of the twentieth century, this sector suffered a setback, and the causes are manifold: road networks began to improve, and greater autonomy was required to make the movements; moreover, in those years gasoline began to cost less than batteries in addition to the fact that cars with combustion engines were easier to drive. This technology remained dormant for many years, until the 1960s, when the price of gasoline began to rise again and the interest in electricity was rekindled. On July 20, 1969, the Moon was first seen trampled by a human foot, but it was in 1971 that the crew of the Apollo 15 used an electric rover for exploration, and this only fuelled the interest that gasoline had rekindled. From here on, electric cars begin to emerge for the city and companies that specialize in this sector; 2003 gave rise to the company Tesla and, in 2004, Elon Musk joined it becoming the CEO in 2008 [15]. Today, more than a century after the birth of the electric car, we dispose of a wide range of vehicles, each with different characteristics, with which you can travel hundreds of kilometres at full load and without the need to refuel, and it is also possible to charge the batteries at a low cost and in an acceptable time.

3.1.1. Existing technologies

It has been said that EV are not a new concept, instead, it took hundreds of years to get to these results; all this effort led to the multitude of vehicles available nowadays and all different technology that can be applied on cars. The main difference between EVs is the percentage of how much they rely on the electric motor rather than on a conventional Internal Combustion Engine (ICE) car.

A **hybrid electric vehicle** [19](HEV) integrates both a conventional internal combustion engine (ICE) and an electric motor to reduce fuel consumption. HEVs' electric motor plays a fundamental role when accelerating from stop and in all the situations ICE is especially inefficient; otherwise, HEV rely on the ICE unit, for example when cruising at highway speeds. Refuel an hybrid vehicle is the same as a normal ICE vehicles, as owners can only top them up with traditional fuels (usually petrol); indeed, HEV's battery are recharged through what's known as 'regenerative braking' and the motor is used only when conditions are suitable, meaning drivers do not have to change their behaviour with

respect to a normal ICE car.

A **Plug-in Hybrid Electric Vehicle**[19] (PHEV) still rely on an ICE, similarly to a hybrid, but comes with distinct differences. PHEVs generally have larger battery packs and more powerful electric motors, as the electric system does a lot of the heavy lifting while driving. This introduces the possibility to choose if driving in electric-only mode, switching the ICE off entirely. A PHEV can be refuelled as a normal ICE, and the car will automatically recharge the battery while driving, or with both fuel and electricity; in fact, a PHEV can run on just petrol if all battery charge is used up, and battery charge alone if all fuel is used up.

A **Battery Electric Vehicle** (BEV)[19] is a ‘full-electric’ car, with no form of ICE so powered exclusively by electricity, whose electric motors drawing current from onboard battery packs. The completely dependency on electricity make mandatory to build larger and larger capacity batteries with a greater kilowatt-hour (*kWh*) output than comparable hybrid and plug-in hybrid electric vehicles; this extra battery tech typically results in higher costs with respect to other type of EVs. To be driven BEVs need to be electrical charged through either an EVSE, a home charger or fast charging station, or energy recouped by regenerative braking.

A **Mild-Hybrid Electric Vehicle** (MHEV)[19] is the less interesting because it uses an integrated starter generator (ISG), a 48-volt starter motor, integrated with the ICE. This electric motor only helps the ICE on a starting phase when the first one is more efficient. Contention remains about whether MHEV can be considered a ‘true EV’ due to the fact that the electric motor actually cannot accelerate the vehicle by itself.

Fuel Cell Electric Vehicles (FCEV) differ from BEVs only in the way they store energy. Unlike BEVs, FCEVs create their own electrical charge through a chemical reaction generally involving hydrogen. This means FCEVs can be filled with hydrogen, do not require charging from the grid and the only waste product is water. In 2013, three of the most known cars producers, Daimler, Renault-Nissan and Ford, stated an alliance to accelerate commercialization of these kind of vehicles. The aim of this cooperation was to create a common standard for the usage of a system based on fuel cells which will help the spread of zero emissions vehicles. The road map provided that in 2017 a FCEV would have been ready to be placed on a mass market and the project includes company from Europe, Asia and America. Today this technology is available and mounted on some cars, but also on some vans: between 2013 and 2019, cars such as the Hyundai ix35 Hydrogen and Nexo SUVs with 670 km mileage and a total power of 163 hp were placed on the market. We could cite other models, other manufacturers, and it would also be interesting to deepen the operation of this technology with added advantages and disadvantages, but this would go beyond the scope of the thesis.

3.1.2. Battery

It should be clear for the reader that the battery is one of the most fundamental parts on an EV; it is, together with the electrical motor, the core of this technology. There exist different types of battery based on their chemical composition, various model for representing battery aging and different way to charge them.

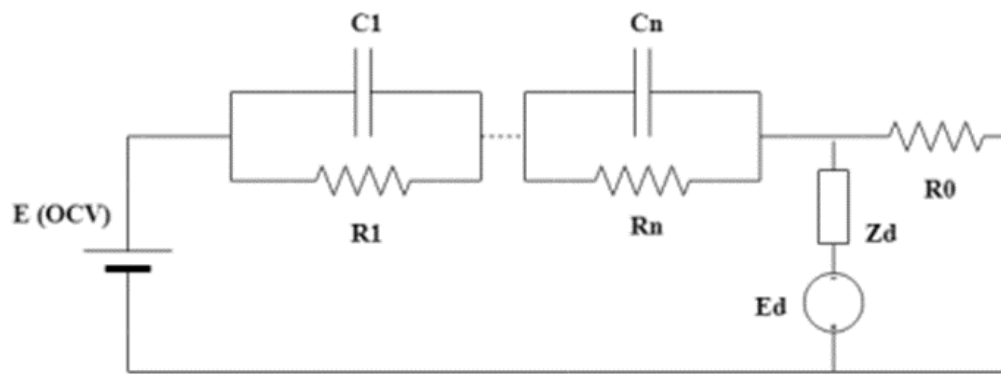


Figure 3.3: Battery electrical schema: each parallel RC circuit represents a single cell.

Battery typologies A battery [22] is an electrochemical power source made connecting rechargeable cells in series and parallel. Each cell has an anodic and a cathodic surface and between them are placed the electrolyte and the separator. When a single cell is being discharged, a redox reaction occurs at the two surfaces, which is the sum of two separate reactions: the anode oxidises releasing electrons while the cathode reduces accepting electrons. By exploiting the difference between the standard reduction potentials of the chemical species at the two surfaces (when charged, the cathode always has a higher potential compared to the anode), an external circuit can be electrified forcing electrons moving in it, therefore transforming the energy of chemical bonds into electric energy.

In a rechargeable cell, what happens during charging is the opposite process: by applying an external voltage source, it is possible to ideally restore the initial conditions. Ni-MH, Lead-acid and Li-ion battery are the most frequently used technologies in the transportation sector together with a new type called ZEBRA batteries which is based on Na-NiCl₂ (sodium-nickel chloride) and still under development. In a nutshell, the Li-ion technology can store more energy and power than the others considering the same weight/volume; furthermore, Li-ion batteries have also high charge and discharge efficiencies (95%-98%). The advantages of Li-ion batteries continue considering that their cycle lifetime is around 5000 cycles compared to ZEBRA (2000-3000), NiMH and Lead-acid (200-300). Although Lead-acid batteries are the cheapest with a cost of 60\$/kWh and due to this still widely

used for very low HD vehicles, in the construction of an EV it is preferable to use Li-ion batteries; for what concerns ZEBRA batteries, they have niche applications due to heavy duty vehicles. At last, although Ni-MH batteries contributed to the rise of hybrid and electric vehicles, their potential for further market penetration has been reduced in favour of Li-ion technology that are expected to reduce their cost and improve their energy density, power density and battery lifetime ???. Generally, there are at least 13 parameters to describe one battery, and to better understand what comes next, it can be useful to present some of them. The main specifications are:

- **Energy [kWh]** It is the amount of Watts that can be provided in 1h; for example, if an EV has a capacity of 42kWh this means that using it at 100% it will be totally discharged in 1h, but using it at 50% the discharging time will be 2h. Same reasoning can be applied to charging: if the same EV is charging connected to a 7kW EVSE, this means that it will be charged with an energy of 7kWh and to fulfil a capacity of 42kW it will take 6h;
- **Capacity [Ah]** Similarly to energy, it is the number of Amps can be delivered by the battery in 1h. The capacity is directly proportional to the amount of electrode material in the battery, so to its geometry;
- **SOC [%]** the State Of Charge is the percentage value that indicates how much of the maximum capacity is actually stored in the battery. It can be computed by the following formula:

$$SOC = 1 - \frac{Q_c}{C_q}$$

where Q_c is the charge extracted from the cell and C_q is the cell capacity, and they are computed as follow:

$$Q_c(t) = \int_{t_0}^t I(\tau)d(\tau) + Q_c(t_0)$$

$$C_q = I_n t^k$$

- **Terminal Voltage [V]** The potential difference between the battery terminals. It depends on the voltage of each cell, which is related to its chemistry, the temperature and SOC.
- **Cycle lifetime** It is the number of times that a battery can be discharged and recharged after which the battery capacity drops below 80% of the nominal value. This parameter is specified by the battery manufacturer as an estimate, because it can vary depending on environment conditions: charging and discharging rates,

temperature and depth of discharge.

Notice that a battery is a simple piece of hardware that do not know how to regulate itself, it only reacts to voltage supply. To regulate this value or the amount of current flowing in it there must be implemented some regulator. To satisfy this need, each vehicle is equipped with a unit called BMS (Battery Management System) mandatory to achieve the cell balancing at module level. The purpose of a BMS is to reveal the state of operation in the form of state of charge and state of health, indicate the end of the life cycle and it must provide to automotive battery management systems critical features such as voltage, temperature and current monitoring, battery SOC and the balancing of battery cells. All the information extracted by the BMS are then transmitted to and elaborated by the on board computer and, in particular, from the EVCC module.

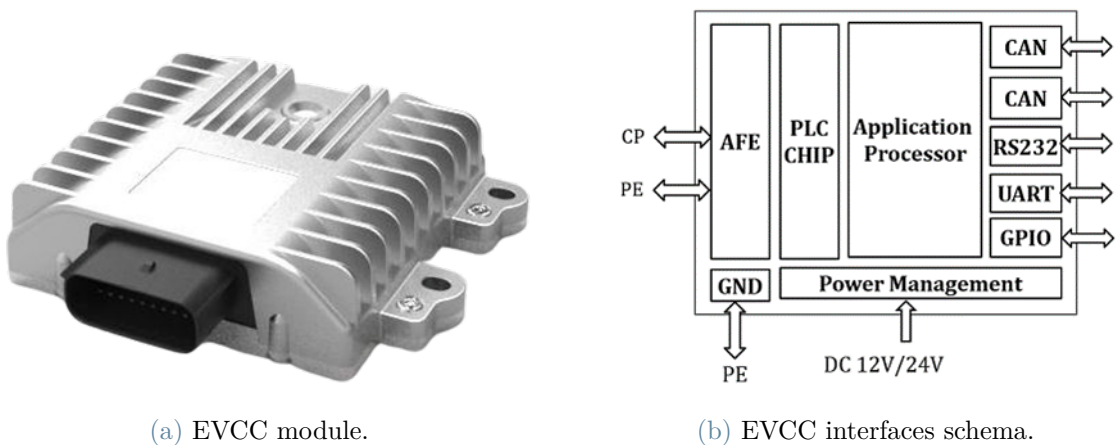


Figure 3.4: The EVCC communicate on one side with the BMS to exchange information about battery status; on the other side it communicate with the SECC, its counterpart mounted on EVSEs [10].

The EVCC (EV Charge Controller) acts as a gateway to mutually exchange information between an external charger (that will be equipped with its own controller, SECC) and the vehicle using different protocols; the one in Fig.(3.4) for example dispose of four serial interfaces to communicate with the BMS and the rest of hardware. It is inside the EVCC that resides the software to communicate with its counterpart within the EVSEs to establish a digital communication: it opens a TCP/IP communication with the SECC to exchange messages, that will be shown later in this document, complying with the standard that is the core of this thesis, ISO15118.

Battery aging

Battery aging is a normal behaviour each battery undergoes causing performances deterioration over time, due to a reduction of its useful capacity and an increase of internal series resistance. The interface between electrolyte and electrodes is the critical area where the side reactions take place; the anode carbon electrode-electrolyte interface is the part mostly subjected to the aging process, which brings to the formation of a Solid Electrolyte Interphase (SEI). It happens during the first cycles of a new battery as a reaction between intercalated Li^+ and the electrolyte solution and, as a result, it is created a protective layer that prevents further degradation of the graphite anode and allows a stable operation of the cell. Unfortunately, this protective layer deteriorates during time, and this cause the irreversible loss of lithium and electrolyte, decreasing battery capacity. Still on the anode surface another aging mechanism occurs: lithium plating. The lithium, indeed, can be accumulated onto the anodic graphite instead of intercalation, so to generate dendrites and lead to hazardous short-circuits in the worst scenarios. Not only anode is subjected to aging; on cathode side, in fact, there are other aging phenomena with a lower impact with respect to the ones seen before: surface film formation and active material degradation.

These phenomena occur whether the Li-ion cell is used or not, but it is common to consider them as a sum of two different reasons: calendar aging and cycle aging.

Calendar aging refers to the warehousing conditions, so when the cell doesn't operate. In this condition the capacity fade can be divided into a Reversible Capacity Loss (RCL), also called self-discharge, that can be recovered recharging the cell after the storage, and Irreversible Capacity Loss (ICL). The main parameter affects calendar aging is the storage temperature: high storage temperature makes easier the side reactions that produce cell degradation. Another important factor is the SOC: cells stored with an high SOC show an increased capacity fade due to the higher internal disequilibrium.

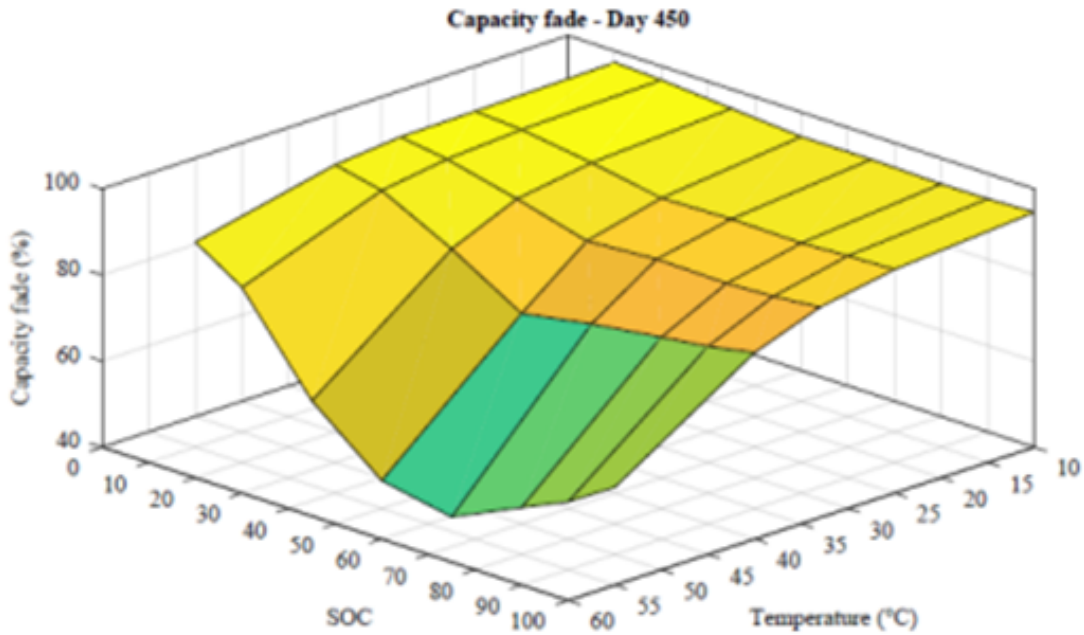


Figure 3.5: Calendar aging simulation over a period of 450 days using as parameters the SOC, temperature and capacity.

Cycle aging, in addition to temperature and SOC, is related to cell charging and discharging cycles and cut-off voltages; in particular, high charging cut-off voltages accelerated the aging phenomena, specifically the capacity fade, while low discharging cut-off voltages affected the aging, particularly the power fade. This aspect of the whole V2G infrastructure is very important because in this new concept of energy sharing the batteries undergo to many incomplete charging/discharging cycles and it is worth wonder if batteries would be subjected to degradation in a sensitive way. Thanks to a study carried out by the Department of Mechanics of the Politecnico di Milano it has been estimated that the performance of a used battery in an infrastructure without controls compared to those of a battery applied to a V2G infrastructure are not so far from each other.

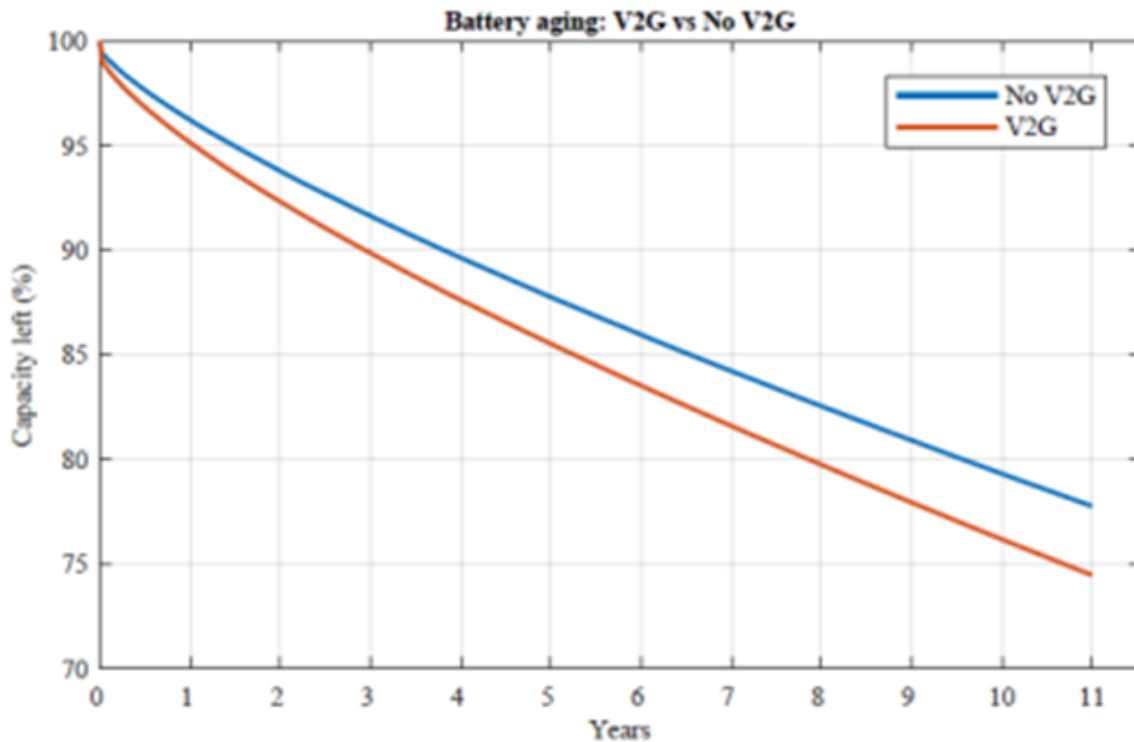


Figure 3.6: Comparison between global aging model whether the battery works its all life in a V2G environment or not.

What is immediate from the plot above is that after a long period of time the offset between a battery used in a conventional way and the one used in V2G is less than 4%; this means that, as could have been thought in a first analysis, V2G technology degrades battery capacity over time, but the economic and technical advantages have more impact than the increasing deterioration. Each battery will last less but with great improving on grid stability and standalone supply technologies.

3.2. EVSE

An EVSE (Electric Vehicle Supply Equipment) is a piece of equipment that supplies electrical power for charging plug-in electric vehicles. Although batteries can only be charged with DC power, most electric vehicles have an onboard AC-to-DC converter that allows them to be plugged into a standard household AC electrical receptacle. Charging stations provide connectors that conform to a variety of standards. DC charging stations are commonly equipped with multiple connectors to be able to supply a wide variety of vehicles. Nowadays, there exists three technologies to recharge EVs, some already in use for years other developing: conductive charging, inductive charging and battery swap. Conductive charging is the most conventional way of charging, it has been used for years and it is the easiest to implement because it is based on everyday charging rules: to exchange energy between an EV and an EVSE it must be created a cabled connection. For what concerns the other two methods, they already exist but they are under developing, and, so far, there is no standard who rules them; actually, the protocol which will handle this new energy transfer mode is what this thesis try to anticipate: in that document, indeed, will be presented and explained the architecture for inductive (wireless) charging and for BPT (Bidirectional Power Transfer) that is the goal of this thesis. Despite the simplicity of the schema in Fig.(3.1), there are different possible configuration depending on how both EV and EVSE are designed: it is possible that the EV would mount the inverters on board so the EVSE will provide just AC current making it easy to build a charging station; on the other hand, it is possible that both inverters are inside the EVSE and the EV accept to be charged only by EVSE which supply DC current. Each of these configurations and their limits in term of time and current are reported in the following sections. [22].

3.2.1. Charging modes and time

Looking carefully at Fig.(3.1), it is reported a label, inside the “Grid” block, that says “Mode 1,2,3,4”. These modes are four different ways of getting power from the grid and depend on how much of the scheme in Fig.(3.1) is part of the vehicle and how much is, instead, inside the EVSE; this, in fact, is strictly connected to what has been said at the end of previous section: there is a specific Mode for each kind of vehicle and, vice versa, each EV can be recharged only by the right EVSE. Before analysing them, it can be noticed from Fig.(3.7), that these modes only refer to conducting charging and, in fact, the standard that defines them, the IEC61851 itself, only regulates the exchange of wired energy.

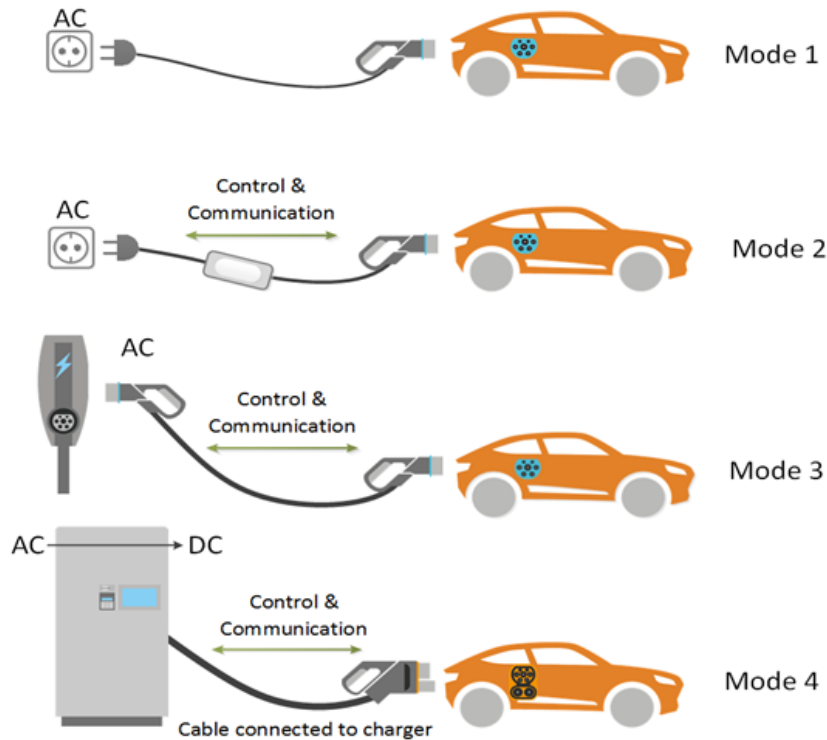


Figure 3.7: Different charging modes depending on what are the functions of each entity.

Let us see more in detail the differences between them [6, 13, 22]:

- **Mode 1** Among the four, this is the more practical way, because the only things needed are an EV with on board inverter, a simple cable (of course complying with IEC61851) and a common AC inlet. This way there is no need of control by the side of the EVSE and the only controller involved is the BMS. The advantage of this mode is that allows the user to be independent from the environment because of simplicity on finding a 220V socket both in private or public places; on the other hand, this is a very slow way of charging due to low value of current and voltage respectively limited to 16A and 250V single-phase or 480V three-phase. This charging mode is prohibited or restricted in many countries;
- **Mode 2** This mode has a lot in common with the previous one, although it introduces a different type of cable equipped with an in-cable control and protection device (IC-CPD); this integration is due to the fact that household socket-outlets do not always provide electric power according to the actual standards or they may not be designed to sustain a continuous current draw at its maximum value for long period, as it would be an EV charging session. The IC-CPD has been introduced with this purpose, to perform the required control and safety functions

which means to detect and monitor the protective earth connection, over-current and over-temperature. The advantage of this mode is the increased current limit, now set up to $32A$, that allow to plug in both household and industrial socket. The disadvantage is that this connection still remains a slow way of charging;

- **Mode 3** In this mode there is a return to a cable without any controller, because it has been substituted and integrated by the EVSE. In this case, as in the previous two, the EV must be equipped with the on-board inverter because the output supplied is in AC current and it will be its job to convert from AC to DC. The safety of these Mode 3 EVSEs is entrusted to certain circuits that are subject to safety standards that will be presented in Chapter 3.2.4. With this system, it is possible to adjust the charging current to the maximum current capability of the cable assembly because it can supply up to $32A$ with either a $250V$ single-phase or $480V$ three-phase network. It is also possible to choose to limit current to $32A$ making the architecture compatible with Mode 2 (both single-phase and three-phase). Slow or fast charging using a EV multi-pin socket with both control and protective functions, with current and voltage respectively limited to $63A$ and $400V$ three-phase. This time the vehicle is connected to the grid through a complete electric vehicle supply equipment (EVSE) that can be represented by a public charging station;



(a) Wallbox by Daze.



(b) Fast charger by Tesla.

Figure 3.8: Comparison between Mode 3 and Mode 4 EVSE.

- **Mode 4** This is like an update of Mode 3 because it adds the ability to charge an EV directly into DC, which means that cars are no longer obligated to have the inverter on board. It is an important concept because it has a great impact on this market because electric vehicle factories could save space in favour of a larger

battery that translates into a longer range. The most important parameters here are the DC supply limits that vary for each different standard (Japanese or European, respectively CHAdeMO and CCS Combo): with this charging mode it is possible to charge up to 400A and 1000VDC, depending on the connector technology. Also in this case, the vehicle is connected to the grid through an EVSE which, thanks to its high output values, allows to load the vehicle in a very short time, deserving the name of Fast Charger.

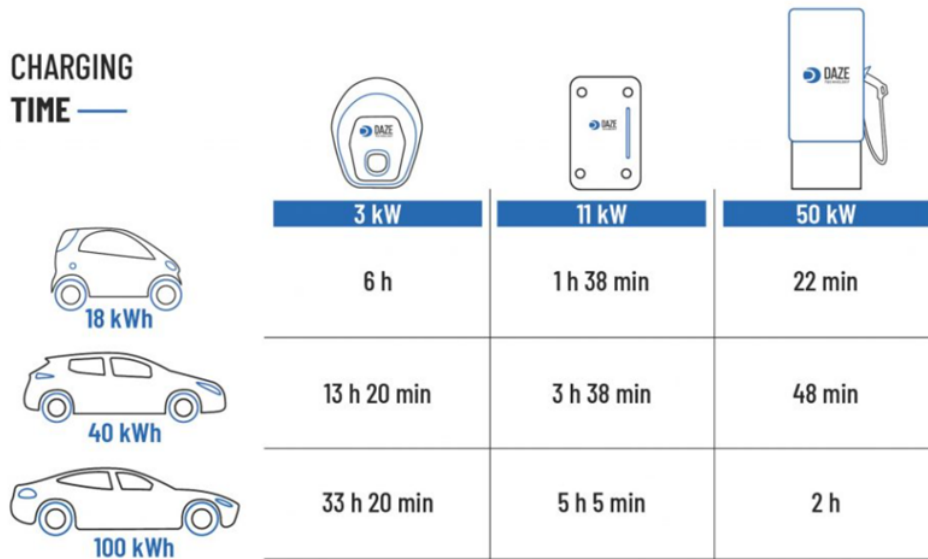


Figure 3.9: Time computation to fully recharge EVs depending on battery capacity and EVSE power [14] .

From Fig.(3.9) it is possible to get an idea of how long it takes to fully charge a battery pack depending on its capacity and type of EVSE. The company who made this computation, Daze, which is also a client of the one where I took my internship, S&h, provides on their site the formula to calculate this time [14]:

$$\text{Charge Time} = \frac{\text{Battery Capacity (kWh)}}{\text{Charging Power (kW)}}$$

Thanks to that, it is possible to compute how long a Mode 4 EVSE takes to charge every vehicle; here, to give an idea about time recharge for affordable EV, it is been going to analyse one of the cheapest electric vehicles available on the market, a FIAT 500e with a capacity of 42kWh. Let us see the best-case scenario for this vehicle, where it is possible

to provide the greatest current available to whatever EV:

$$Charge\ Time = \frac{42kWh}{400kW} = 6,3mins$$

Unfortunately, as it has been said, this is not exactly how it works due to design limitations: the car just mentioned, indeed, is able to accept up to 11kW, in Mode 2 and 3, or up to 85kW, in Mode 4, that lead to a minimum time, in DC, of 29,6min. This time may increase if several vehicles are connected to the same column and the controller inside the first chooses not to deliver the maximum power, in order to safeguard the stability of the grid.

3.2.2. Type of connector

[22] In the previous chapter we have analysed the general configuration of the recharge distinguishing the actors of the system and their roles, now we will see in what the various types of connectors differ and what are the motivations of their designs. During these years, many standards have been developed both for AC and DC charging due to differences between American, European and Asian car manufactures; this distinction leads to mainly four different type [11], as shown in Fig.(3.10).
















			
	Type 1	Type 2	GB/T Standard
AC charging			
Mode 2			
Mode 3 case b			
Mode 3 case c			
DC charging			
Mode 4			

Figure 3.10: Caption of the Figure to appear in the List of Figures.

- **Type 1** This American and Japanese standard has a single-phase plug which allows

to supply up to $7.4kW$ ($230V$, $32A$) with neutral and phase conductors in the upper part and ground, control and protection conductors at the bottom. Power pins can be distinguished by signal ones by their dimensions: the bigger pins are dedicated to power exchange while the others deliver signal from the EVSE to the EV and vice versa;

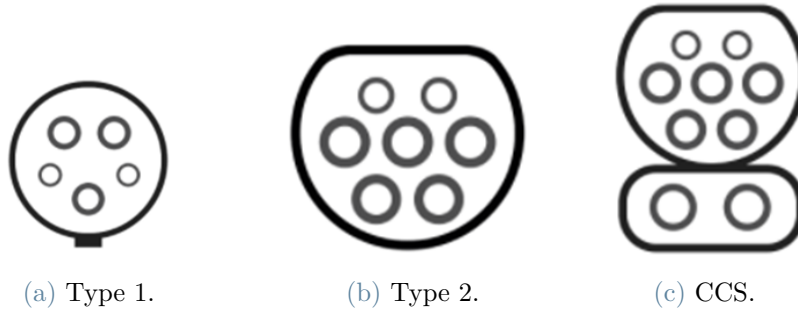


Figure 3.11

- **Type 2** It represents the European standards that allows the three-phase recharge; it is easy to notice that it has 2 more connectors with respect to type 1 and they are dedicated to carry the 2nd and the 3rd phase. Charging power level is limited to $22kW$ in private spaces and to $43kW$ ($400V, 63A$ in AC) in public charging stations. Tesla itself mounts a modified version of this socket on its superchargers and, interesting fact, offers charging to its customers for free, recharging a Model S to 80% within $30mins$;
- **CCS** The Combined Charging System is a hybrid standard because integrates type 2 to another new socket. Looking at the picture, it is not difficult to notice a strong similarity between type 2 and CCS; the upper part of the socket, indeed, has the same function for both type and it is used for AC current. The two additional connectors, instead, is used to carry DC current making it the perfect socket for Mode 4 charging. Its physical limit in power is set to $170kW$, but usually it works around $50kW$;

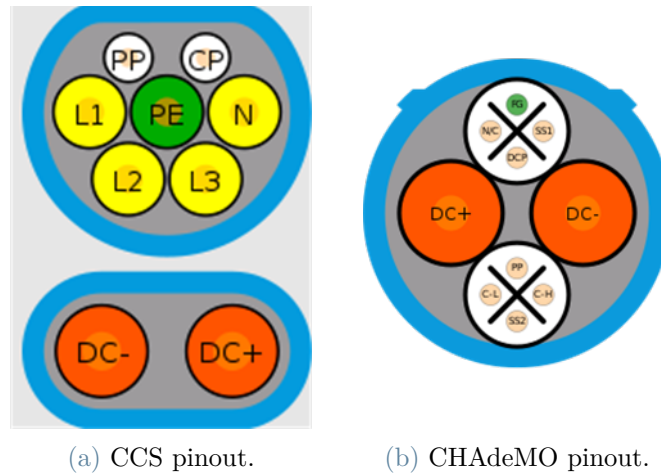


Figure 3.12

- **CHAdeMO** This socket, whose acronym stands for “CHArge de MOve”, equivalent to “move using charge”, is completely different from the others, and it is clear looking at the schemas above. As CCS, it allows fast charging sessions but unlike it, it is a Japanese and not European standard. This original connector can provide [25] up to $62.5kW$ (500V, 125A, in DC) while the newest one, CHAdeMO 2.0, can supply up to $400kW$ (1000V, 400A, in DC), but aims to a limit of $900kW$ in the next future.

These are the most standardized kinds of connectors, but there are more, and it would be desirable to opt for only one standard for the deployment of the charging infrastructure. Nevertheless, it is still not clear which one will be the DC unique standard, as other connectors like the Tesla supercharger, the Chinese GB and the Japanese CHAdeMO are all equally adopted by car manufacturers and charging station producers[22].

3.2.3. Protocols involved

There is a large number of communication protocols, involved in this structure, that are implemented in each level of the ISO/OSI stack to allow communication between SECC and EVCC. At the lower layer, connection are handled by the standard **IEC61851** that will be explained a little bit more in detail in Chapter 4.1: basically, a PWM signal is used to determine the connection state between EV and EVSE and, based on that, it modifies its duty cycle with the aim to require less or more current to the EVSE. When PWM is around 5%, comes into play another protocol, the one analysed in this thesis: **ISO15118**. This standard will be treated deeply in Chapter 4.2, so by now let us just say that this protocol explains how to build a digital communication between EV and EVSE

and how this architecture establishes relationships with all the actors of the system. These standards, in turn, use a huge number of other protocols such as, for example, the well-knowns **TCP** and **UDP**; these protocols are used in the transport layer to tell a Pc or a device how packets should be sent over internet. TCP is a more reliable protocol and ensures each packet to be delivered, but it takes more time to terminate the transmission; on the other hand, UDP is used when a faster communication is required and there is little interest in packet-loss. At the same level of these protocols, but with a safety purpose, it is used **TLS** to encrypt sensitive data exchanged during charging session; the usage of this protocol works a bit differently depending on the environment (public or private). One level above, in the network layer, **IP** protocol is used in communication between EV and EVSE; it may sound strange in a first moment, but IP used in ISO15118 is an IPv6 instead of the usual IPv4:

- **IPv4** An example of IPv4 address is 192.168.1.190. It is a 32 bits address usually in decimal base, it defines a Class A, Class B or Class C network depending on how many hosts there can be connected and is commonly used all over the world to universally identify a device inside a network;
- **IPv6** An example of IPv6 address is 2002:c0a8:01be::c0a8:01be. It has quite the same purpose of its older version except it is a 128bits address usually expressed in Hex format, so it allows to define a greater number of hosts.

Ethernet protocol is the last network standard which is worth to mention because it allows EVCC and SECC to communicate each other at the lowest level. All these are network protocols but there are other protocols, such as for serial communication, that was implemented here. Taking a look at lower levels of the case study, for example, **CAN BUS** is used to allow communication between the inverter and the SECC, but this is quite outside the area of expertise of this document. What is more pertinent and interesting is the protocol used during the interaction between PLC module and EVCC (or SECC): the **SPI**; this standard is a serial communication 4-wire based, that let a master communicate with one or more slaves, that will be better described in Chapter 5.2. Last mention goes to **HPGP**, HomePlug Green PHY, a protocol based on Power Line Carrier (PLC), chosen for PLC modules to talk to each other.

3.2.4. Safety measures

According to Internal Energy Agency, it has been registered an annual average increase, in EV and PHEV global stock, of 60% over the period 2014-2019; it also has been estimated by Bloomberg New Energy Finance that, in 2040, sales of electric vehicle could represent

up to 50% of global vehicles. This trend will lead to an intense growth of charging infrastructure that, of course, need to be energized; the problem here is that EVSE can be installed in various environment, private or public, maybe reachable by kids, exposed to rain and so on; it is clear that such system must be provided of the right security measures. When an EV is going to be charged, in the moment when the cable is plugged in, information about the vehicle flow into the SECC and values like maximum power, voltage and current are stored. This phase, the handshake, is useful to prevent damage to the battery pack, but is unable to react to damage on the EVSE circuit. To avoid these failures to ruin the vehicle or, in the worst-case scenario, hurt people getting in touch with the EV itself or with EVSE, different kind of electrical protections are adopted, and they can be sum up in three categories:

1. Protection against short-circuits and overloads
2. Protection against electric shocks and risks of electrocution
3. Protection against over-voltage

Protection against short-circuits with circuit-breakers Like for any final circuit supplying a load, IEC 60364 part 7-722 requires over-current protection which means, practically, protection against short-circuits or overload in the final circuit. In this field, these kind of failures are really meaningful and it is easy to understand why if we think that some charger can be rated up to 22 kW (carrying a current of 32A) or 50 kW (carrying a current of 63A). As suggested by the title of this section, to face and avoid this undesired situation, circuit-breaker are used to protect both circuit and car users who get in touch with the EVSE for whatever reason; of course, each of these breaker is complying with their relevant standard, namely IEC 60898-1 or IEC 60947-2, installed in the distribution switchboard. Compliance to these standards provides safe behaviour during the entire life of the installation. This includes the case of high short-circuit (e.g. 6 kA, 10 kA or 20 kA), overload in the circuit, temperature rise behaviour when nominal current is passing, ageing, behaviour of terminals, insulation, electrical or mechanical endurance.

Protection against electric shocks with 30mA RCD When an EV charger is installed, it must be thought about the environment where it will be used: it can be located in public areas, outdoor, with presence of water, presence of children; moreover it must be considered that an electric vehicle has a large conductive area and it is very easy that it could come in contact with human body. Obviously, protection against electric shocks becomes one of the most important feature of this system. Firstly, to cover an insulation

fault, connect all accessible metal parts to the earth (Protective Earth) is always used as common safety measure such as disconnect the supply whether a fault is detected. This function is performed either with a circuit-breaker in TN earthing system or with a medium sensitivity RCD for TT earthing systems; just for note, TN and TT are two of the possible configurations of supply connection: in TN connection the use of neutral is mandatory while in TT are used RCD (Residual Current Devices).

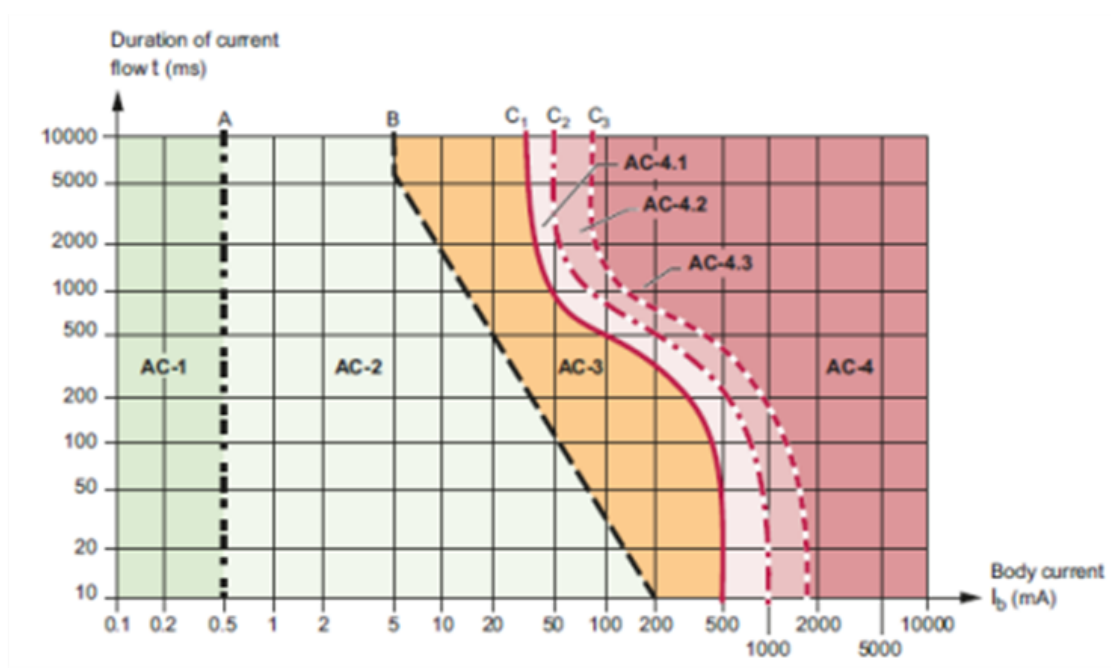


Figure 3.13: Time-current effect of AC current on human body (IEC 60479-1) – curve c1 defines the threshold of ventricular fibrillation.

Secondly, according to what is stated in IEC 60479 series about the effect of currents on human beings, the threshold of ventricular fibrillation is defined by curve c1 in Fig. 3.13; looking at those value it is immediate that the current limit must be set above $30mA$ and, indeed, this is one of the most important requirement of IEC 60364 part 7-722: each connected point must be protected by a $30mA$ RCD. As the Electric Vehicle may reject DC residual current during charging, the selection of type of RCD shall be done carefully. The most advanced solution is to install a $30mA$ type B RCD, complying to IEC 62423, that provide protection against residual AC, pulsating DC and smooth DC residual currents.



Figure 3.14: RCD of type B complying to IEC 62423.

One advantage of this solution is to ensure continuity of service in case of small DC residual currents, provided that it is not dangerous for human beings. On the other hand, it is possible to choose a type A RCD in conjunction with an EVSE equipped with a Residual Direct Current Detecting Device (RDC-DD) that has a sensitivity of $6mA$ and that let the whole system to be more sensitive as well, so no service continuity is ensured [21].

Protection against overvoltage If a lightning strike near an electricity network, the power surge generated would propagate into the network without any significant attenuation. As a result, the acceptable levels to withstand the voltage recommended by IEC 60664-1 and IEC 60364, in a low voltage system, can be overcome by surges. Protections on electric vehicle have already been integrated by law and, if they are designed with an overvoltage category II of IEC 17409, therefore, they should be protected against overvoltage with an amplitude greater or equal to $2,5 kV$. As a consequence, IEC 60364-7-722 requires that EVSE installed in not private environment must be protected against transient overvoltage. For this purpose, there are devices, called SDP (Surge Protective Device), which, complying to IEC 61643-11, could be of a type 1 or type 2 and are installed in the switchboard; their usage need to supply the electric vehicle keeping a protection level $U_p \leq 2,5 kV$.

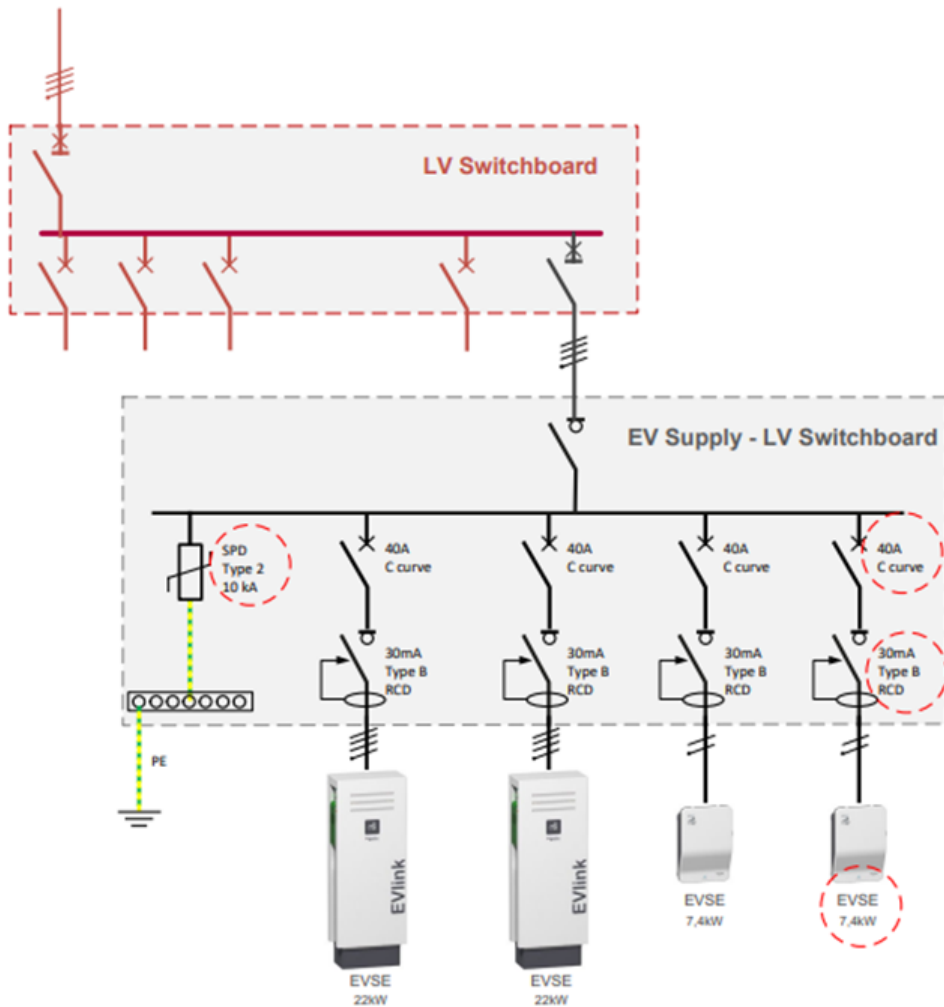


Figure 3.15: Recommended single line diagram for charging electric vehicle.

In this chapter it has been mentioned a large number of standards related to their sub-parts and it has been seen that each of them tends to prevent different kind of damage or failure; all of them are collected by the standard IEC61851 which contains all the safety requirements covering the EVSE, as an equipment. For example, it is stated that each EVSE shall rely on IEC60364-7-722 and thus provide protection against short-circuits, electric shocks, and overvoltages. Moreover, as shown in Fig.(3.15), they should be supplied by:

1. a switchboard which is equipped with a dedicated circuit with circuit-breaker complying to IEC60898-1;
2. a 30mA type B RCD complying to IEC62423;
3. a surge protective device complying to IEC 61643 series, where the connected point is accessible to public.

This was only an example, it has been said that there are different type of protection (have a look at the RCD itself that could be Type A or Type B) so these choice are made in design phase, in the full respect of all existing standards.

4 | Standards overview

The protocol that manages the communication between vehicle and column is called IEC61851. This standard defines an analogical type of communication based on voltage thresholds and PWM modulation, it has been in force for years and is still being revised; among the latest updates, indeed, it has been introduced the dependency to a newer standard: ISO 15118. Introducing this dependency, practically, means that it has been created a new communication standard, the one described by the ISO rules, that is accessible by the IEC one, which allows digital communication between EV and EVSE. To have a complete view of where it has been operated, let us have a look at both standard, with particular interest in the ISO norm.

4.1. IEC61851

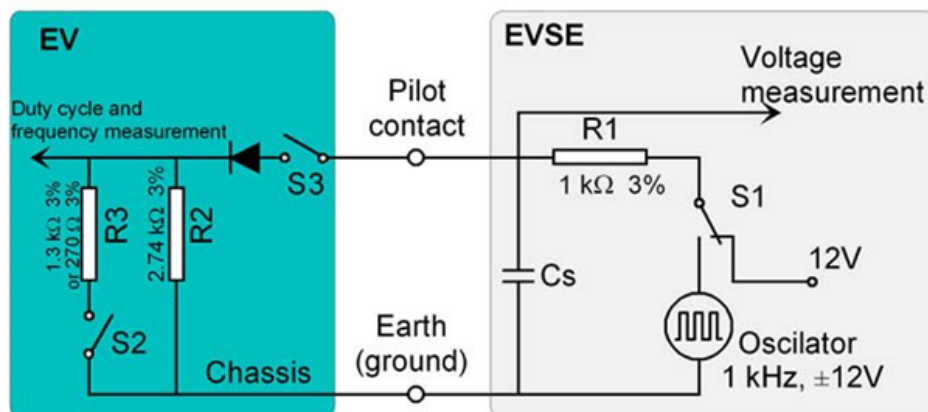


Figure 4.1: Control pilot circuit in Mode 3 and 4.

Depending on the status of the switches [6] S1, S2, and S3, shown in Fig. 4.1, different voltage levels will appear at the “pilot contact”, also known as Control Pilot, CP, and these will represent different charging states basing on which decisions about the charging session will be taken. It is possible to follow a general charging session in Fig.(4.2). Before

plugging in the charging cable, the switches S2 and S3 are off and S1 is connected to the 12VDC supply, so this is the value measured by the EVSE at the pilot contact. This information is interpreted by the EVSE as the EV is not connected yet and the state of the system is “STATE A”. After the physical connection between the EV and the EVSE is established, the controller on the EV side can close S3, increasing resistance on the CP and reducing the measured voltage to about 9V. This drop informs the EVSE that the cable is connected to both the EV and the EVSE, but it tells the EV that the EVSE is not ready yet. When the EVSE is ready to charge the EV, S1 is connected to the oscillator and the PWM signal at the pilot contact tells the EV that the EVSE is ready: the state of the system goes from “STATE A” to “STATE B”. Then, the EV close S2 to make the CP voltage drop to 6V to indicate that it is ready as well and the state of the system goes to “STATE C”. The voltage readable in this stage depends on the value of the R3 resistor whose amount specifies whether ventilation is

Duty cycle	Maximum current
Value < 3%	Not allowed
$3\% \leq \text{value} \leq 7\%$	Digital communication
$7\% < \text{value} < 8\%$	Not Allowed
$8\% \leq \text{value} < 10\%$	6A
$10\% \leq \text{value} \leq 85\%$	Current = value x 0.6A
$85\% < \text{value} \leq 96\%$	Current = (value - 64) x 2.5A
$96\% < \text{value} \leq 97\%$	80A
Value > 97 %	Not Allowed

required in this charging area or not. With $R3=1.3k\Omega$, the CP voltage is about 6V. This corresponds to charging areas where ventilation is not required. For areas that need ventilation, $R3=270\Omega$ is utilized, which gives a CP voltage of about 3V and let the system enter in “STATE D”. When the vehicle is charged or wants to abort the charging cycle for any reason, it can turn S2 off changing the positive voltage level of the PWM to 9V and informs the EVSE that the EV is not ready anymore to get charged. It is necessary to make a clarification for what concerns the states of the system: until now we have distinguished the states in A, B, C and D, but we must add some information. First of all, there is an additional state "STATE F" that identifies a foul situation; in addition, the actual number of states is double compared to those seen so far, because each one, in the state machine, could be both with PWM active or not. This modulation [16] it is fundamental in this protocol because for each value of the duty cycle corresponds a certain action, as shown in the table above. It returns some helpful information about which information does the PWM bring: if the duty cycle is under 3%, for example, no current is allowed to flow and in the range between 8% and 97% current is computed following what could be consider

a “LookUp Table”. What is very interesting for our purpose, is the range between 3% and 7%; between these two values, indeed, IEC61851 knows that digital communication is required to determine the amount of current to provide, so ISO15118 comes into play, a charging session is set up and the amount of current is established based on charging scheduling that the protocol itself is able to design.

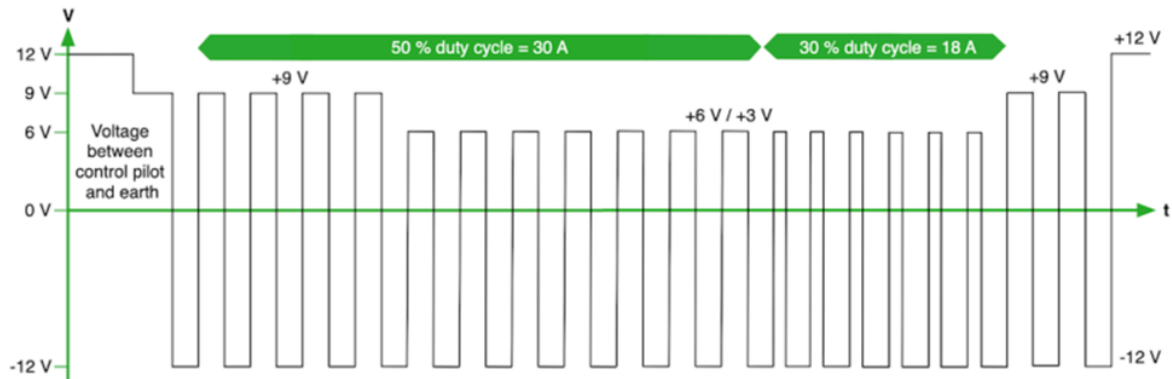


Figure 4.2: PWM signal representation in IEC61851.

4.2. ISO15118

This standard [12] establishes the rules for digital communication between EV and EVSE; to fully understand the updates made to this protocol provided in this project, it could result useful to have an overview on which are the states in ISO15118, how this standard is composed and where is the working area of this thesis. The document “ISO15118 – Road Vehicles – Vehicle to grid communication interface” is made of 9 parts as shown in Fig. (4.3), ISO15118-1, ISO15118-2 and so on, and each of them regulates a certain aspect. The core of this thesis is “Part 2: Network and application protocol requirements” where all messages exchanged between EV and EVSE are defined and the way in which they are sent is explained in details. It is clear, observing this scheme, that there are many aspects touched by the standard, even from part 2, and range from computer security (TLS) to the choice of internet communication protocol (TCP/IPv6, UDP), up to the encoding of messages (V2GMessage, V2GTP, EXI). Analyse every single section would be a useless job since the thesis is not about internet protocols, but more focused on the messages exchanged; for this reason the analysis and contribution of this work will be limited to the 7th layer of the stack, the Application Layer.

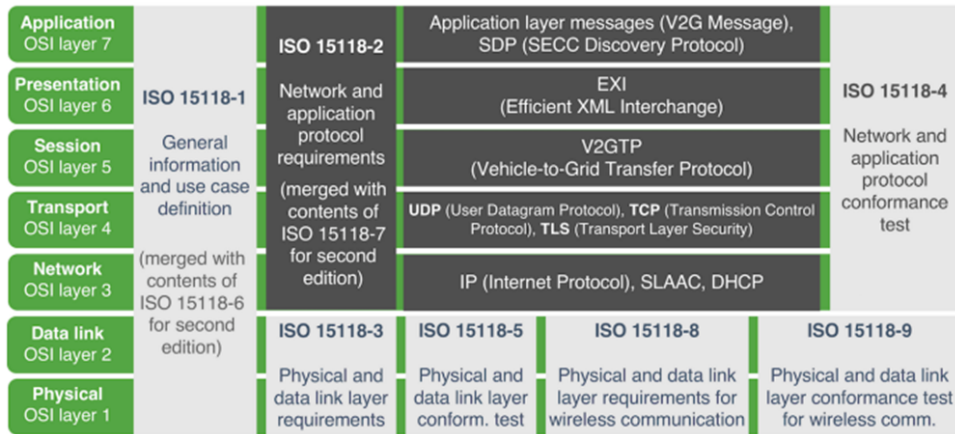


Figure 4.3: Network layers in ISO15118-2.

It is in this level, in fact, that all the messages of the standard are defined, read and interpreted; thanks to the state machine also defined by the standard (a version cleaned by the error signals is reported in Fig.(4.4)), it is possible to follow the sequence of messages and, consequently, to understand where and how to intervene. There are two versions of this graph, one for DC charging and one for AC charging, which, in turn, are distinguishable in two other versions, one describes the behavior of the SECC and one of the EVCC. Due to the architecture of the two different systems, making changes in DC is easier, considering that the company in which this thesis was developed has access to EVSE and not to EV. If we think of an AC recharge, in fact, the currents and voltages taken from the network pass through the column and come out not yet straightened; this means that the conversion from AC to DC takes place inside the car and, consequently, that the bidirectional inverter is located on it. In this case, it will be the manufacturer of the vehicle to write the software for the BPT management of the inverter; working in DC, instead, the inverter will be located on the column itself and it will be easier to implement this new feature being this basically an update of the software. The first approach to this project, therefore, was to modify the state machine for a DC charging, leaving the AC one for future developments. The diagram in Fig.(4.4) shows only the operations side vehicle, the operations side column, instead, are visible in the Appendix, together with the same cases in AC and the diagrams complete of the error signals.

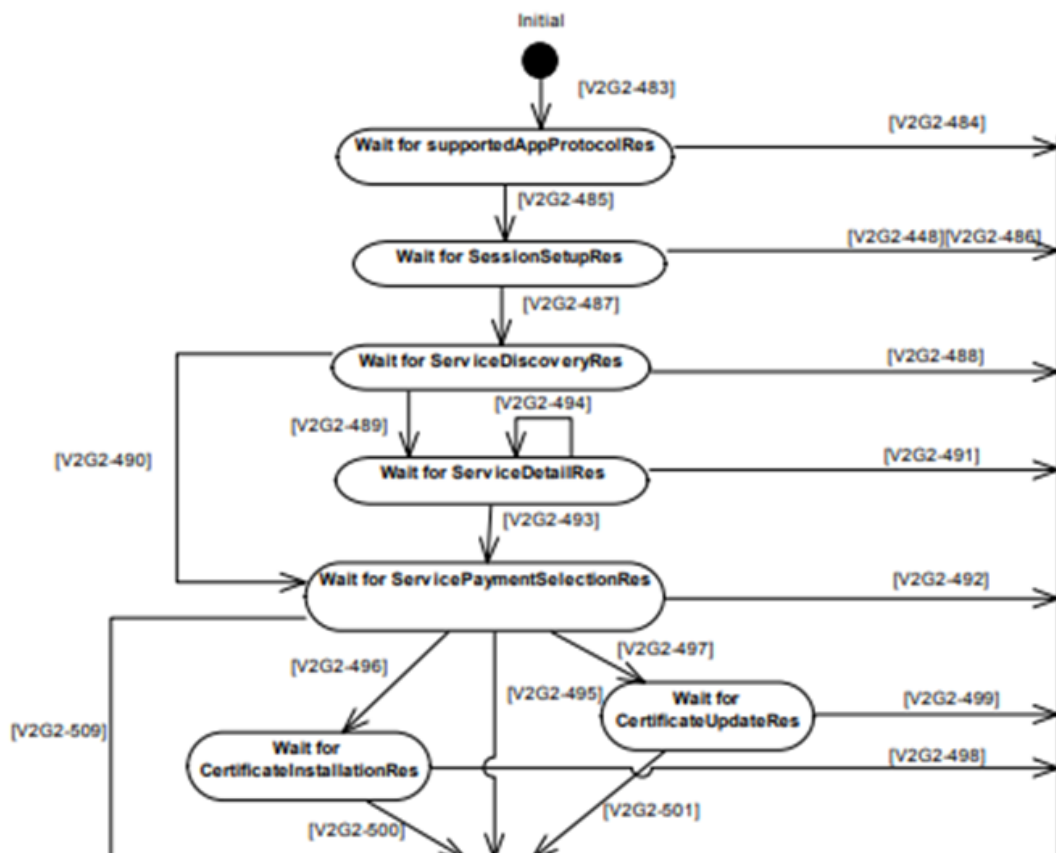


Figure 4.4: EVCC states machine for DC charging.

Messages in details [12] It has been said that in the ISO standard [20] are described all the messages, their operations, the data they carry and the structure they must have at the code level. Here is an example of the analysis of the first message exchanged, the "supportedAppProtocolReq", but what is going to be aid is easily applicable to others; obviously the schemes will be designed differently and they may carry different parameters and information, but in this section the aim is to show the documentation that can be consulted when working with this architecture. Each message type has a graphic structure that vaguely reminds of the UML structure of a Class Diagram, as shown in Fig.(4.5). The software designer who will program the logic for the implementation of this standard will have to respect this structure and report it faithfully, as Marc Mültin and his staff did in the RISEV2G software. As previously mentioned, the document also shows the XML structure to which the message must be subjected, as shown in Fig.(4.6); the definition of the BPT message is visible in Fig.(5.22) and, together with its implementation, in the Appendix A.

8.2.2 Message definition supportedAppProtocolReq and supportedAppProtocolRes

[V2G20-175] The SECC and EVCC shall implement the message elements as defined in Figure 19.

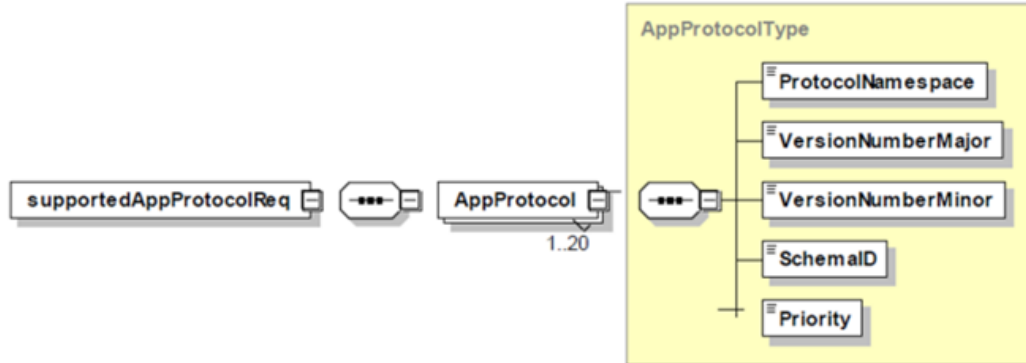


Figure 19 — Schema diagram – supportedAppProtocolReq

Figure 4.5: Message schema definition: AppProtocolType for supportedAppProtocol.

In the present case, Fig.(4.6), it can be seen how the 3 protocols that can be used during a possible charging session have been defined; each of these respects the structure shown before being equipped with a name, a version, an ID and priority number.

```
<?xml version="1.0" encoding="UTF-8"?>
<n1:supportedAppProtocolReq xsi:schemaLocation=" um.iso:std.iso:15118:-20:AppProtocol ../V2G_CI_AppProtocol.xsd"
xmlns:n1="um.iso:15118:2:2015:AppProtocol"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <AppProtocol>
    <ProtocolNamespace>um.iso:15118:2:2015:MsgDef</ProtocolNamespace>
    <VersionNumberMajor>3</VersionNumberMajor>
    <VersionNumberMinor>0</VersionNumberMinor>
    <SchemalD>10</SchemalD>
    <Priority>1</Priority>
  </AppProtocol>
  <AppProtocol>
    <ProtocolNamespace>um.iso:15118:2:2013:MsgDef</ProtocolNamespace>
    <VersionNumberMajor>2</VersionNumberMajor>
    <VersionNumberMinor>0</VersionNumberMinor>
    <SchemalD>20</SchemalD>
    <Priority>2</Priority>
  </AppProtocol>
  <AppProtocol>
    <ProtocolNamespace>um.iso:15118:2:2010:MsgDef</ProtocolNamespace>
    <VersionNumberMajor>1</VersionNumberMajor>
    <VersionNumberMinor>0</VersionNumberMinor>
    <SchemalD>30</SchemalD>
    <Priority>3</Priority>
  </AppProtocol>
</n1:supportedAppProtocolReq>
```

Figure 4.6: Example of information carried by messages: in this case the AppProtocol informs about supported protocols.

All these data are essential during the handshake to determine which protocol is implemented by both the charging column and the car and, consequently, which will be used

during the charging session to dialogue, but each message will carry different information depending on its purpose. This was just an example, but it is essential to understand how the BPT message we are going to create will be defined.

Charge process [17, 20] The message just presented is the first that is exchanged during the session, after the transition from "STATE A" to "STATE B", as can be clearly seen from the state machine in Fig.(4.4). Then, the EV sends another message, the "SessionSetup", with which it is assigned an ID that is unique for that communication session; thanks to this value, session can be paused and resumed at a later time. In this case, the previously agreed upon charging parameters will remain unchanged to ensure charging continues as originally intended by the driver. Next message send by the EV is the "ServiceDiscovery" to ask EVSE about its offerings, so if AC or DC are available and which variant could be used (e.g. single or three-phase for AC), it is also used to know the available identification mechanisms, namely External Identification Means (EIM) or Plug&Charge, and some more services; the EV can also request, for each service, some more information by using the optional "ServiceDetail" message pair. Going on, once this first part of the handshake is over and the identification mechanism, the charging mode, and optional value-added services to use are clear, the EV then will inform the charging station with the "PaymentServiceSelectionRequest". It is important here to state that with term "Payment" has been referred to the authentication and authorization method that has nothing to do with actual payment. For this purpose, ISO 15118 provides two ways of authenticating and authorizing a user for a charging session and they have been presented few rows before: EIM and Plug & Charge (PnC). The first one represents any user authentication method that requires additional action by the driver such as presenting an RFID card to the reader or scanning a QR code on a label at the charging station. Plug & Charge, instead, is more convenient for user because he only needs to plug the vehicle and all information about his contract are automatically exchanged between the EV and the backend of the EVSE; this method also ensures security, data integrity and authenticity through digital certificates and digital signatures. In case Plug & Charge has been chosen as an identification method, a valid digital contract certificate must be installed on the EV in order to be automatically authenticated and authorized by EVSE. If this certificate is not installed yet or if its existing contract certificate has expired, the "CertificateInstallation" message pair can be invoked to install a new one from the charging station. In cases where an EV has a soon-to-be expiring contract certificate installed, it can be sent a "CertificateUpdate" message pair to receive a new contract certificate. Using the "PaymentDetails" messages, if the EV is programmed to select Plug & Charge as its identification method, its contract certificate must be

presented to the charging station so that the driver could be authenticated and authorized for charging. Here comes one of the most important messages for the purpose of this document: the “ChargeParameterDiscovery”. It is very important because with these messages, the EV and EVSE mutually exchange their respective technical charging limits, in particular: maximum and minimal allowed voltage levels and amperage,, the amount of energy needed and the desired departure time provided by the driver. The SECC will then propose a charge schedule to the EV and the EVCC will compare it with a charging schedule calculated by its own. If charging has been set to DC, two more states must be considered: “CableCheck” and “PreCharge”. The first one has the aim to ensure a safer DC charging by checking the cable, whether it is locked into the EV and ready to charge. The second message pair, instead, is another pivotal point of the standard because it is used for adjusting the EVSE output voltage to the EV battery voltage. For DC charging control during an active charging session, cyclic exchange of the requested current from EV side is necessary and this happen through one of the most important message from our point of view: “CurrentDemand”. This message is so important because, by sending it, the EV requests from the EVSE a certain value of current and this will be the point where to ask for negative current during BPT session; in case of AC charging process, this message would be substituted by the “ChargingStatus”. At this point of the standard, the EVCC sends its schedule to the SECC via the “PowerDelivery” message and, if accepted, it can decide to wait a certain amount of time or instantly start charging; to do that, when the PowerDelivery message is sent, its “ChargeProgress” parameter will be set to either “START” or “STOP”. In the first case, the EV will trigger a switch to cause a voltage drop from 9V to 6V and, according to IEC 61851, this will cause a state transition from state B to state C: “EV detected and ready for charging”; consequently, the SECC will communicate to its inverter to close its contactors to initiate the flow of energy. Around the messages exchange that has just been described, it is focused all the work of this document and what is going to be presented is almost the key to the whole work. Depending on the individual circumstance, EV could be asked to renegotiate its requested charging schedule or even instantly stop charging to prevent the grid from a potentially dangerous overload; again, this concept is fundamental because bidirectional power transfer has been intended, during design phase, as result of a renegotiation. It will be at this point that the system will have to take decision on whether activate BPT or not. The charging station can also request to EV a digital signature for the current meter values by using the “MeteringReceipt” message; this triggers the EV to send a signature declaring that it has seen the meter values. When it is time for the EV to pause or end the charging session based on its calculated charging schedule or some client choice, another PowerDelivery message will be sent by EV, where the ChargeProgress parameter is set

to "STOP". Before the end of the process, it can be sent a "WeldingDetection" message to trigger the so called welding detection, according to IEC CDV 61851-23, inspection to avoid contactors to break. A PowerDeliveryReq(ChargeProgress="STOP/PAUSE") causes the communication close sending the "SessionStop" message pair. This message is characterized by the ChargingSession parameter which can be set to either "PAUSE" or "TERMINATE": in the first case, certain parameters, like the agreed-upon charging schedule, are temporarily stored by the charging station so it can apply these values when the charging resumes otherwise they are dropped and communication and session are closed.

5 | Practical work

Up to now it has been presented the system of production and transport of energy, analysed the main actors of these architectures and taken notes of the functioning of the charging processes analysing in detail the protocols. What would be interesting to get, at this point, is an updated charging column (**RPiEVSE**) with a new protocol that supports the BPT; unfortunately, to do this, an electric vehicle that allows this new exchange of energy is required, to test its operation. It is obvious that buying a car with the only purpose of developing a thesis is not the best possible route, for this reason the first thing to realize will be a vehicle simulator (**RPiEV**) enabled to BPT. Once it works fine, it is possible to also implement BPT functions on the column. In this chapter, it will be described the choices that have been made to identify the various hardware components and how we have intervened at the software level to create this new protocol that we can define as proprietary and that has been called **ModRISE**.

5.1. Hardware

There are several ways to realize the EV, all with a common matrix: a hardware block that emulates the behavior of the EVSE through a software and that is equipped with an interface, physical or virtual, to facilitate the introduction of decision variables such as the remaining battery level, the time left before stopping charging and whether or not the user consents to the use of this service; we will refer to this interface under the name of HMI (Human-Machine Interface). One of the possible solutions would be to run the software on a PC and put it in communication with the EVSE through an Ethernet-PLC bridge that, as the name suggests, takes incoming Ethernet packets and forwards them over cable via PLC to emulate the real behavior of the EV, where the signal travels on the CP. This solution would be extremely simple and easy to navigate, since the bridge identified in a preliminary phase, the INSYS POWERLINE GP (shown in Fig. 5.1), is designed for this purpose.



Figure 5.1: Enabler for Intelligent Charging INSYS Powerline GP.

It establishes point-to-point communication between the EVSE and the EV so that the EVCC can communicate with the SECC via Ethernet. In fact, it operates at the first and second level of the ISO/OSI [2] stack and, if used in combination with the official software of the standard, RISEV2G (which we will meet in the next section), communicates with its lowest level, the V2GTP layer; unfortunately, although it seems to be the best solution, it turns out to be a difficult road to walk due to the high cost of the product, about 500€ (550\$ more or less).

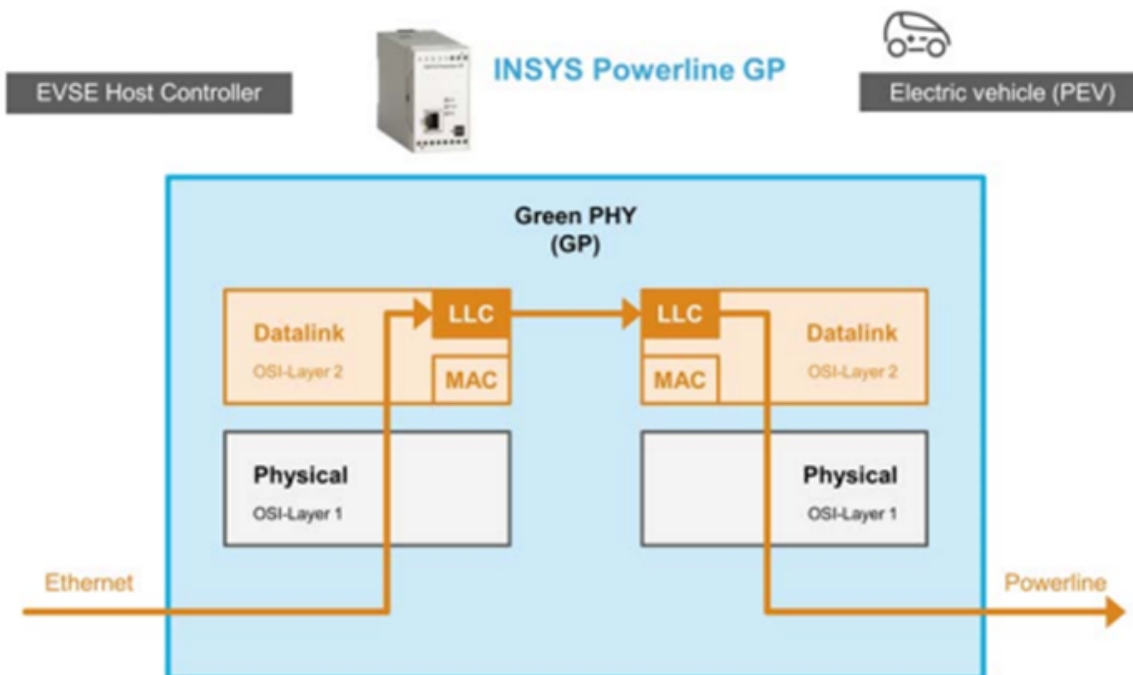


Figure 5.2: INSYS Powerline GP working scheme.

To reduce costs, therefore, it is possible to use a RaspberryPi, a single board low costs and small size computer, no bigger than a credit card. It can be used both as a PC and, if equipped with the right hardware, as a bridge; in other words, you can both load the software on its ROM, and let it run by itself, or run the software on a PC and use Raspberry as a bridge to the outside, just like in the previous case. Since the final idea would be to obtain an object that can be contained in a pre-existing structure, it was necessary to reduce the spaces and make it stand alone, thus opting for the first configuration.

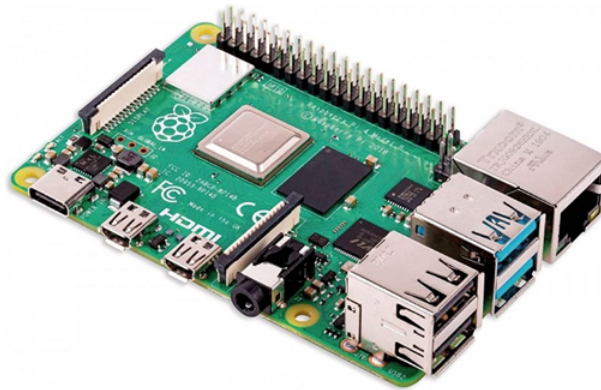


Figure 5.3: RaspberryPi 3B+.

The choice of the model fell on the RaspberryPi 3B+ because it integrates some features convenient or essential for communication with other devices. In particular, this device allows you to use the SPI serial communication protocol that will be indispensable for interfacing with another hardware component that we will see shortly, the PLC module. In addition, this version of RPi has a built-in Wi-Fi module that allows you not to waste a USB port for an external network card, saving costs and space. The latter feature is not essential for the operation of the architecture, but is very convenient in programming, configuration and use of devices, as well as deploying code. While the implementation of INSYS would be limited only to the connection of wiring and configuration, choosing an RPi there are several scenarios. One option would be to buy one of the commercially available HAT's such as EVSE-HAT or Raspberry Pi EVSE Hat, but at the time this document was drafted, presented two problems: the first was out of production, while the second was not equipped with a PLC module. Rather than waiting for the production of the first card and rather than spending time and energy designing one, I decided to use a PLC module from I2SE (now In-Tech); the advantages are multiple, starting from a saving in time and cost, the convenience of having it already present in the company, as it is already mounted on the S&h pilot cards; moreover, it has a high educational

impact. The module used (shown in the Fig.(5.4)) is the PLC StampMini2 and allows an application access to powerline communication.



Figure 5.4: PLC module based on HomePlug Green PHY.

Equipped with a QCA7000 chip, it enables point-to-point and multi-point connections as needed. Just like INSYS, it allows to retransmit incoming Ethernet packets to the powerline. This type of module is available with two types of serial communication, SPI and UART, choice that must be made when ordering; the modules present in the company use the SPI protocol and this was a fundamental constraint for the choice of Raspberry to buy. To physically connect the PLC module to the Raspberry it can be created a board that acts as an adapter, elegant solution, less bulky and more stable, or use a breadboard recreating the circuit on it; generally the best approach would be to first build and test the circuit on breadboard and then create the unique board, but due to its educational prototype nature, the work stopped at the first step; the creation of the real prototype is referred to a later and more in-depth study of the system.

5.2. Connection and environment

Once the hardware has been chosen, two RaspberryPi 3B+ and two PLC modules StampMini2, it is time for wiring and software configuration. As for the wiring, the interest is in connecting, first of all, the RPi to their respective PLC modules; to do this a study of the pinout of the devices is needed and it has been reported in Fig.(5.5).

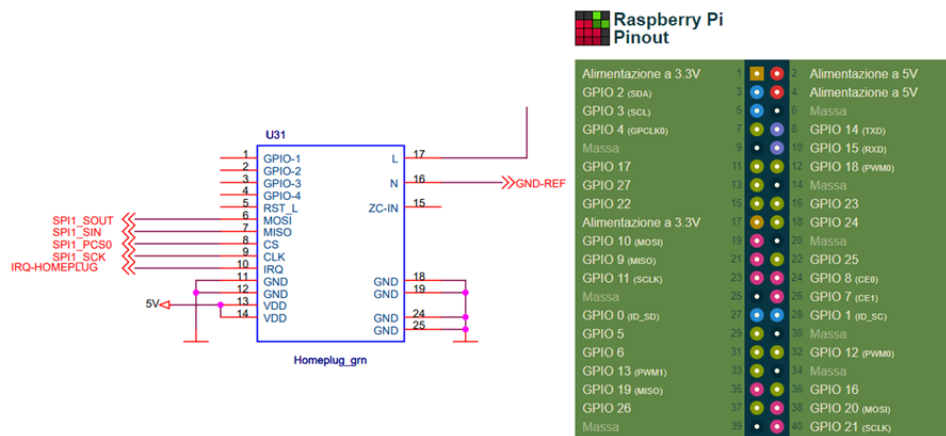


Figure 5.5: Pinout of: a) PLC Module, on the left, b) RPi, on the right.

As already mentioned, the serial communication protocol used by these two cards to exchange information is SPI. To meet the communication requirements, data must travel on 4 wires (in some cases 3 wires are enough): MISO (Master Input Slave Output), MOSI (Master Output Slave Input), CS (Chip Select) and CLK (Clock). The first two wires are dedicated to data transmission, the CS allows to choose the Slave to interact with during that specific communication, since there is the possibility of controlling several Slaves with only one Master, and, finally, the Clock is used to synchronize the rising and descending fronts. In this phase it must be chosen one of the four modes of the SPI communication, which can be SPI0, SPI1, SPI2, SPI3, depending on how the values CPOL (Clock Polarity) and CPHA (Clock Phase) are set; this choice impacts, as is easy to guess, how the clock signal is generated and, consequently, how the data will be read. Usually, changing this setting on the slave is not allowed, so the master must adapt. In the present case, the datasheet says the PLC module communicates with SPI3 and it follows that, on RPi which is the master, CPOL and CPHA must be both equal to 1 and we will shortly see how to instruct the RPi with this information.

Modalità	CPOL	CPHA
0	0	0
1	0	1
2	1	0
3	1	1

Figure 5.6: SPI mode: for each configuration data are read on the raising or falling edge and clock signal starts high or low.

From the datasheet of the PLC module, in addition to the information on the communication mode, we also find information on the transmission speed that should not exceed $12MHz$; in our case the signal has been set to a speed of $10MHz$ obtaining excellent results. Before moving on to the software configuration of the devices, we complete the wiring by adding the interrupt signal, which in Fig.(5.5) is called IRQ, then passing in parallel the power supply of the Raspberry (GND and 5V) to the module and finally going to extract the signals L and N (pin 16 and 17 of the module) that will be the CP and PP and will be connected to the CP and PP extracted from the other module. For what concerns software, all configuration takes place within the RaspberryPi, adapting it to the preset needs of the PLC slaves. In details, the first step is to get a microSD card and a PC, using the Raspberry Pi Imager software (downloadable from the official RaspberryPi page, section "Software") and load the operating system, in our case Raspbian.



Figure 5.7: RPiImager: allows to write OSs on a SD card both suggested by the software or downloaded by internet.

Once the procedure is complete, connect the Raspberry Pi to a monitor via HDMI cable, insert the mouse and keyboard into the USB ports and insert the microSD card that will act as a hard disk. The actual configuration can begin and the first step is to connect the RPi to the network and assign it a static IPV4 address so that it can always be reached on the network; for this purpose, it is important to ensure that the routing rules within the network allow the detection of other devices locally. Once you have assigned the static address, it is time to enable the SPI, SSH and VNC interfaces; the first is indispensable for the operation of our architecture, as is obvious from the previous paragraph, while the other two are services that are used for programming and using the device. In more detail, by activating the Raspberry SSH server and an SSH client on a PC or smartphone, it is

possible to operate it without physically connecting an external monitor or peripherals; the use of this service is limited to the terminal, but it can be still done any task like download packages, install software, upgrade the system or make files and folders. The other service, the VNC, allows the same degree of freedom as the previous one, but equips the user with a graphical interface, in order to see on the PC the same things that were seen previously, directly from the PC without direct access to the device. The use of these services is important from the point of view, already presented, of inserting the prototype into an existing system, where space is limited and there is no possibility of using hardware peripherals such as mice, keyboards and external network cards. From here on it is possible to work on the PC by connecting with one of the modes described above. At this point it is necessary to tell the RPi that on the SPI serial interface there is a Slave, which communicates in SPI3 and with a speed of $10MHz$; since the chip mounted on the PLC module is a QCA7000 that is a widely used device in this field, in the Linux libraries of the Raspberry is already present the driver for its management, so the only step needed is to import it inside the "overlay tree", the part of the operating system that keeps track of the drivers in use; to do this, simply open the text file `/boot/config.txt` and, in the field of the SPI interface, add the following line of code:

```
dtoverlay=qca7000 , int_pin=5, speed=10000000
```

where `dtoverlay` specifies the chip in use by the slave, `int_pin` indicates the physical pin on the RPi to which the interrupt signal is connected (the IRQ in Fig.(5.5)). This operation can be done remotely, via SSH and VNC, or locally, using the terminal or graphically, with any text editor present or not on the machine (e.g. Vim, Gedit, Nano etc.). Now that the Raspberry knows the slave, what is required is the download and installation of the right version of Java so that the software can run, then, finally deploy the software from the PC to the RPi card. The first goal is very easy to achieve, just know which version to download: in our case, the correct one is Java8, because it is the one used by RISEV2G; to perform this operation from the terminal simply run the command:

```
$sudo apt install openjdk-8-jdk
```

although, again, the same operation can be done graphically. The configuration is completed, it's time to write the software that will command its working and to deploy it on the RPi; in the next chapter we will see what software has been chosen as the skeleton of the project, what changes have been made and why.

5.3. Communication test

During an early phase of the project, a performance study on the communication just build reported serious problems: it was very slow and unstable, with a high rate of packet loss, and this caused the software to malfunction. By inspecting the hardware and the connections with one of the oscilloscopes present in the company, it was clear that a hint of communication was present, but produced an incomplete and noisy signal; thanks to the use of multimeters and the oscilloscope itself I was able to make improvements: it has been identified and replaced an unstable cable that made a false contact, a small resistor was added to impose the same voltage reference between all GNDs and the circuit was simplified and lightened by a large number of cables that only increased the equivalent resistance of the system.



Figure 5.8: The oscilloscope shows the result of the communication before the changes.

In detail, at first each Raspberry was powered by a 10W (5V, 2A) microUSB power supply and these, in turn, fed the PLC modules. While the voltage is imposed by hardware, 5V for the PLC module and 5.1V for the Raspberry, the current varies, of course, according to the number of devices to be powered; since the Raspberry Pi 3B+ consumes about 350mA in idle and 980mA at 400% of CPU usage, we can estimate a consumption of about 665mA for 200% usage. Since the PLC modules, according to the data-sheet, have a consumption of 0.5W each, we arrive at a total of 6.8W that we can round up to 7W if we consider the voltage losses in the wires. With the updates made, by connecting everything in parallel, I was able to extract only two cables that would have fed the entire system; to provide a sufficient current to the whole system, it was powered by a 5V and 3A set bench power supply, voltage subsequently brought to 5.4V to compensate for voltage drops due to the resistance in the wires; with these values of voltage and current, the maximum output power is 15.6W, well above the previously calculated operating power. A last change was made at the end of the work, to reduce more voltage drops on RPi and this allowed

to set the power supply to 5.2V and 3A, further reducing consumption. Below is the performance of wired communication between the EV and EVSE, both consisting of an RPi and a PLC module, before the changes were made.

```

95 --- fe80::2009:298f:d056:4510 ping statistics ---
96 59 packets transmitted, 26 received, 55.9322% packet loss, time 431ms
97 rtt min/avg/max/mdev = 6.026/6.997/13.492/2.157 ms

```

(a) EVSE unidirectional ping result.

```

59 --- fe80::cc38:3350:6ed6:56a9 ping statistics ---
60 60 packets transmitted, 56 received, 6.66667% packet loss, time 1019ms
61 rtt min/avg/max/mdev = 6.213/6.754/13.154/1.168 ms

```

(b) EV unidirectional ping result.

```

33 --- fe80::f26b:bb07:20f8:4e2b ping statistics ---
34 59 packets transmitted, 30 received, 49.1525% packet loss, time 290ms
35 rtt min/avg/max/mdev = 6.117/6.985/16.998/1.938 ms

```

(c) EV bidirectional ping result.

Figure 5.9: Before fixing communication, 64 packets have been send by both sides and performances have been stored. During (a) and (b) only one machine were performing this test, in (c) they both were transmitting.

As a result of the improvements, the packetloss was restored around 0% ensuring excellent stability and good performance.

5.4. Software

It has been said that, whatever the choice of hardware, it must be controlled by a software. The purpose of the latter is to emulate the operation of an electric vehicle or a charging station, depending on the board on which it is running. Although the software used was RISEV2G, it is also worth describing a second software, OpenV2G, as it will be useful in future developments. RISEV2G, in fact, being written in Java, cannot be run on a common microcontroller, only on Linux-based ones, because it only recognizes the C language and some of its derivatives. Being OpenV2G written in C, however, makes it compatible with the controllers used by S.&h. boards, so this software will be the skeleton to follow during the porting from Java to C.

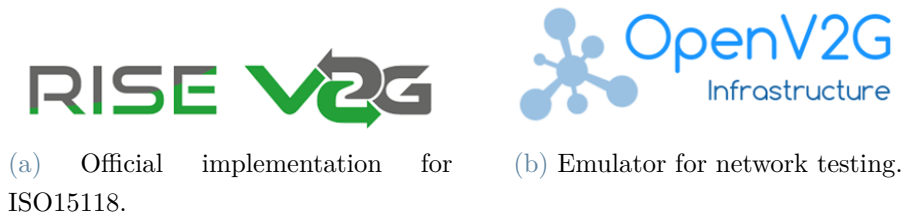


Figure 5.10: (a) will be the base of all the work, it will be modified adding function and messages useful for the project; (b) will be used for future developments.

5.4.1. RISEV2G

This software, made by Marc Mültin and his staff, is, according to their official webpage on GitHub (SwitchEV/RISEV2G), «the Reference Implementation Supporting the Evolution of the Vehicle-2-Grid communication interface ISO 15118. [...] It allows for a user-friendly "Plug And Charge" mechanism for authentication, authorisation, billing, and flexible load control based on a wide set of information exchanged between the EV and EVSE. [...] The RISE V2G project serves as an open source standard-compliant reference implementation and documentation. As such, the objective is to provide a test platform for interoperability testing and to offer an information resource for interested parties.»

One of the most interesting features of this program is that it integrates the top five levels of the stack ISO/OSI as defined by the standard 15118-2, Fig.(5.11).

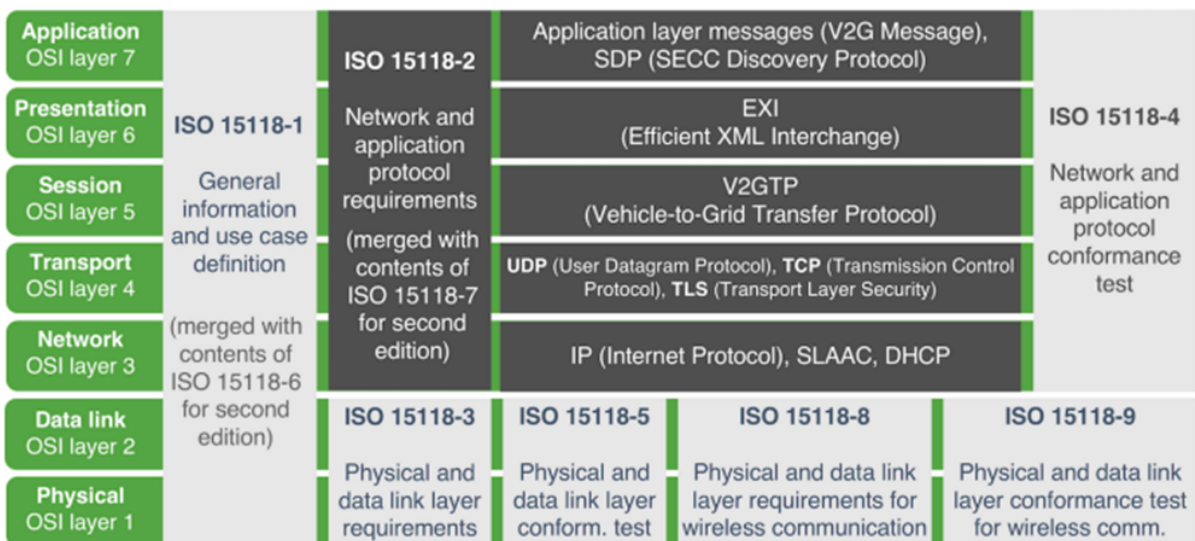
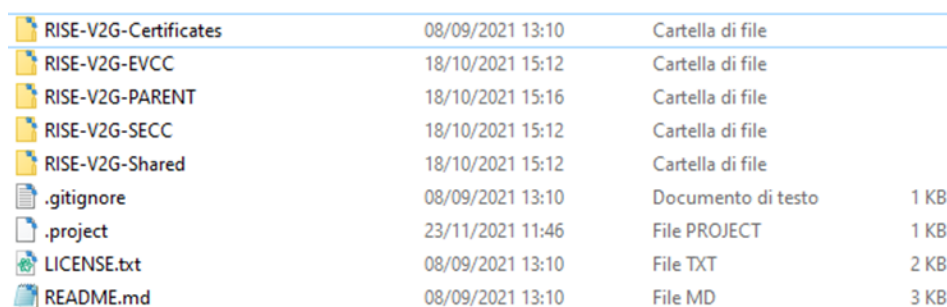


Figure 5.11: Network layer in V2G infrastructure.

[17] The highest level is the Application Layer, where all messages, V2G and SDP messages, encoded in an XML format, are elaborated; the above level is the Presentation Layer, so called EXI, where all messages are compressed in the EXI format. Data are sent to the V2GTP Layer which handles communication sessions between the EV and EVSE; in the fourth level of the ISO/OSI stack a TCP connection needs to be established between them in order to reliably exchange data packets via the communication link and guarantee eventual error recovery and retransmissions. Although such communication is stable and reliable, messages are not only sent via TCP, indeed, SDP messages are sent via UDP for convenience; to ensure security communications, also the TLS protocol is used in some cases. Last layer is the IP and it is used by TCP to assign unique IP address for the EVCC and SECC. This open-source project is composed, as shown in the figure below, by three main nested projects: EVCC, SECC and Shared.

As it is immediate to understand looking at the name, the first and the second ones emulate the whole process for, respectively, the EV and the EVSE, while the third one defines all shared functions and messages; this structure allows to export only the desired part of the code without useless loss of memory.



RISE-V2G-Certificates	08/09/2021 13:10	Cartella di file	
RISE-V2G-EVCC	18/10/2021 15:12	Cartella di file	
RISE-V2G-PARENT	18/10/2021 15:16	Cartella di file	
RISE-V2G-SECC	18/10/2021 15:12	Cartella di file	
RISE-V2G-Shared	18/10/2021 15:12	Cartella di file	
.gitignore	08/09/2021 13:10	Documento di testo	1 KB
.project	23/11/2021 11:46	File PROJECT	1 KB
LICENSE.txt	08/09/2021 13:10	File TXT	2 KB
README.md	08/09/2021 13:10	File MD	3 KB

Figure 5.12: This project is composed by two more project which implement the operation of SECC and EVCC separately.

Moreover, there is a folder, RISE-V2G-Certificates in Fig.(5.12), which contains files to generate certificates useful for TLS protocol, option that could be enable or disabled during debug by settings. Talking about settings, both EVCC and SECC projects contain a **.properties* file (shown in Fig.(5.13)) where it is possible to choose whether to show messages in EXI and/or Hex format, which controller, energy transfer mode and network interface are allowed, which one to use and so on.

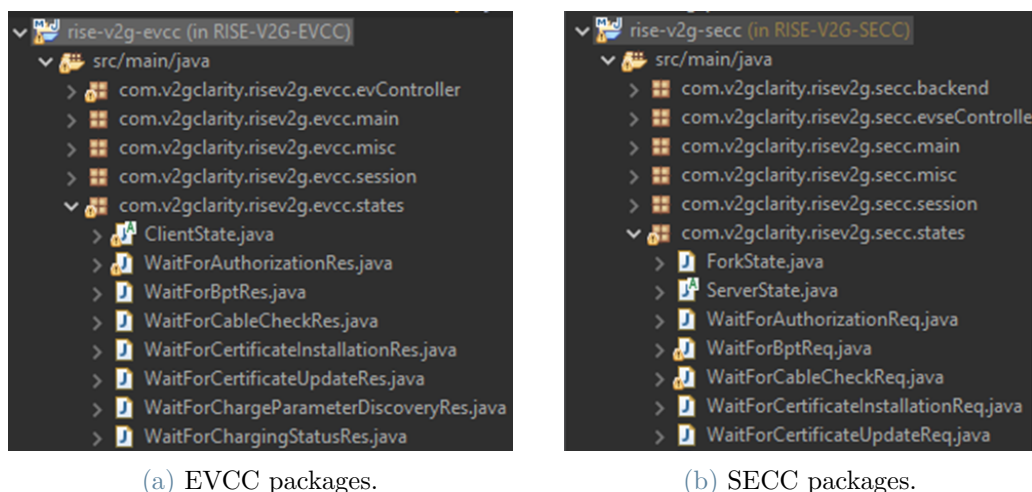

```

EVCCConfig.properties
1 #
2 #Tue Nov 16 17:23:40 CET 2021
3 exi.messages.showhex=false
4 implementation.evcc.controller=com.v2gclarity.risev2g.evcc.evController.DummyEVController
5 exi.messages.showxml=false
6 voltage.accuracy=5
7 network.interface=30
8 session.id=00
9 tls=false
10 authentication.mode=
11 signature.verification.showlog=false
12 exi.codec=exifcient
13 energy.transfermode.requested=DC_core
14 contract.certificate.update.timespan=14

```

Figure 5.13: EVCC configuration file, also accessible from the HMI, Chapter 5.5.

For sure, one of the most important part of RISEV2G is the states package. In this package are described, for both EVCC and SECC, all the states defined by the ISO15118-2 and, for each state, it is implemented the correct behaviour the EV and EVSE should undergo. It may not be as immediate, but it can be seen in Fig.(5.14a) and Fig.(5.14b) that in this system EV always sends request type messages and the EVSE responds with response type ones; once SECC receives a request message, the software enters the relative WaitFor<Message>Req state and elaborates it, taking actions and sending a response message, if required; same on the other side.



(a) EVCC packages.

(b) SECC packages.

Figure 5.14: EVCC and SECC packages; inside the extended states packages are visible some of the classes implementing the states of the standard.

The whole state machine is shown in the Appendix and it can be used to follow all steps of a charging process which, however, has been already described in Chapter 4.2.

In that chapter, it has been presented the charging process by the time the SECC received the supportedAppProtocolReq, but nothing has been said about what happen before. Basically, before connect the EV, on starting, the SECC opens a TCP and a UDP channel and starts listening to vehicle connection requests. When the EV is plugged, it starts a multicast transmission of UDP packets (Fig.(5.15)) with the aim of finding an active SECC on the same network.

```

pi@RPiSECC:~/Desktop/SECC $ java -jar rise-v2g-secc-1.2.6.jar
2021-11-24T11:44:03,269 INFO [main] UDPSEServer: UDP server initialized at link-local address fe80:0:0:6cc1:67c4:a034:1d8e%wlan0 and port 15118
2021-11-24T11:44:03,372 ERROR [main] SecurityUtils: Keystore file location './seccKeystore.jks' not found (FileNotFoundException).
2021-11-24T11:44:03,373 ERROR [main] SecurityUtils: Keystore file location './seccTruststore.jks' not found (FileNotFoundException).
2021-11-24T11:44:04,798 INFO [main] TLSServer: TLS server initialized at link-local address fe80:0:0:6cc1:67c4:a034:1d8e%wlan0 and port 58384
2021-11-24T11:44:04,803 INFO [main] TCPServer: TCP server initialized at link-local address fe80:0:0:6cc1:67c4:a034:1d8e%wlan0 and port 49864
2021-11-24T11:44:09,228 INFO [TCPServerThread] TCPServer: Waiting for new TCP client connection ...
2021-11-24T11:44:09,228 INFO [TLSServerThread] TLSServer: Waiting for new TLS client connection ...

```

Figure 5.15: Terminal view of RISEV2G emulating an EVSE waiting for UDP message from one connected EV.

The EV sends 50 packets after which, if no EVSE is found, the process is terminated; otherwise, if the connection is established, the handshake process starts, beginning with the supportedAppProtocolRequest and continues following the ISO15118-2 standard as described in Chapter 4.2.

```

pi@RPiEvcc:~/Desktop/EVCC $ java -jar rise-v2g-evcc-1.2.6.jar
2021-11-24T11:45:49,412 INFO [main] V2GCommunicationSessionHandlerEVCC: Security level TCP was chosen
2021-11-24T11:45:49,482 INFO [main] UDPClient: UDP client initialized at address fe80:0:0:f7e:2389:5383:af2%wlan0 and port 61956
2021-11-24T11:45:49,489 DEBUG [main] V2GCommunicationSessionHandlerEVCC: Preparing to send SECCDiscoveryReq ...
2021-11-24T11:45:49,493 DEBUG [main] UDPClient: Message sent
2021-11-24T11:45:49,745 ERROR [main] UDPClient: A SocketTimeoutException was thrown while waiting for input stream from UDPSEServer
2021-11-24T11:45:49,746 WARN [main] V2GCommunicationSessionHandlerEVCC: Number of SECCDiscoveryReq messages so far: 1
2021-11-24T11:45:49,747 DEBUG [main] V2GCommunicationSessionHandlerEVCC: Preparing to send SECCDiscoveryReq ...
2021-11-24T11:45:49,748 DEBUG [main] UDPClient: Message sent
2021-11-24T11:45:49,999 ERROR [main] UDPClient: A SocketTimeoutException was thrown while waiting for input stream from UDPSEServer
2021-11-24T11:45:50,000 WARN [main] V2GCommunicationSessionHandlerEVCC: Number of SECCDiscoveryReq messages so far: 2
2021-11-24T11:45:50,001 DEBUG [main] V2GCommunicationSessionHandlerEVCC: Preparing to send SECCDiscoveryReq ...
2021-11-24T11:45:50,002 DEBUG [main] UDPClient: Message sent
2021-11-24T11:45:50,253 ERROR [main] UDPClient: A SocketTimeoutException was thrown while waiting for input stream from UDPSEServer
2021-11-24T11:45:50,254 WARN [main] V2GCommunicationSessionHandlerEVCC: Number of SECCDiscoveryReq messages so far: 3
2021-11-24T11:45:50,255 DEBUG [main] V2GCommunicationSessionHandlerEVCC: Preparing to send SECCDiscoveryReq ...

```

Figure 5.16: Terminal view of RISEV2G emulating an EV connected to an EVSE that starts sending UDP packets and waits for any response.

5.4.2. OpenV2G

OpenV2G has the same purpose of the previous project, but it is more focused on the PKI (Public Key Infrastructure), so on what concerns security of this protocol. However, this software can be very useful because it is written in C language, that means it can be implemented on a microcontroller not Linux-based, representing the vast majority. Unlike RISEV2G that generated two different terminals one for the EVCC and one for the SECC, here there is only one which shows the whole interaction between the two endpoints (Fig. (5.17)), instead of having one terminal for each entity. The output code

is much less verbose of the other program and it looks more simple and direct; the reader can compare the output of these two programs looking at the Appendix. Obviously, the message flow is the same for both because they implement the same standard and due to that we can define the same handshake, charging phases and so on.

```

+++ Start application handshake protocol example +++
EV side: setup data for the supported application handshake request message
EV side: send message to the EVSE
EVSE side: List of application handshake protocols of the EV
    Protocol entry #=1
        ProtocolNamespace=urn:iso:15118:2:2010:MsgDef
        Version=1.0
        SchemaID=1
        Priority=1
    Protocol entry #=2
        ProtocolNamespace=urn:din:70121:2012:MsgDef
        Version=1.0
        SchemaID=2
        Priority=2
EV side: Response of the EVSE
    ResponseCode=OK_SuccessfulNegotiation
    SchemaID=1
+++ Terminate application handshake protocol example with errn = 0 +++
+++ Start V2G client / service example for charging (IS01) +++
EV side: call EVSE sessionSetupEVSE side: sessionSetup called
    Received data:
        Header SessionID=0 0 0 0 0 0 0 0
        EVCCID=10
EV side: received response message from EVSE
    Header SessionID=1 2 3 4 5 6 7 8
    ResponseCode=0
    EVSEID=20
    EVSETimeStamp=123456789
EV side: call EVSE ServiceDetail

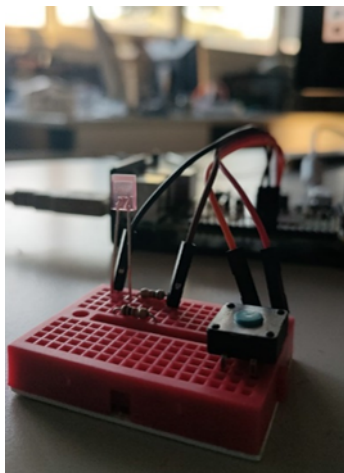
```

Figure 5.17: Terminal view of an OpenV2G session.

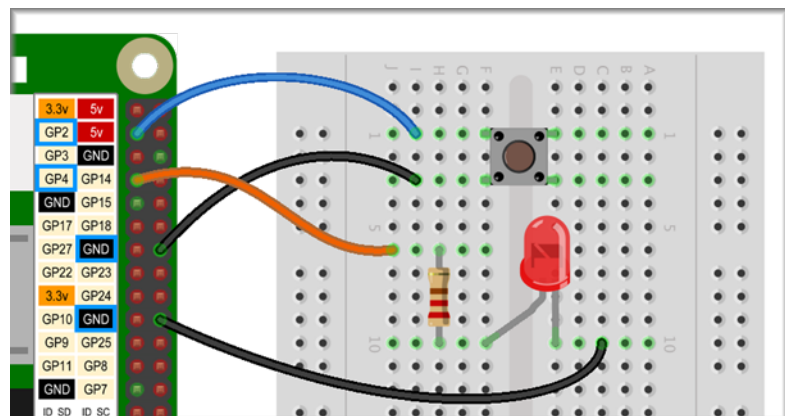
Although this two software look similar, there is a fundamental difference: instead of implementing the last 5 levels of the stack ISO/OSI, OpenV2G only embed the last three of them; this means that V2GMessages and SDP messages are defined in the higher level, then they are compressed in the EXI Layer and are sent to the V2GTP Layer. From here on, coding layer is left to developers which will build the lower architecture implementing the TCP/IP structure and Data and Physical Layer using the Ethernet protocol. What is clear so far for what concerns the software is that RISEV2G is better coded, implements the ISO standard in the best way possible and allow to separate EV and EVSE projects easily, but, on the other hand, it cannot be deployed on microcontrollers; OpenV2G is a little "worse coded", but it is an optimal starting point for developing the software on a microcontroller. They are both useful due to their peculiarity, but in this phase of the work only RISEV2G has been used while OpenV2G will be necessary to speed up the coding phase on the S.&h. board.

5.5. HMI

For the realization of the HMI there are two basic solutions: a hardware or a software interface. Since one does not exclude the other and being hardware implementation the most immediate, I preferred to start with this and then switch to a digital HMI. Unfortunately, the realization has never gone beyond the prototype shown in Fig.(5.18) but it was thought to be equipped with LEDs which would have shown the actual state of the standard IEC 61851 in real time, buttons for sending some commands and performing certain functions and a LED strip showing the direction of current. The prototype in Fig.(5.18) consisted only of a button and an LED and was intended to test the libraries and the general interface between Java, its libraries and hardware. The first step was to write a code that would allow to verify the correct interface between button and Raspberry, regardless of the emulator; to do this, two open source libraries were used taken from GitHub, famous among developers and lovers of DIY: Pi4j and WiringPi.



(a) Prototype of the physical HMI.



(b) Electric diagram of Rpi-HMI connection.

Figure 5.18: First prototype of HMI and its diagram.

The only difficulty at this stage was the compatibility of Java and the various libraries with the hardware due to new versions of programs and firmware: the solution was to make a downgrade of the firmware of Raspberry and use an old version of Pi4j, the V1.3, the last to support the JDK1.8 (the same as RiseV2G). Having found that the test software was working, it was integrated into a version of the RISEV2G software in the form of an additional class that, for coding convenience, was put in the same package as the EVCC “Main” class. Going on with the work, however, the physical implementation was increasingly limiting and quite useless; the idea of a physical command, therefore,

was soon shelved in favor of a virtual interface. The advantages in doing that are multiple because it opens the way to a multifunction control, a simplicity in the use of the software, allows to lay the foundations for the development of a web interface and much more.

To realize this new part of the project, it has been done some research to understand which was the best graphic library for this project, but, having no particular claims or needs, it has been opted for the most used: Swing; before downloading and importing it, it has been downloaded a jar file provided by the developers in order to test the graphics of the library and, once satisfied with the result that it would be obtained, it has been imported into the project. Since the Rise project is made using Maven, it was not necessary to manually download anything, it was enough to add the dependencies to this library inside the *pom.xml* file, the rest is done by Maven itself, as well as the management of the program build and the creation of *jar* executables. The starting point was to manually program all the objects provided by the library, placing them on the screen through lines of code and defining in the same way their size, their color, the events to which they are associated and the functions that each element must perform. Unfortunately, however, the only visual feedback available used to appear only after the compilation of the program and this implied that to place an object inside a window or to verify the color and all the graphic aspects, the program needed to be run lots of times. To overcome this problem, it has been opted for a framework for the creation of graphical environments, *WindowsBuilder* (in Fig.(5.19)), a plugin for Eclipse that also implements Swing, allowing me to reuse the code written until then. The operation of this add-on is very simple: from a menu select the object that should be inserted in the building window, drag it inside the frame and place it in the desired position.

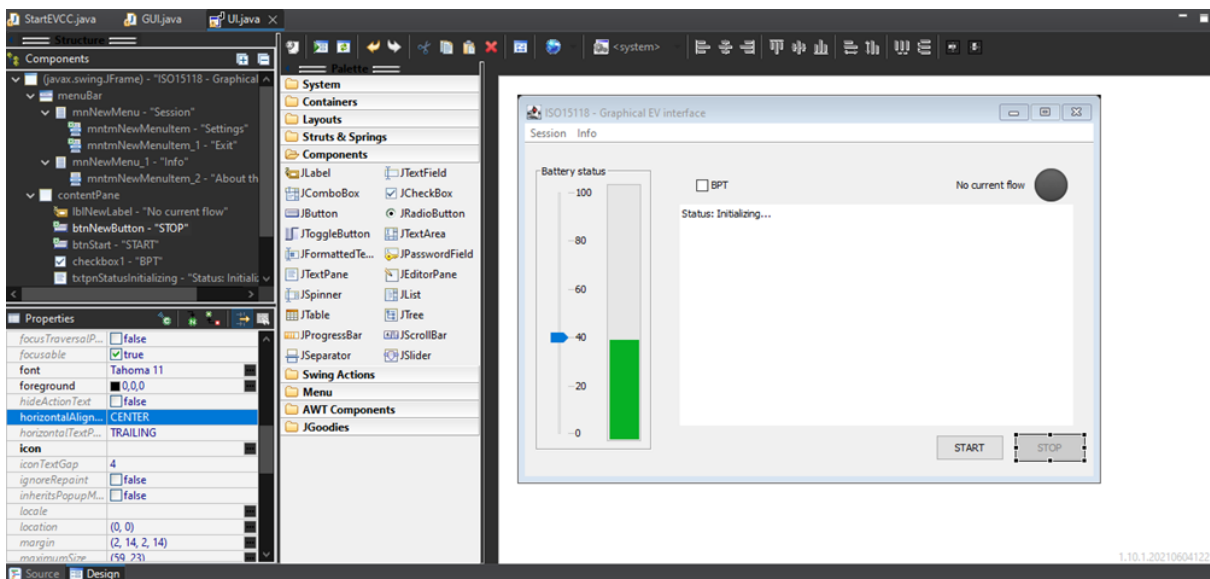


Figure 5.19: WindowsBuilder interface.

At this point it can be modified at will by changing the size, color, adding an icon, text and much more. When the interface aspect is satisfactory, it is possible, by selecting the object, to add *event detectors* with a simple click; in this way the plugin refers to the Java code that, in the meantime, has been written in background and there it can be written the desired functions. Now we come to the advantages and features of the graphical interface that has been designed (Fig.(5.20)).

First of all, the fundamental thing is the slider on the left: thanks to that, it can be changed the residual battery value, allowing to simulate different scenarios quickly and easily. To this slider has been connected an *ActionListener* that captures the changes in the value of this object and updates the battery value in the program; this aspect is fundamental because in the next paragraph the battery level will be a very important data for the logic of communication. Inside the same panel of the slider, there is a progress bar that represents the previous value in a qualitative way coloring red, green or yellow. Proceeding from left to right, it can be found a panel called "Change state" in which it is possible to choose whether the program will run the machine in autonomous states or if it will need the operator to intervene manually.

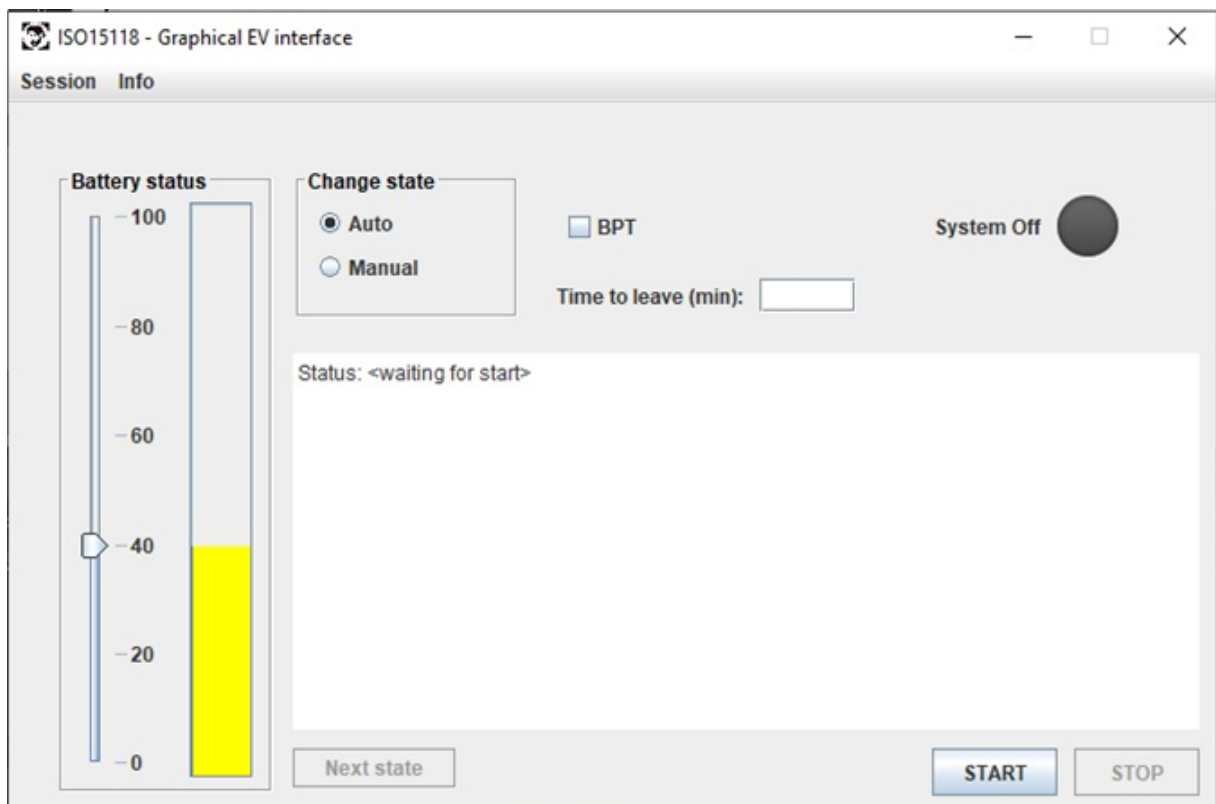


Figure 5.20: Software HMI.

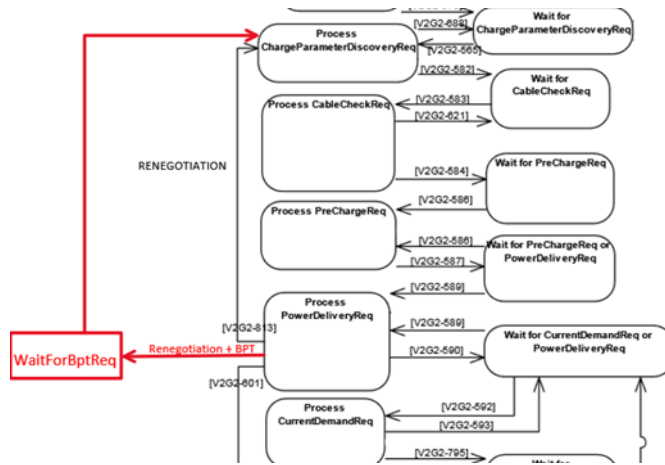
Choosing the "Auto" setting, the states will not be looked at in the same way as the

original program, but a delay between one state and another of 1s has been inserted to facilitate the reading of the interface; this option could be adjusted in the future depending on the need if this software were to be implemented on a real machines. Moving to the top center of the interface, we find a checkbox called "BPT": it allows, as is easy to guess, to enable or disable the BPT; every change of its state is interpreted by the program, which will make decisions based on the logic which it was written with and which it will be seen in the next section. In order to make the interface intuitive and quick to interpret, it has been equipped with a light, in the upper right corner, that allows to control the direction of current. At the beginning of the program, it is turned off and this indicates that there is no current, but the same condition occurs, in general, whenever you are not in *STATE C* of the IEC standard; when you switch, instead, in this state, the indicator lights up red, if the vehicle is absorbing current or green if the vehicle is providing power. To make the logic a little more complex and close to reality, a parameter has been introduced that indicates the time left for the vehicle to complete the charging cycle. This value, in the real system, will be calculated by the EVSE as a subtraction between the end of charging time set by the customer and the time when schedule is provided; here, for simplicity, it can be set via textbox. Below this element there is another, not writable, textbox where you can view the charging information; not having access to the vehicle data, it only shows the status of the ISO15118-2 standard you are in, but for future implementations, it is possible to think of also showing current and voltage output values, the time remaining at the end of charging or any other parameter may be interesting. At the top of the screen, finally, there are two menus: Session and Info. From the first you can access the properties of the vehicle in order to carry out all the operations without having to look for the project folder inside the PC, very useful in the design phase, but definitely to be improved in view of a future implementation. The last item, surely less important and inserted with the only didactic purpose and completeness of presentation, is the section "Info", where it is possible to find information about the project.

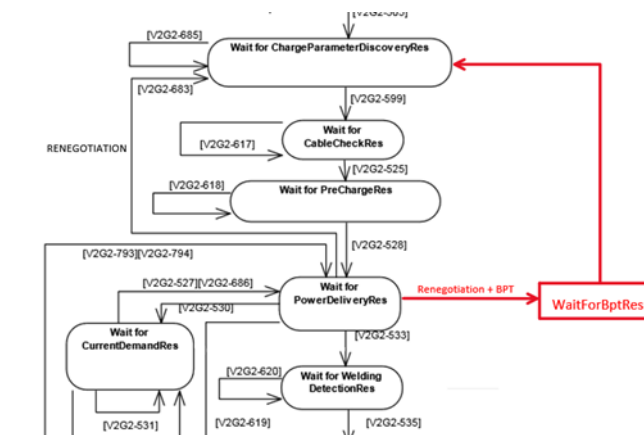
5.6. Software design and implementation

Now that we have clear in mind what these software do, all passages explained in Chapter 4.2 and how the UI works, it is time to see how the state machine has been modified to insert bidirectionality in the flow. What can be seen in Fig.(5.21) is a cleaner version of the states machine, the reader can consult the complete one in the Appendix of this document. In ISO15118-2, once the handshake is over and all parameters have been exchanged by the ChargeParameterDiscovery pair of messages, when the machine is ready to send the PowerDelivery message, the charge is allowed to start and it will remain

inside the WaitForCurrentDemand loop since a PowerDelivery message or other type of interrupting messages are sent. The upgrade, as shown in Fig.(5.21) consists on adding an extra state called WaitForBpt, reachable from PowerDelivery. This additional state has been thought for academic purpose, but in a real application or in an advanced stage of this work, it will be quite useless and it will only introduce an undesired delay, so it would be removed and bypassed keeping only the condition under which this state has been visited. What has been designed during this project has to be intended as the base for future work, so what is following has been defined with the purpose to open an existing software to future developments and that is the reason why it has been defined parameters or variables that will not be implemented or used at this stage.



(a) WaitForBptReq state.



(b) WaitForBptRes state.

Figure 5.21: State machine modified to introduce BPT.

Talking about variables and parameters, as we saw in Chapter 4.2 to create states and messages there are certain structures, the so called “*schemas*”, that must be respected and, indeed, the first goal was to design these structures for BPT messages. In Fig.(5.22)

we can see how each message, both response or request, has been thought, but, again, this is only a choice made during designing (the number of these values can increase, decrease or they could be substituted by more interesting parameters).

More in detail, both messages are equipped with the *ResponseCode* that, as in all messages of the norm, helps to intercept any failures of the system and allows to take decisions such as, for example, the interruption of the session following an error or a customer decision. The introduction of this parameter was indispensable for the general operation, while for what concerns the others, they have been added because useful to the logic of the software. An important thing to say is that the choice of parameters is closely linked to the operation of the system and how the designer thought it; having clear in mind what information should exchange the EVSE and the EV is a fundamental prerogative. In the present case, since the EV sends requests, it will also send the BPT request and, together with this, may want to send certain parameters so that the column can make optimal decisions. EVSE receives from EV all the values that have been chosen by the user, as seen in Chapter 5.5, which in this project can be set via HMI, but later could be entered by the Web via App or site. Communicating the battery level of the vehicle, as well as the residual charging time and the choice of whether or not to enable the BPT, is crucial for the control logic and for the creation of a scheduling, while the direction can be useful to understand in which state the system is located or even just as an auxiliary indication.



Figure 5.22: BPT messages definition.

Once the request is sent, the EV receives the response from the EVSE together with the parameters it carries and which, at this stage of the design, are equal, but in smaller number, to those of the request, Fig.(5.22); as already mentioned in this paragraph, these data can and will be updated adapting them to the real case, modifying and expanding the schematic constructed in this phase. All the work done up to this point helps future developers in the design of the real system since messages were created to comply with the standard; being this a case study, the operation is slightly different: while maintaining the possibility of developing a system that allows the column to take decisions, in this

case the choice is exclusively of the vehicle, since the column is always willing to accept a change of direction in the current. For the creation of a more realistic model it would be necessary to receive information from the network or a command, coming from the aggregators, that the column can interpret, in turn, like a request of energy.

Before going through the state machine, let's understand what is the logic behind the decisions made by the vehicle. To know if the BPT is required and allowed, three elements have been introduced: the **batterylevel**, that is the value saved every time the slider changes state, the **timetoleave**, that is between how much time the battery of the vehicle will have to be at 100% and the checkbox of the **BptEnabled**, needed to choose at any moment whether to allow the EV to send BPT request.

```
87
88 public boolean isBptRequired() {
89     if (UI.getUi().checkbox1.isSelected() &&
90         UI.getUi().progressBar.getValue()>UI.getUi().batteryflg &&
91         (UI.getUi().timetoleave>=60 || UI.getUi().timetoleave==0)) {
92         return true;
93     }
94     return false;
95 }
96 }
```

Figure 5.23: This function makes the system able to understand whether BPT is required.

The batterylevel is inserted within a hysteresis cycle built between the values 20% and 80%, Fig.(5.24), therefore, if the vehicle is in charge, it will be able to start BPT only if it has exceeded 80% of charge, otherwise if it is discharging following the sale of its energy, it will not fall below a threshold of 20%; this value is indicated by a flag, **batteryflg**, whose value changes based on the function reported in Fig.(5.23), which returns the value of 20 or 80 depending on whether the software has intercepted a charging or discharging phase. By convention, it has been established that, if the vehicle started from an intermediate situation with a level of 40%, it would be entitled to begin with a session of BPT; therefore, the only case where this would not happen would be if the battery level at session start was less than 20%. As for the charging time, however, it was decided to leave the vehicle a margin of 1h, as seen in Fig.(5.23), to be able to charge; if the time set by the user is less than 60min, BPT would not be enabled even if the battery level was respected. The most careful reader may have noticed that also a timetolive equal to 0 it is acceptable and this due to a function that has been written which takes as input the text in the "timetoleave" textbox on the HMI and translate it into an integer unless blank space is left, in which case the program knows that it does not have to consider the time limits and set it to 0.

```

71  if (getCommSessionContext().getEvController().getBptEnable())
72      UI.getUi().batteryflg=20;
73  else
74      UI.getUi().batteryflg=80;

```

Figure 5.24: Function to define hysteresis cycle.

Now let us see how to go through the new state machine and analyse what are the decisional parameters of logic control. All decisions regarding whether to choose or not the bidirectional, are made within the PreCharge state which passes this information to WaitForPowerDeliveryRes state which redirect to the right following state: if the machine is enabled to the BPT (BPT checkbox checked) and conditions are respected (see Fig.(5.23)), a BptReq is sent, otherwise the graph is traced following the normal flow. The scenarios that can occur, when starting the charging session, are basically 3:

1. at the time when the vehicle is connected to the column, it already meets the prerequisites to operate the bidirectionality and can start its session discharging further in favour of the network;
2. at the time the vehicle is connected to the column, it is ready for a BPT session, but the user's choice is not to discharge the vehicle and therefore not to run the BPT; in this case, the vehicle begins its charging session and completes it without deviating from the original operation, unless the user decides to change settings;
3. at the time the vehicle is connected to the column, the car is not available for a BPT session, although the customer has given his availability.

Depending on the situation, then, various requests are sent in different sections of the code and the states machine. Let's analyse the various situations through an example; imagine to be in situation 1 and that an electric vehicle user goes to the supermarket to do the shopping, you arrive at the parking lot with a range of 40% and decide to put the vehicle to charge, although it is not completely drained. After passing the first steps of the handshake, it comes the PowerDelivery state, the EV knows (from PreCharge state) that the vehicle can be discharged and sends a BptReq, passing, so, in the state of WaitForBptRes. Similarly, the EVSE, which is located in the so-called ForkState (that is a state that allows to be in multiple states at the same time), responds with a BptRes and passes into the state of WaitForChargeParameterDiscoveryReq from which follows the flow. During these steps, when the path to BPT states is chosen, a global flag is activated to track the direction of current and it is accessible from any part of the EV side program. The flow continues through the states of CableCheck and PreCharge and

then back into the PowerDelivery from which it continues its cycle entering the charging loop of the CurrentDemand. In fact, if the BPT is active, what was a charging cycle, now it is interpreted as a discharging cycle, so the battery level drops (operation that is now carried out by hand through the interface, but in the real system will follow the physical principles of discharge) and, once reached 20%, thanks to the logic that has been implemented, disables the BPT; at this point the battery starts charging again and the cycle continues in this way unless renegotiations or errors. In case 2, on the other hand, the BPT checkbox is disabled, but the time and battery conditions are respected; in this case, as in case 3, from PowerDelivery state it goes, without changes, towards the CurrentDemand charging loop from which, as usual, it is possible to exit in case of user interaction or in case of errors. Going into more detail of what happens in the core of this standard, the PreCharge, let's see what messages and what parameters are exchanged, so that the scheme just shown can work. According to ISO15118-2, whenever there is a need to change some charging parameter, be it voltage, current, charging time or other, it must go through a "renegotiation" that leads from the PowerDelivery state to the ChargeParameterDiscovery. To activate this transition it is sufficient, from the state of PreChargeRes, to send a powerDeliveryReq(ChargeProgress) where ChargeProgress is an argument set to RENEGOTIATION; since the BPT was intended, in this document, as a special renegotiation, to keep track of this peculiarity, a specific parameter has been added to RENEGOTIATION: ChargeProgress =BPT. By sending a powerDeliveryReq(BPT), in the next state it will be kept track of this difference, it will be directed to the WaitForBpt state, and then to the ChargeParameterDiscovery; from here it proceeds as described above and, once reached the PreCharge state, a powerDeliveryReq(ChargeProgress) is sent with ChargeProgress=START. Understood how to navigate the states of this system, it remains to learn what changes are made to the charging session and what possibilities remain open for the future.

```
197 if(getCommSessionContext().getDCEvseController().getBptEnable())
198     evseMaxCurrentLimit.setValue((short) -32);
199 else
200     evseMaxCurrentLimit.setValue((short) 32);
201
202 return evseMaxCurrentLimit;
```

Figure 5.25: Function to set upgraded value of current.

At the beginning of the chapter it was clarified that, in a real case or for a future development, it would be the column itself to take decisions on BPT, but that, as regards this document, for simplicity and time, it was preferred to leave the decisions to the logic

of the vehicle; it is also true, however, that charging limits are imposed by the column, so when it receives a BptReq, activates the same flag present in the vehicle, with the purpose of intercepting this change of direction of current, and allows its controller to set new values (e.g. voltage, current) , as in Fig.(5.25).

In the same way this value is changed, it can be changed all the other maximum limits in addition to the actual values provided by the inverter. This last part, unfortunately, could not be implemented because the communication with the inverter was not regulated by RISE and adding it would have been too long work for a thesis of a few months. Below is a CurrentDemand message in *xml* format with which the two actors of the system exchange information about the status of the parameters. In particular, in the figure of left we can see the value of current before the reception of a request of BPT and in that of right we can see the change of sign.

```
<ns4:EVSEIsolationStatus>Invalid</ns4:EVSEIsolationStatus>
<ns4:EVSEStatusCode>EVSE_Ready</ns4:EVSEStatusCode></ns4:DC_EVSEStatus>
<ns4:EVSEMaximumCurrentLimit><ns4:Multiplier>0</ns4:Multiplier><ns4:Unit>A</ns4:Unit>
<ns4:Value>32</ns4:Value></ns4:EVSEMaximumCurrentLimit>
<ns4:EVSEMaximumPowerLimit><ns4:Multiplier>3</ns4:Multiplier><ns4:Unit>W</ns4:Unit>
<ns4:Value>63</ns4:Value></ns4:EVSEMaximumPowerLimit>
<ns4:EVSEMaximumVoltageLimit><ns4:Multiplier>0</ns4:Multiplier><ns4:Unit>V</ns4:Unit>
<ns4:Value>400</ns4:Value></ns4:EVSEMaximumVoltageLimit>
<ns4:EVSEMinimumCurrentLimit><ns4:Multiplier>0</ns4:Multiplier><ns4:Unit>A</ns4:Unit>
<ns4:Value>16</ns4:Value></ns4:EVSEMinimumCurrentLimit><ns4:EVSEMinimumVoltageLimit>
```

(a) CurrentDemandReq before BPT.

```
<ns4:EVSEIsolationStatus>Valid</ns4:EVSEIsolationStatus>
<ns4:EVSEStatusCode>EVSE_Ready</ns4:EVSEStatusCode></ns4:DC_EVSEStatus>
<ns4:EVSEMaximumCurrentLimit><ns4:Multiplier>0</ns4:Multiplier><ns4:Unit>A</ns4:Unit>
<ns4:Value>-32</ns4:Value></ns4:EVSEMaximumCurrentLimit>
<ns4:EVSEMaximumPowerLimit><ns4:Multiplier>3</ns4:Multiplier><ns4:Unit>W</ns4:Unit>
<ns4:Value>63</ns4:Value></ns4:EVSEMaximumPowerLimit>
<ns4:EVSEMaximumVoltageLimit><ns4:Multiplier>0</ns4:Multiplier><ns4:Unit>V</ns4:Unit>
<ns4:Value>400</ns4:Value></ns4:EVSEMaximumVoltageLimit>
<ns4:EVSEMinimumCurrentLimit><ns4:Multiplier>0</ns4:Multiplier><ns4:Unit>A</ns4:Unit>
<ns4:Value>16</ns4:Value></ns4:EVSEMinimumCurrentLimit><ns4:EVSEMinimumVoltageLimit>
```

(b) CurrentDemandReq after BPT.

Figure 5.26: CurrentDemand messages in *XML* format before and after BPT process has been triggered.

6 | Conclusions and future developments

In a world that has been fighting for years against the greenhouse effect, the ozone hole and the massive production of toxic gases from the combustion of oil, the development of green technologies is fundamental. One of these technologies on which in recent years is focusing a lot is the EV (Electric Vehicle). With the advent and spread of EVs, not only it will increase the energy demand that the network will have to meet but will decrease one of the fundamental values that until now has allowed to intervene promptly on the stability since it introduces a delay in the system: the network inertia. The current standard that regulates the operation of EV recharges in a context of V2G (Vehicle-to-Grid), the ISO15118, will be updated shortly normalizing a useful concept to lower the reaction time of the network to possible contingency: the BPT (Bidirectional Power Transfer). The BPT is a system that will allow an electric vehicle connected to the network to interrupt its charging cycle and to reverse the current flow to favour the network; this exchange will be necessary only in cases where the grid is in deficit and requires producers a greater power. The purpose of this thesis was to create an EV simulator that was BPT enabled and had a graphical interface for managing parameters in real time during a charging session; this simulator will be useful to S.&h., the company that hosted my internship, to test the new generation of EVSE stations that will project in the near future. To do this it was modified a simulation software of EV and EVSE, the RISEV2G, respecting the constraints imposed by the ISO15118-2 standard, to recognize and manage any requests of BPT; the modified software was then deployed on two microcontrollers that have to communicate with PLC modules, as required by the standard, which in turn communicate with each other as one will simulate the EV, the RPiEV, and the other the EVSE, the RPiEVSE. To better understand the concepts in this thesis, during the internship the author has been implied in a V2L project: a Wallbox update to let the car energize a socket mounted on the charging device itself. Fig.(6.1a) and fig.(6.1b) show the prototype of the Wallbox V2L made by S.&h; this device does not implement the ISO15118 standard but stops at the IEC61851 which means that the only communication available between the charging

station and the vehicle is analogical. Due to the V2L function implemented in this Mode 3 Wallbox, at any time the EV will stop charging and start supply current.



(a) First prototype of V2L Wallbox by S.&h.



(b) Second prototype of V2L Wallbox by S.&h..

Figure 6.1: In (a) is shown the prototype that a customer brought to S.&h. to work on it and make it better; in (b) the intermediate work before the actual final product.

Obviously to make this happen it is necessary that the car is also enabled to bidirectional power transfer: the EV used for testing purpose mounted, indeed, the inverters on board and, as already explained, this makes sure that from the vehicle battery, $77.4kWh$, a $220V$ AC voltage is extracted. Being able to get a voltage in AC, like the one present in the houses, it is possible to redirect this current to an electrical socket mounted on the Wallbox and, through the use of teleprompters, choose whether to power it through the electrical network or the EV battery. It is clear that V2X infrastructures are already a reality and are increasingly gaining ground in the society. The practical application just described shows that the technologies of electric vehicles are already mature, now it is time to work on the EVSE side. This V2L application is a fairly simple case as it requires a simple electrical circuit and a small software update for the management of new added signals by Pilot Board; when instead the charging column must interact with the network, as in V2G, the process becomes more complicated and within EVSE it is needed to run a software to manage this communication. The work done in this thesis is only a part of a big project that, for time reasons, it has been stopped at this stage; the following list

show the further steps and is to be intended as the Roadmap for future developments:

1. Once the stable communication between RPiEV and RPiEVSE has been established, the right version of ModRISE has been deployed on the right hardware and the correct functioning of the BPT has been tested on both devices, it will time for porting from Java to C. The C version of ModRISE will be deployed on a S.&h. pilot board and tested throughout the design phase thanks to the RPiEV. During the CurrentDemand messages the real values will be registered and all variables created for simulation purpose will be connected to real signals; in the ChargeParameter state will be set all current and voltage maximum limits according to available devices and the BPT states will be removed and bypassed to save time. Once the operation of the modified board has been verified, S.&h. will have a BPT-enabled charging column.
2. At this point the RPiEV will be mounted in the EVSE tester S.&h., the electric vehicle simulator already present in the company. From now on, a new EV simulator and a new pilot board will be available.
3. Once the physical architecture has been built and the BPT management software has been imported, the product will be improved: an **online** HMI will be created to manage the real signals coming from the vehicle through web page or App which must be able to send commands to the vehicle.
4. At this point, S.&h. will dispose of an EVSE with new BPT-enabled software and an EV simulator with an advanced PC-manageable GUI, also BPT-enabled. It remains to manage the backend service of the EVSE through which the individual column receives information from the UVAM and, consequently, demands for energy. To do this one of the possible solutions is to handle internet requests through a server which responses to REST API.

Bibliography

- [1] Fast reserve - information pack. Technical report, Terna S.p.A.
- [2] Enabler for intelligent charging insys powerline gp. Technical report, INSYS MICROELECTRONICS GmbH.
- [3] A. Amin, W. U. K. Tareen, M. Usman, H. Ali, I. Bari, B. Horan, S. Mekhilef, M. Asif, S. Ahmed, and A. Mahmood. A review of optimal charging strategy for electric vehicles under dynamic pricing schemes in the distribution charging network. *Sustainability*, 12(23):10160, 2020.
- [4] M. S. Carmeli, M. Rossi, and M. Marco. *Macchine elettriche Modelli a regime: teoria ed esercizi*. Società Editrice Esculapio, 2020.
- [5] Coabt. I 150 anni di storia dell'auto elettrica. URL <https://www.cobat.it/comunicazione/rivista-ottantadue/articolo/i-150-anni-di-storia-dellauto-elettrica#:~:text=Il%20primo%20veicolo%20a%20trazione%20elettrica%20venne%20presentato%20a%20Parigi,prototipo%20di%20city%20car%20elettrica>.
- [6] A. a. c. Dr. Steve Arar. Charging modes for electric vehicles. URL <https://www.allaboutcircuits.com/technical-articles/four-ev-charging-modes-iec61851-standard/>.
- [7] E.Dso. Why smart grids. URL <https://www.edsoforsmartgrids.eu/home/why-smart-grids/>.
- [8] Enel Green Power. Hydroelectric plants. URL <https://www.enelgreenpower.com/learning-hub/renewable-energies/hydroelectric-energy/hydroelectric-plants>.
- [9] ENTSOE. Picasso. URL https://www.entsoe.eu/network_codes/eb/picasso/.
- [10] Gloquadtech. Ev charging - evcc for ev. URL http://www.gloquadtech.com/gloquad/bbs/board.php?bo_table=EVCharging&wr_id=8.

- [11] T. M. House. All relevant charging cable and plug types. URL https://www.mobilityhouse.com/int_en/knowledge-center/charging-cable-and-plug-types#:~:text=To%20charge%20your%20electric%20car,e.g.%20for%20the%20Nissan%20Leaf.
- [12] ISO/IEC. Iso/iec dis 15118-2: Road vehicles - vehicle to grid communication interface – part 2: Network and application protocol requirements, 2012. URL http://www.iso.org/iso/iso_catalogue/catalogue_ics/catalogue_detail_ics.htm?ics1=43&ics2=120&ics3=&csnumber=55366.
- [13] D. Lorenzo Ferrari. Charging modes for electric vehicles, . URL <https://www.dazetechnology.com/charging-modes-for-ev/>.
- [14] D. Lorenzo Ferrari. How long does it take to charge an electric vehicle?, . URL <https://www.dazetechnology.com/how-long-does-it-take-to-charge-an-electric-vehicle/>.
- [15] W. Luca Scialò. Elon musk: storia del fondatore di tesla motors che ispirò iron man. URL https://www.webeconomia.it/elon-musk/#Tesla_Inc.
- [16] S. Martinenas, C. Træholt, P. Andersen, and M. Marinelli. Enabling technologies for smart grid integration and interoperability of electric vehicles. *Technical University of Denmark, Department of Electrical Engineering*, 2017.
- [17] M. Mültin. Iso 15118 as the enabler of vehicle-to-grid applications. In *2018 International Conference of Electrical and Electronic Technologies for Automotive*, pages 1–6. IEEE, 2018.
- [18] Next. Mercato per il servizio di dispacciamento per tutti: Unità virtuali abilitate miste (uvm). URL <https://www.centrali-next.it/hub-della-conoscenza/>.
- [19] NRMA. What are the different types of electric vehicles? URL <https://www.mynrma.com.au/cars-and-driving/electric-vehicles/buying/types-of-evs>.
- [20] A. Observations, L. Des, et al. Road vehicles–vehicle-to-grid communication interface–part 2: network and application protocol requirements, 2014.
- [21] J.-F. REY. Safety measures for electric vehicle charging. Technical report, University of Zurich, Department of Informatics, 01 .
- [22] M. SCARFOGLIERO. Electrical, thermal and aging model of li-ion batteries for vehicle-to-grid services. 2018.
- [23] Schneider Electric. Smart charging perspectives for optimal ev inte-

gration. URL https://www.electrical-installation.org/enwiki/Smart_charging_perspectives_for_optimal_EV_integration.

- [24] Terna S.p.A. Driving Energy. How the electricity system works. URL <https://www.terna.it/en/electric-system/terna-role/how-electricity-system-works>.
- [25] Wikipedia contributors. Chademo — Wikipedia, the free encyclopedia, 2022. URL <https://en.wikipedia.org/w/index.php?title=CHAdEMO&oldid=1076634150>. [Online; accessed 10-April-2022].

A | Appendix A

In this section there will be found material about codes and state machines.

```

60 public class UI extends JFrame implements UIInterface{
61
62     private static final long serialVersionUID = 1L;
63     public boolean changestate=false,stopflg=false;
64     public int batteryflg;
65     private static volatile UI istanza;
66     private JPanel contentPane,panel_1;
67     public JCheckBox checkbox1;
68     private JTextPane txtpn;
69     public V2GCommunicationSessionHandlerEVCC connection;
70     public JRadioButton rdbtnManual,rdbtnAuto;
71     public JButton btnStop, btnStart, btnNext;
72     public JProgressBar progressBar;
73     public JLabel lbl;
74     private JTextField timeToLeaveText;
75     public int timetoleave;
76
77     private UI() {
78
79         this.setVisible(true);
80         this.setResizable(false);
81         this.setTitle("ISO15118 - Graphical EV interface");
82         this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
83         this.setBounds(100, 100, 730, 481);
84
85         JMenuBar menuBar = new JMenuBar();
86         setJMenuBar(menuBar);
87
88         JMenu mnNewMenu = new JMenu("Session");
89         menuBar.add(mnNewMenu);
90
91         JMenuItem mntmNewMenuItem = new JMenuItem("Settings");
92         mntmNewMenuItem.addActionListener(new ActionListener() {
93             public void actionPerformed(ActionEvent e) {
94                 File file = new File("EVCCConfig.properties");

```

(a) UI Class definition

```

282 public static UI getUi() {
283     if (istanza == null) {
284         synchronized (UI.class) {
285             if (istanza==null)
286                 istanza= new UI();
287         }
288     }
289     return istanza;
290 }

```

(b) Singleton declaration

Figure A.1: UI is a private class designed as a "Singleton": thanks to this declaration it is possible to create one single object and refer to it through the function `UI.getUi()`

```

349 @Override
350 public boolean isChargingLoopActive() {
351     // Keep charging until 100 charging loops are finished
352     if (getChargingLoopCounter() < 100) {
353         if (getChargingLoopCounter() < 10) {
354             setChargingLoopCounter(getChargingLoopCounter() + 1);
355         }
356         /*
357          * OPTIONAL:
358          * Trigger a renegotiation after 50 charging loops (for testing purposes); you can comment this out if you do not want to test an E
359          */
360         if (getChargingLoopCounter() == 50) {
361             getCommSessionContext().setRenegotiationRequested(true);
362             getLogger().debug("EV triggered a renegotiation (for testing purposes)");
363         }
364         if (getCommSessionContext().getEvController().isBptRequired() && !getCommSessionContext().getEvController().getBptEnable()) {
365             getCommSessionContext().setRenegotiationRequested(true);
366             getLogger().debug("EV triggered a renegotiation to enable bpt");
367             setChargingLoopCounter((short) 0);
368         }
369         else if (getCommSessionContext().getEvController().getBptEnable() && !getCommSessionContext().getEvController().isBptRequired()) {
370             getCommSessionContext().setRenegotiationRequested(true);
371             getCommSessionContext().getEvController().setBptEnable(false);
372             getLogger().debug("EV triggered a renegotiation to disable bpt");
373             setChargingLoopCounter((short) 0);
374         }
375     }
376     return true;
377 } else
378

```

Figure A.2: ChargingLoop: this is an example of how the code as been modified; the added part is the one below the grey comment.

```

ConnectionHandler: Length of V2GTP payload in bytes according to V2GTP header: 36
ConnectionHandler: Message received
EXIficientCodec: Received EXI stream: 8000EBAB9371D3489B79D189A98989C1D191D19181899
WaitForSupportedAppProtocolReq: SupportedAppProtocolReq received
WaitForSupportedAppProtocolReq: Preparing to send SupportedAppProtocolRes
ConnectionHandler: Message sent
V2GCommunicationSessionSECC: New state is WaitForSessionSetupReq
ConnectionHandler: Length of V2GTP payload in bytes according to V2GTP header: 14
ConnectionHandler: Message received
EXIficientCodec: Received EXI stream: 8098004011D018513E2BCA354800
WaitForSessionSetupReq: SessionSetupReq received
WaitForSessionSetupReq: Preparing to send SessionSetupRes
ConnectionHandler: Message sent
V2GCommunicationSessionSECC: New state is WaitForServiceDiscoveryReq
ConnectionHandler: Length of V2GTP payload in bytes according to V2GTP header: 13
ConnectionHandler: Message received
EXIficientCodec: Received EXI stream: 80980223CF61CCBB8DEC62D1B8
WaitForServiceDiscoveryReq: ServiceDiscoveryReq received
WaitForServiceDiscoveryReq: Preparing to send ServiceDiscoveryRes

```

(a) Handshake in RISEV2G

```

+++ Start application handshake protocol example +++
EV side: setup data for the supported application handshake request message
EV side: send message to the EVSE
EVSE side: List of application handshake protocols of the EV
    Protocol entry #=1
        ProtocolNamespace=urn:iso:15118:2:2010:MsgDef
        Version=1.0
        SchemaID=1
        Priority=1
    Protocol entry #=2
        ProtocolNamespace=urn:din:70121:2012:MsgDef
        Version=1.0
        SchemaID=2
        Priority=2
EV side: Response of the EVSE
    ResponseCode=OK_SuccessfulNegotiation
    SchemaID=1
+++ Terminate application handshake protocol example with errn = 0 +++
+++ Start V2G client / service example for charging (ISO1) +++
EV side: call EVSE sessionSetupEVSE side: sessionSetup called
Received data:
Header SessionID=0 0 0 0 0 0 0
EVCCID=10
EV side: received response message from EVSE
Header SessionID=1 2 3 4 5 6 7 8
ResponseCode=0
EVSEID=20
EVSETimeStamp=123456789
EV side: call EVSE ServiceDetail

```

(b) Handshake in OpenV2G

BS EN ISO 15118-2:2016
 ISO 15118-2:2014(E)

Licensed copy: Care Segreteria - MILANO POLITECNICO, CRUI Conferenza dei Rettori delle, Version correct as of 03/08/2021

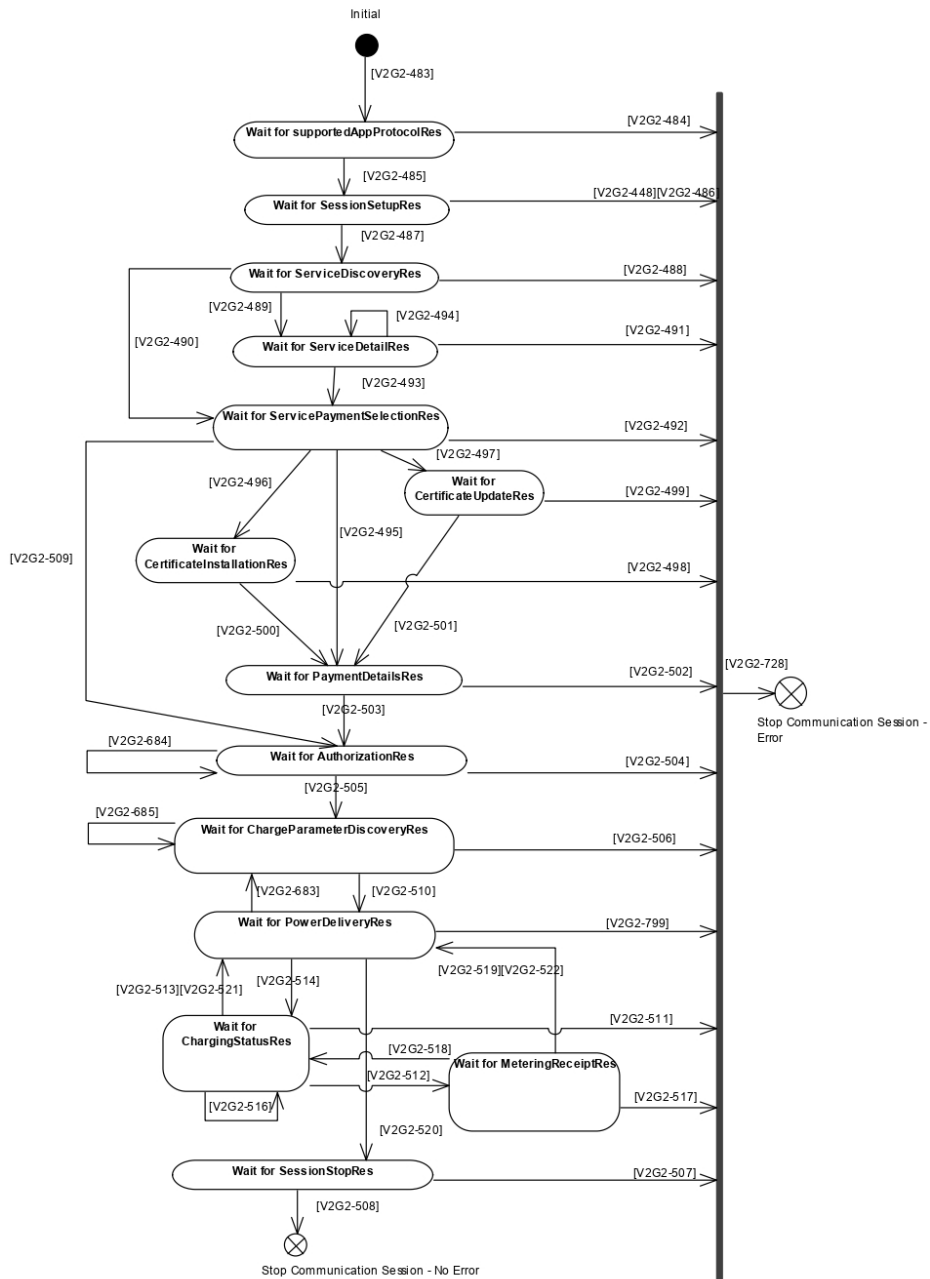


Figure 101 — EVCC Communication states for AC V2G messaging

Figure A.3: EVCC-AC charging

Licensed copy: Care Segreteria - MILANO POLITECNICO, CRUI Conferenza dei Rettori delle, Version correct as of 03/08/2021

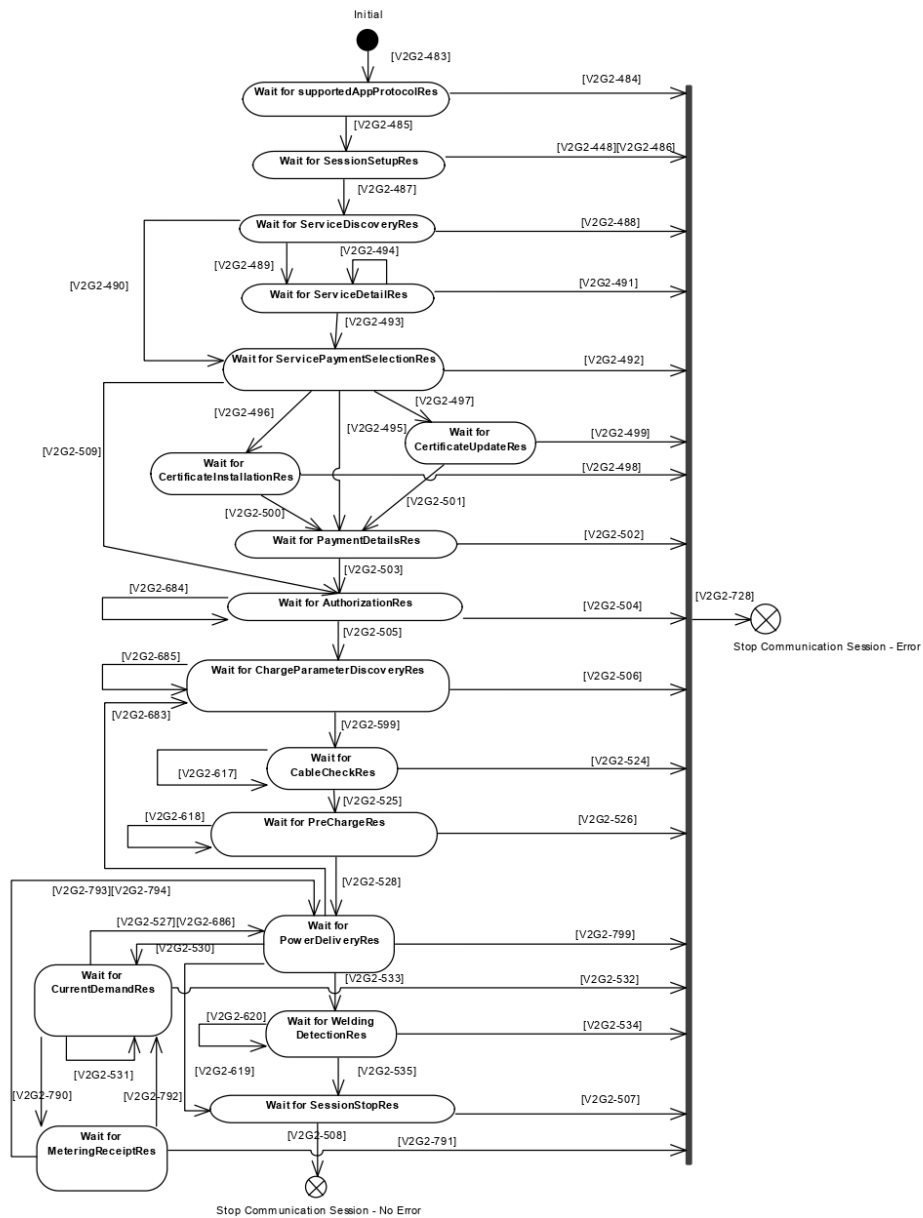


Figure 102 — EVCC Communication states for DC V2G messaging

Figure A.5: EVCC-DC charging

Licensed copy: Care Segreteria - MILANO POLITECNICO, CRUI Conferenza dei Rettori delle, Version correct as of 03/08/2021

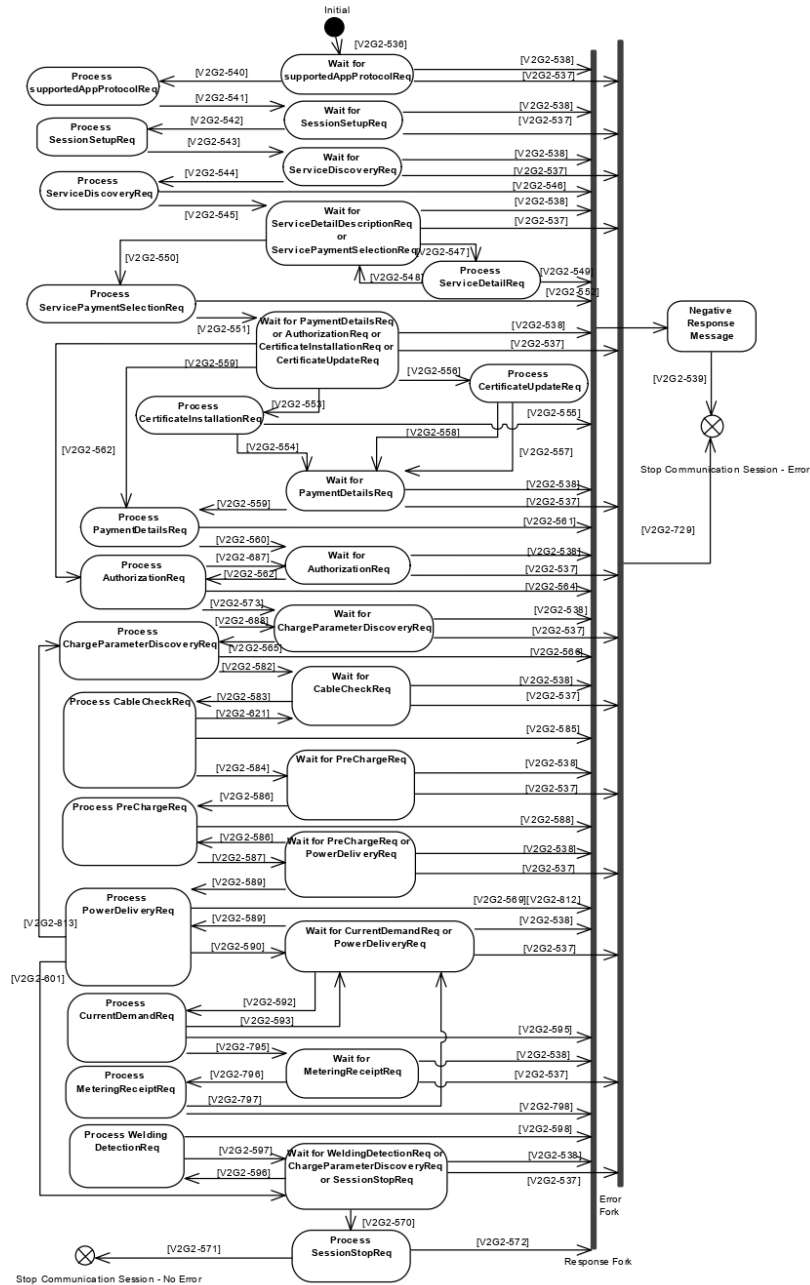


Figure 104 — SECC Communication states for DC V2G messaging

Figure A.6: SECC-DC charging

B | Appendix B

In this section will be found material about hardware in use.

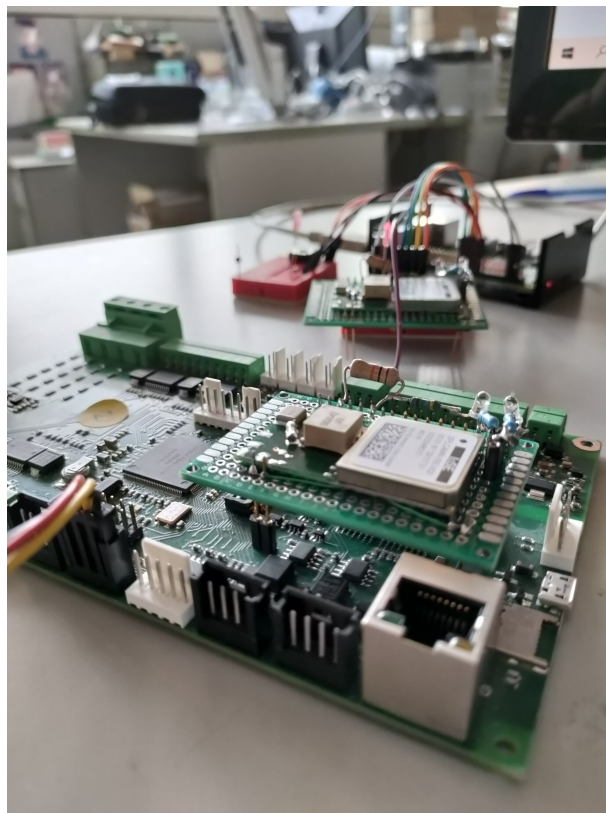


Figure B.1: In the front of the picture there is the S.&h. Pilot Board with the PLC module on top; on the background the RPiEV connected to the board through CP and PP cables.

The PLC module has been removed by the Pilot Board and mounted on a bread board ready to be connected to its RPi. Following pictures show step by step the realization of RPiEV and RPiEVSE.

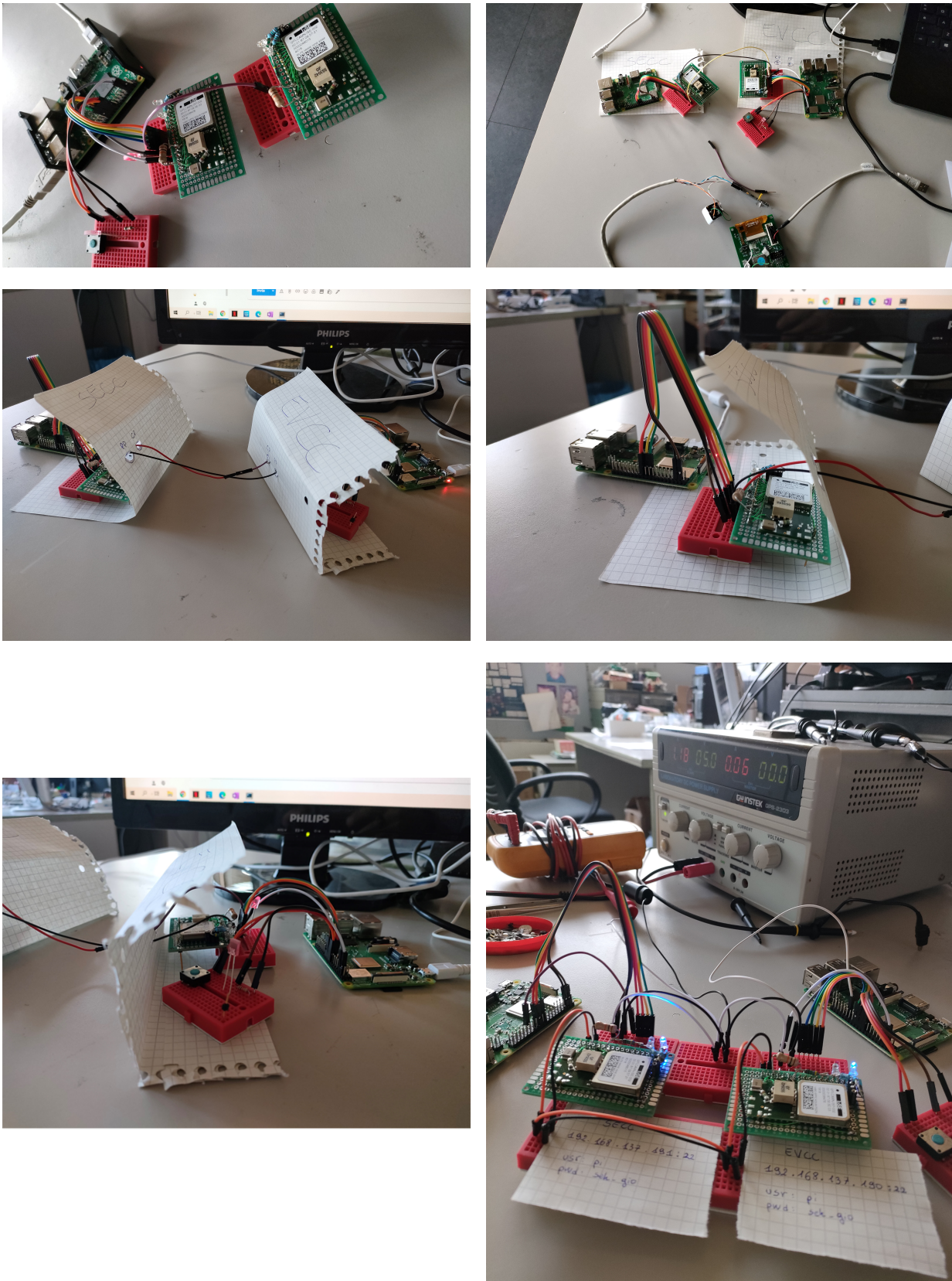


Figure B.2: Stages of the architecture building: each step provide a coding/configuration phase and a connection test.

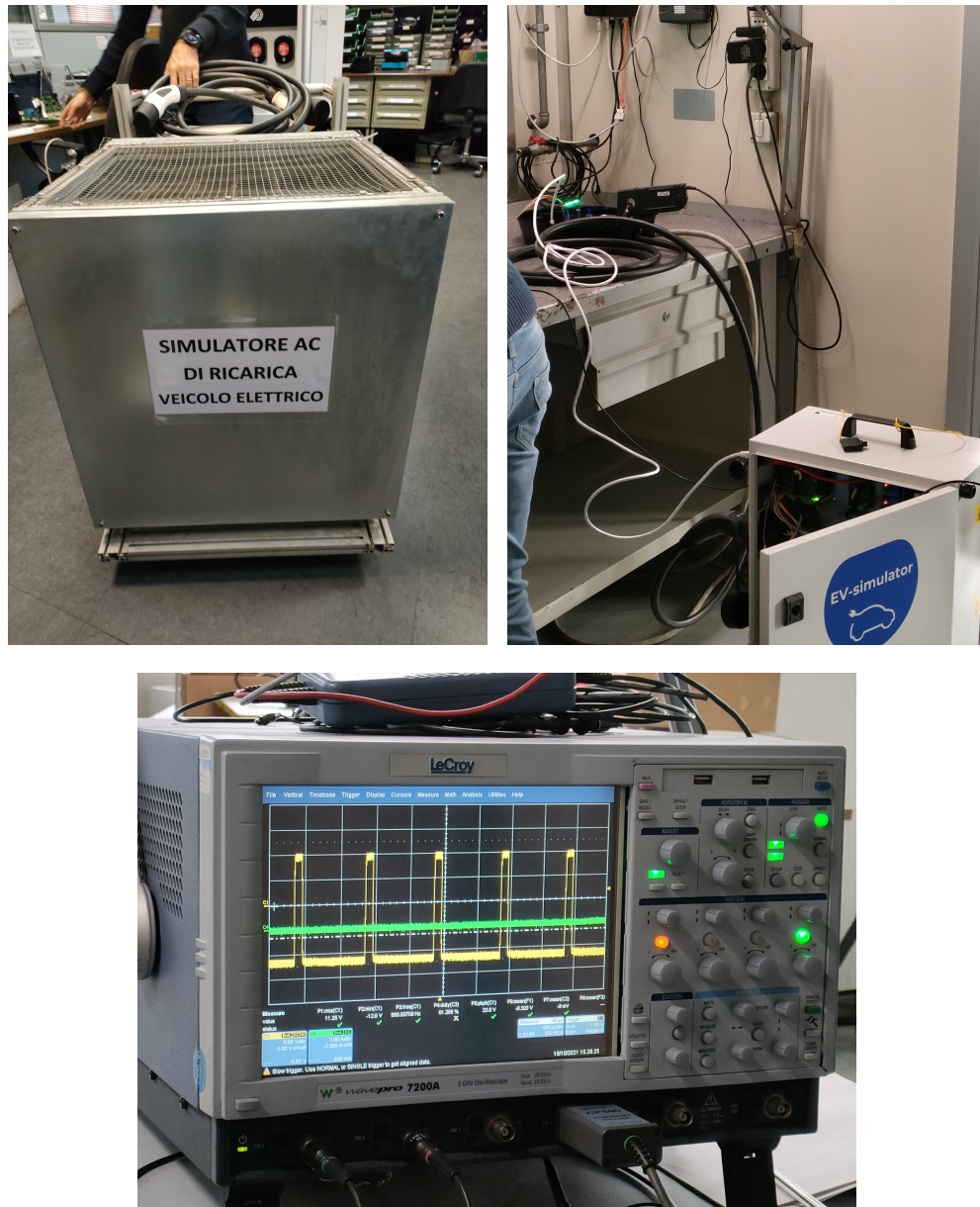
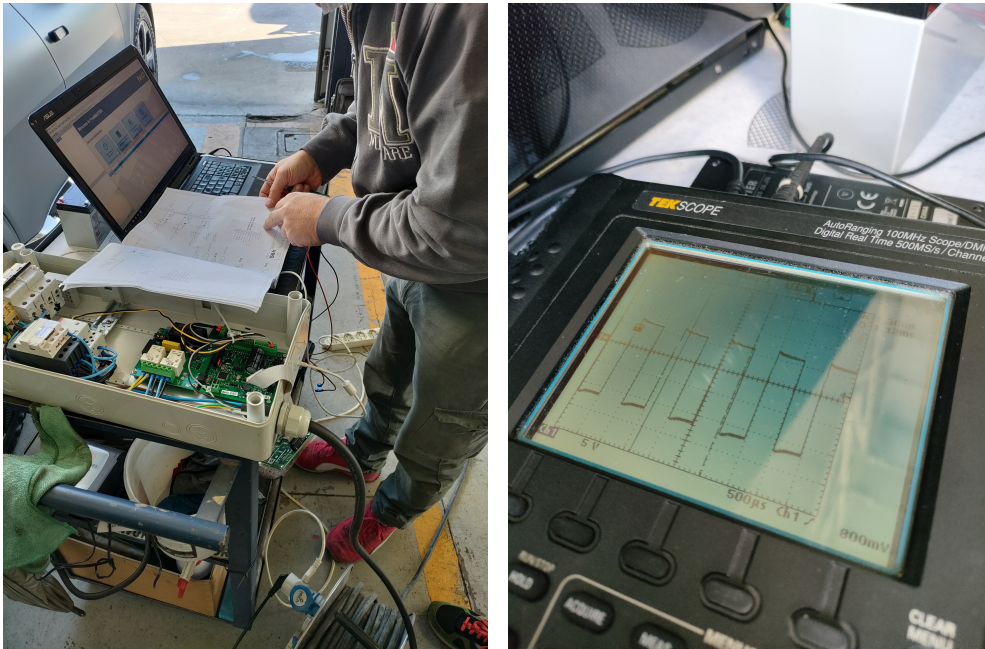
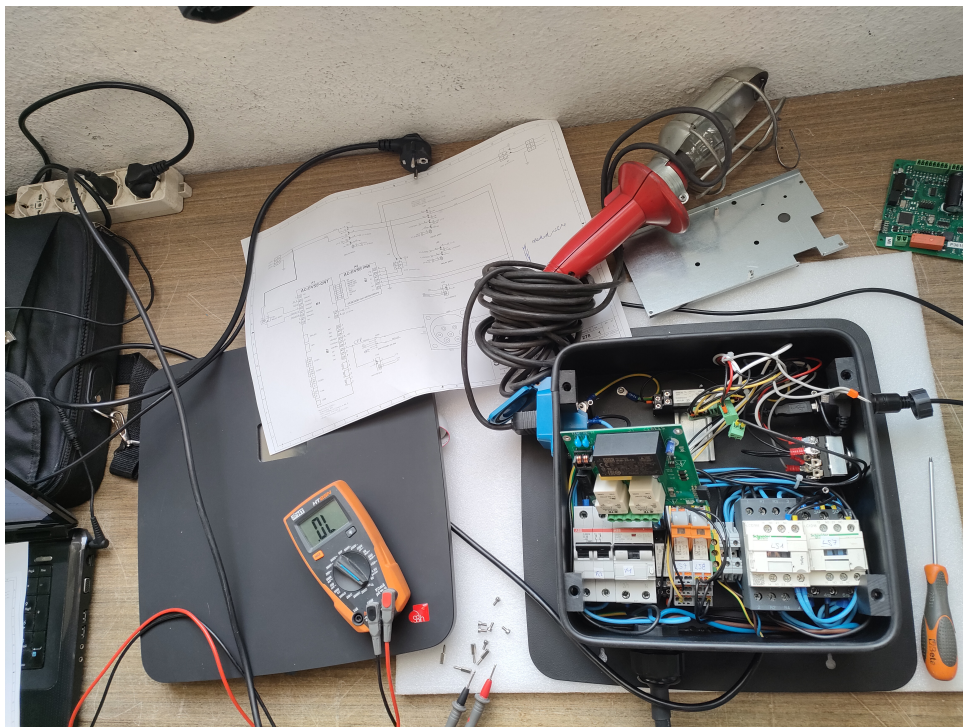


Figure B.3: On the left, an EV simulator equipped with power modules; on the right a small EV simulator without load connected



(a) V2L Wallbox: test on 2nd prototype. (b) V2L Wallbox: portable oscilloscope to test IEC61851 signal.



(c) V2L Wallbox: test on 3rd prototype.

Figure B.4: On the left, an EV simulator equipped with power modules; on the right a small EV simulator without load connected

List of Figures

1.1	Energy production and distribution chain with focus on Terna's position.	5
1.2	Working principle of an AC generator.	8
1.3	Power generator in a hydroelectric plant.	9
1.4	Terna reaction to out of range frequency values.	11
1.5	Frequency snapshot	12
1.6	Comparison between the Fast Reserve and FCR resource.	14
1.7	Charging station with 7 Mode 4 EVSEs.	15
1.8	Italian UVAM owned by Terna.	16
2.1	Comparison between different uncontrolled methods.	18
2.2	Caption of the Figure to appear in the List of Figures.	19
3.1	EV powertrain under charge.	21
3.2	First experimental electric bus.	22
3.3	Battery electrical schema: each parallel RC circuit represents a single cell.	25
3.4	EVCC	27
3.5	Calendar aging	29
3.6	Aging model comparison	30
3.7	Charging modes	32
3.8	Mode 3 Mode 4 comparison	33
3.9	Time computation to fully recharge EVs depending on battery capacity and EVSE power [14]	34
3.10	Caption of the Figure to appear in the List of Figures.	35
3.11	Socket type: 1, 2 and CCS	36
3.12	CCS and CHAdeMO pinout	37
3.13	Time-current effect of AC current on human body (IEC 60479-1) – curve c1 defines the threshold of ventricular fibrillation.	40
3.14	RCD of type B complying to IEC 62423.	41
3.15	Recommended single line diagram for charging electric vehicle.	42
4.1	Control pilot circuit in Mode 3 and 4.	45

4.2	PWM signal representation in IEC61851.	47
4.3	Network layers in ISO15118-2.	48
4.4	EVCC states machine for DC charging.	49
4.5	Message schema definition: AppProtocolType for supportedAppProtocol.	50
4.6	V2G message example	50
5.1	Enabler for Intelligent Charging INSYS Powerline GP.	56
5.2	INSYS Powerline GP working scheme.	56
5.3	RaspberryPi 3B+.	57
5.4	PLC module based on HomePlug Green PHY.	58
5.5	Pinout of: a) PLC Module, on the left, b) RPi, on the right.	59
5.6	SPI mode: for each configuration data are read on the raising or falling edge and clock signal starts high or low.	59
5.7	RPiImager: allows to write OSs on a SD card both suggested by the software or downloaded by internet.	60
5.8	The oscilloscope shows the result of the communication before the changes.	62
5.9	Communication tests	63
5.10	Software logos	64
5.11	Network layer in V2G infrastructure.	64
5.12	RISE project folder	65
5.13	EVCC configuration file	66
5.14	EVCC and SECC packages	66
5.15	Terminal view of EVSE waiting for EV	67
5.16	Terminal view of EV trying to connect the EVSE	67
5.17	Terminal view of an OpenV2G session.	68
5.18	Physical HMI	69
5.19	WindowsBuilder interface.	70
5.20	Software HMI.	71
5.21	Modified state machine	73
5.22	BPT messages definition	74
5.23	isBptRequired	75
5.24	Function to define hysteresis cycle.	76
5.25	Function to set upgraded value of current.	77
5.26	Modified information inside CurrentDemand messages	78
6.1	V2L prototypes	80

A.1	UI is a private class designed as a "Singleton": thanks to this declaration it is possible to create one single object and refer to it through the function UI.getUi()	87
A.2	ChargingLoop: this is an example of how the code as been modified; the added part is the one below the grey comment.	88
A.3	EVCC-AC charging	89
A.4	SECC-AC charging	90
A.5	EVCC-DC charging	91
A.6	SECC-DC charging	92
B.1	In the front of the picture there is the S.&h. Pilot Board with the PLC module on top; on the background the RPiEV connected to the board through CP and PP cables.	93
B.2	Stages of the architecture building: each step provide a coding/configuration phase and a connection test.	94
B.3	On the left, an EV simulator equipped with power modules; on the right a small EV simulator without load connected	95
B.4	On the left, an EV simulator equipped with power modules; on the right a small EV simulator without load connected	96

List of Symbols

Abbreviated terms	Description
<i>BEV</i>	Battery Electric Vehicle
<i>CPO</i>	Charging Point Operator
<i>DER</i>	Distributed Energy Resources
<i>DRES</i>	Distributed Renewable Energy Sources
<i>EV</i>	Electric Vehicle
<i>EVCC</i>	Electric Vehicle Communication Controller
<i>EVSE</i>	Electric Vehicle Supply Equipment
<i>EXI</i>	Efficient XML Interchange
<i>PHEV</i>	Plug-in Hybrid Electric Vehicle
<i>PKI</i>	Public Key Infrastructure
<i>PLC</i>	Power Line Carrier
<i>PnC</i>	Plug and Charge
<i>RES</i>	Renewable Energy Sources
<i>SDP</i>	SECC Discovery Protocol
<i>SECC</i>	Supply Equipment Communication Controller
<i>SOC</i>	State of Charge
<i>TCP</i>	Transmission Control Protocol
<i>ToU</i>	Time of Use
<i>V2G</i>	Vehicle-to-Grid
<i>V2GCI</i>	Vehicle-to-Grid Communication interface
<i>V2GTP</i>	V2G Transfer Protocol
<i>UDP</i>	User Datagram Protocol
<i>XML</i>	Extensible Markup Language

Acknowledgements

The author would like to thank the following individuals for their contributions. First, thanks to Eng. Gabriele Marchegiani, the person who presented me this opportunity, introducing me to this thesis and to the company who hosted my internship, but also thanks to Prof. Marco Mauri who believed in me and chose me to be involved in this journey. Special thanks go to Eng. Alberto Crivellaro, the head of S&h S.r.l., for accepting me as an intern and for following me all along the way, despite his countless commitments, reassuring me and motivating me in moments of stalemate. Thanks to the whole company, no exceptions, for made me feel part of a family, with special regards for Marco Aversa, the internal tutor who constantly followed the progress of work during these months, but also to the other colleagues of the R&D division for their technical support and teachings: Paolo Baita, Gianluca Barbaro, Fausto Dalla Vecchia and Filippo Marconcini; all of them have been a great source of both technical and moral teachings, guiding and including me from day one. In the list of those who have made their contribution in recent months must be mentioned and strongly thanked Eng. Attilio Borri who, with his coding skills and his great altruism, spent hours to teach me how to untangle myself in the complicated interpretation and modification of RISEV2G. Thanks to all my friend who have been with me during all these busy years, but the most heartfelt thanks goes to my family and my parents who gave me the chance to get here and that with their sacrifices have given me a bright future.

