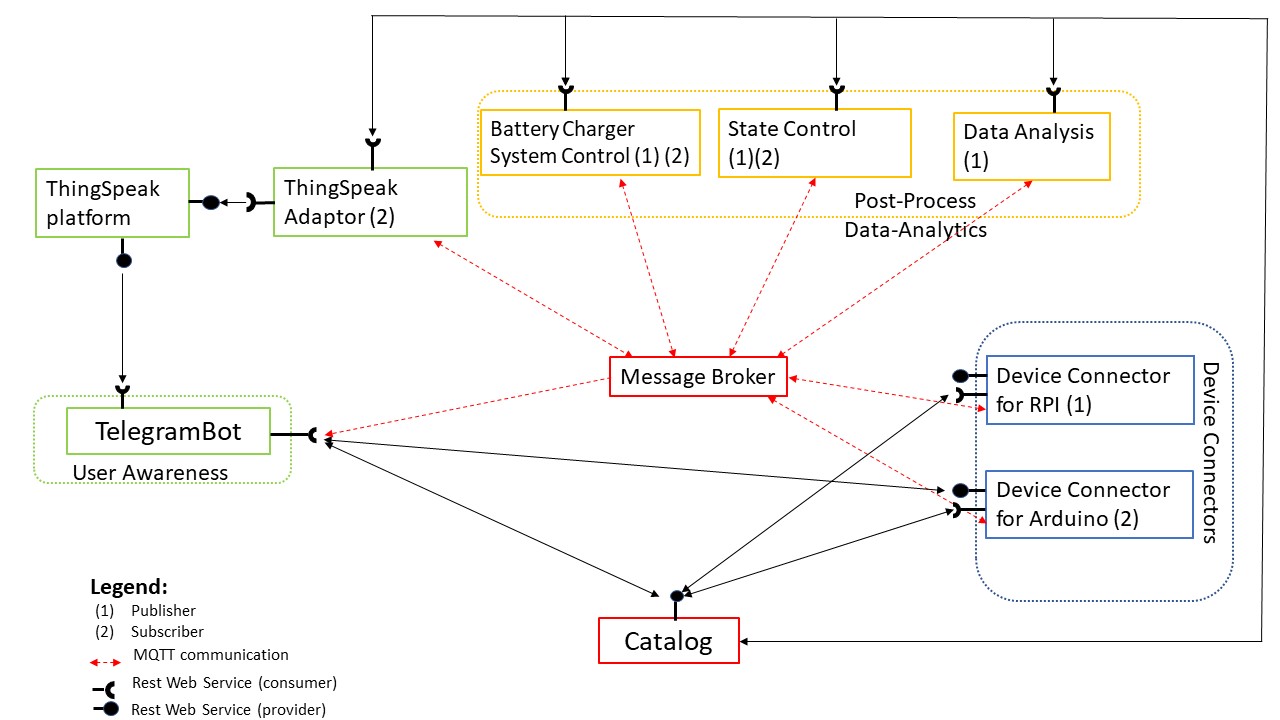
1. **Name of Use Case**

| **Name of the Use Case** | **Smart Car Battery Charging** |
| --- | --- |
| **Version No.** | 01 |
| **Submission Date** | 10/12/2022 |
| **Team Members (with student ids)** | Anna Geraci s296018  Fabio Gianino s301556  Carlo Simone s297278  Michele Claudio Petrocelli s305945 |

1. **Scope and Objectives of Function**

| **Scope and Objectives of Use Case** | |
| --- | --- |
| **Scope** | The IoT platform has the scope to providing services for a smart car battery charging. |
| **Objective(s)** | Electric vehicles are going to dominate the transport sector in the near future, but at the same time, currently, the power used to charge them is mostly produced by burning fossil fuels. This will lead to an unsustainable scenario if actions are not taken to smoothen the transition towards electric vehicles. Under the outlined point of view, this report presents a solution that has the potential to mitigate the inconveniences related to a massive diffusion of electric vehicles, that are overloading the grid power demand, long charging times, etc.. |
| **Domain(s)** | Smart Automotive, Smart grid, Smart Building |
| **Stakeholder(s)** | Car manufacturers, Citizens, Automotive companies |
| **Short description** | The solution consists of both software and hardware parts. From a software point of view, the report explains the kind of data that will be collected in order to implement the choice about the charging time and duration of our car. The overall platform provides unified interfaces (through both REST and MQTT) to integrate the car into Smart Building and Smart Grid environments. It provides control strategies for managing car battery recharge to minimise the energy waste, through light sensor and temperature sensors. The platform allows end-users to give information about their agenda, to better schedule the charging process. Finally, provide an alert message when it is necessary.  Summarising, the main features it offers are:  - remote control of appliances.  - control strategies to minimise energy cost.  - end-user application for energy-awareness  - end-user application for battery autonomy.  - unified interfaces (i.e. REST Web Services and MQTT queues) available to enable Demand/Response.. |

1. **Diagram of Use Case**



1. **Complete description of the system**

The **Message Broker** provides an asynchronous communication based on the publish/subscribe approach, it exploits the MQTT protocol.

**Battery Charger System Control**:Control strategies to manage optimal charging:

- Keep track about temperature outside the house.

- Verifies the presence of the sun and consequently activation of solar panels.

- Keep track about users' daily commitments in order to plan the charger.

- Manages the battery level.

It works as a MQTT subscriber to receive data from sensors and process them, and as a MQTT publisher to send actuation commands to IoT Devices. It works depending on time-schedules provided by the Catalog.

**State Control:**Monitoring the car battery and detect important information

- Analyze presence sensor, charger state, battery level

- Notify an alert in case of detectable issues to the users.

It works as a MQTT subscriber to receive data from sensors and process them, and as a MQTT publisher to TelegramBot for communicating alerts.

**Data Analysis:** Process environment measures to get periodic statistics. Information about Energy consumed by your car. It receives information about Catalog through Rest protocol. It retrieves data from sensors through the MQTT protocol acting as subscriber. Then it provides statistics to send to TelegramBot through ThingSpeak Adaptor thanks to MQTT protocol acting as publisher.

The **Thingspeak Adaptor** is an MQTT subscriber that receives environmental measurements and uploads them on **Thingspeak** through REST Web Services.

**Thingspeak** is a platform used like a database in which the data coming from different sensors, provided by the ThingSpeak Adaptor that uses an MQTT communication protocol, are stored and processed. It uses Rest API to retrieve information by ThingSpeakAdaptor and to send information to TelegramBot and.

The **Catalog** works as both service and device registry system and all the actors in the system are connected to them with a Rest protocol. It gives information about device connectors, users data, services and resources in the platform.

The **Raspberry Pi Connector** is a Device Connector that integrates into Raspberry Pi platform. Temperature sensors, light sensor, presence sensor, battery charge detector are managed by Raspberry. It works as MQTT Publisher to send information retrieved by sensor on the environment temperature, battery temperature, level of insolation, presence of the car in the charge station and percentage of available battery. It provides Rest WebService to communicate with the Catalog.

The **Arduino Pi Connector** is a Device Connector that integrates into the platform Arduino boards. In each Arduino there is a Relay to switch ON/OFF the connected appliance. In this case is a switch to enable and disable current flow through the car battery charger. It works as MQTT Subscriber to receive actuation command by other actors that exploit the MQTT protocol. It provides Rest WebService to communicate with the catalog, so to change and retrieve information about actuator status.

**Telegram Bot** is a service to integrate the proposed infrastructure into Telegram platform, which is a cloud-based instant messaging infrastructure. It works as a MQTT subscriber and communicates with State control that sends alert messages. It also allows users to send actuation commands to IoT devices and to receive graphs, statistics and data from both ThingSpeak and Catalog exploiting REST.

1. **Desired Hardware components (only among those we can provide)**

| **Device Name** | **Quantity** | **Needed for…** |
| --- | --- | --- |
| Temperature Sensor | 2 | To detect both battery and environmental temperature. |
| Digital Push Button (Presence Sensor) | 1 | To check the presence of the car. ( In alternative to the presence sensor, the application could be adapted with the digital push button) |
|  |  |  |
|  |  |  |
|  |  |  |