Domotic Platform for Energy Management

André Eusébio
andreusebio@gmail.com
Instituto Superior Técnico, Universidade de Lisboa

Abstract—Electricity is one of the commodities where demand is expected to continue to increase in the near future. To avoid the growth of pollution caused by the use of fossil fuels for energy production, society should not only aim for a wider usage of renewable energy sources but also for a more responsible and optimized consumption and management of electricity. By the use of Home Energy Management Systems (HEMS), consumers will be able to improve their energy usage and thus, reduce their electricity bill and carbon footprint.

This thesis analyzes the concepts of smart grid, HEMS and Internet of Things, and proposes a low-cost HEMS that gives users the ability to monitor their consumption and remotely control their appliances. In addition, a real-time load control algorithm was developed and tested, which implements automatic control over appliances, to reduce house peak consumption and the electricity bill, especially on households with a local micro-production energy source.

The proposed HEMS uses an architecture supported on Wi-Fi and MQTT protocols for communication, ESP8266 modules for measuring energy consumption and integrates a platform and user interface created with the Node-RED tool. A prototype of the HEMS was assembled and tested, fulfilling the established goals.

I. INTRODUCTION

Electricity is one of the commodities that is present in most of the households. Considering that each day more houses get access to this important resource and that demand is expected to continue to increase in the near future, society should guarantee that the increase in energy production won’t bring pollution to an even more dangerous level. According to the International Energy Agency, in 2015 more than 80% of the energy produced in the world came from non-renewable natural resources [1] which are known for producing CO2 and other pollutant gases. In order to try to reduce this problem, countries should aim to shift their energy production as much as possible to renewable energy sources. Relying on renewable energy sources presents many challenges, like the fact that production can’t follow demand and renewable sources like wind can be unreliable.

Achieving sustainable electricity production won’t be enough to surpass this challenge, so consumers should also optimize their energy consumption. One way to achieve this is developing smart grid and microgrid systems in order to achieve a more controlled consumption that follows as much as possible the energy available by the renewable sources in order to minimize the use of fossil fuels.

Trying to achieve a more responsible energy consumption is not a straightforward task for the average consumer. Most consumers only know their energy consumption at the end of each month after receiving their energy bill. They may read their supplier power meter but that only gives them the ability to know their accumulated energy consumption till that moment and not the instantaneous power consumption of their home.

For consumers to be able to know and improve their energy management, they need tools that help them analyze and regulate their consumption habits. This means to know their instantaneous power demand and also a history of their power consumption throughout the day. In the case of consumers with micro-production, having an autonomous system that manages the house consumption might also be an important tool that gives them the ability to use their produced energy with more efficiency and with a smaller monthly electricity bill.

In order to address these needs, several home energy management systems (HEMS) have been developed, some in the form of products available on the market, other as systems developed or proposed on studies.

Currently available products mostly consist in power meters or domotic modules, that give users energy consumption information regarding house and/or individual appliances, and in some cases, offer the possibility to manually schedule loads. These products are usually expensive and closed systems that offer little adaptability to consumer preferences and rarely offer any type of automation. On other hand, studies performed regarding HEMS, simulate complex systems that are able to, using multiple electronic plugs, control appliances individually and schedule actions based on artificial intelligence or using real time algorithms. A common issue with these systems is that they are not tested as physical HEMS and in real scenarios, this way lacking validation.

This thesis targets some of the problems identified in current HEMS systems, proposing an open and expandable infrastructure, based on available low-cost modules, that allows local and remote monitoring of appliances and their manual control. The proposed solution offers also a real-time algorithm to autonomously control house appliances.
II. LITERATURE REVIEW

A. Smart grid

The grid of the future should ensure that the flow of electricity supply and demand should work bi-directional way [2]. In order to have a functional bi-directional flow of energy, the grid of the future must be smart; This means that data should be exchanged between the producers and the consumers with the help of multiple sensors and computers that are able to communicate and act automatically [3].

The smart grid may be divided into three distinct parts [4]:

• Main power generation – It consists on electricity producing infrastructure. The smart grid uniformly moderates the power generation over the day with the involvement of the consumers who communicate their daily demand and production.

• Distribution system – This part regulates the electricity voltage based on the final electricity destination, which can either be residential, industrial or commercial.

• Consumers – Consumers use, whenever available, energy produced through their own microgrids and send the overproduction to the grid. Also with the help of HEMS they manage their energy consumption and communicate their energy production and consumption.

B. Internet of Things

The “Internet of Things” commonly known as IoT is one of the main concepts that powers communications on the smart grid. It generally refers to the multiple networks of devices or platforms that communicate with each other via wireless protocols and without direct human interaction. Connections made through IoT-enabled devices facilitate the rapid and efficient transfer of data needed to support a wide range of activities and operations. In this way, the application of IoT technologies can lead to significant operational improvements, such as increased efficiency, better performance and enhanced safety [5].

IoT networks rely on different wireless communications protocols to enable the transmission of data between devices. Some of the more widely used communications protocols are [6]:

• Bluetooth and variants: The Bluetooth protocol operates at distances typically up to 60 meters (version 4.0). Bluetooth’s principle advantages are its low power consumption, the capacity to handle multiple devices simultaneously, and its ability to transmit wirelessly without visual line of sight contact [7].

• Zigbee: Designed for use in IoT applications operating in environments that require secure networking and long battery life but that are not data transmission intensive [8]. Zigbee-based IoT devices can communicate via line of sight at distances between 10 and 100 meters and work in mesh networks, which makes it possible to create huge networks since any node can become a coordinator and expand the network [9].

• Wi-Fi: Wi-Fi works within the 2.4-5GHz frequency band to connect wireless devices to a network access point, it has an indoor range of about 20 meters and is able to handle high quantities of data [10]. Although its fast data transfer speeds make it perfect for file transfers and media streaming, Wi-Fi is too power-consuming for many IoT applications that use batteries. One big advantage of Wi-Fi is recent appearance of the low-cost ESP8266, which is an easily programable microcontroller with integrated antenna device [11] with a large community of users.

C. Home Energy Management System

An HEMS, or also known as SHEMS (smart HEMS) is an essential domotic system for the successful demand-side management of smart grids [12]. HEMS monitor and, in some cases, control various home appliances in real time according to user’s preferences in order to reduce electricity bill and improve energy utilization efficiency [13]. Most HEMS have a wide range of functionalities than can be gathered in five main functional modules, that include monitoring, logging, control, management and alarm [14].

Apart from basic HEMS functionalities that only provide users with data regarding energy consumption and simple scheduling, some systems have advanced tools that autonomously may reduce energy bills or provide users with more consumption data using fewer hardware.

As it is explained in [15], over the last decade many researches have been developed about HEMS Most of these researches are divided into two main categories: predictive energy management and consumption and real-time management. While the former category uses prediction models and data to forecast the consumption and available supply in order to find the optimum strategy to control the electrical devices, usually trough load disaggregation algorithms, like in [4], [16], [17] and [18]. The second category uses real-time algorithms to control the thermal devices or shift the controllable devices to get two important aims, reduce the peak to average ratio in load demand and reduce the electricity bills, like in [15] and [19]. Regarding energy disaggregation algorithms, they are able to provide user with specific appliance consumption, without the need of individual power meters. This type of approach can also be found in many recent studies.

D. Load disaggregation

As mentioned above predictive management models rely on load disaggregation algorithms. The way this algorithms work is by dispatching tasks in a certain order throughout the day in order reduce the costs [4]. The scheduling of tasks is done according to multiple factors:

• Variation of electricity prices: Trying to shift consumption to off-peak hours with cheaper prices proves
to be a good way to reduce electricity bills when using not fixed tariffs [16].

- **Local energy production:** Tasks should be done whenever its possible, using energy produced by local sources [16].
- **Storage system:** Managing when to charge the battery or when to use stored energy creates a great opportunity for savings [17].
- **User needs:** Finally, and most importantly algorithms should schedule tasks according to priorities and deadlines set by the user when he defined a specific time interval for the appliances to run [4].

The main issue associated with these load disaggregation algorithms is that they are very hard to implement in a real HEMS. Also, when they are used, they may require a vast change of consumer habits, which are required to have all their appliances use scheduled, and use appliances exactly at scheduled slot in order to save the most.

### IV. Real-time algorithms

These types of algorithms are usually linear and easy to implement. These algorithms aim to reduce energy bill manly using real-time data and shifting/delaying appliances tasks usually not relying on prediction methods. There are different ways that real-time algorithms can be applied. The simplest way is to control peak power usage, defining a target maximum peak power consumption for the house that, if crossed, the algorithm should actuate in order to reduce it. Another way is, using local storage units and real-time pricing, in which the algorithms can choose for the appliance to run on battery if enough charge is available and electricity price is too high [15]. In other studies, the algorithm relies on total available power from PV production and the power limitation to proceed with appliances control [19].

Savings in this type of algorithms are reduced comparing to load disaggregation methods. Nevertheless, these types of algorithms require less computer power and can be easier implemented in a real HEMS, ready to be installed at any house.

### F. Energy disaggregation

Energy disaggregation is done by taking a whole house energy consumption, using a single power meter (aggregate value), and separate it into appliance specific data, with the help of different statistical approaches and algorithms [20]. This type of HEMS advanced functionality, although doesn’t alone reduce the electricity bill or gives user suggestions on how to do so, represents a very interesting advanced functionality that can be very significant if combined with other techniques, since uses a single measuring device to create a very complete house consumption profile.

### G. Power meters and power measurement

The power meter (PM) is the most important tool of the HEMS. As the name suggests power meters measure energy consumption and may be implemented using a wide number of technologies. PMs usually have one or two sensors, one to measure current and other to measure voltage.

The most accurate method to calculate AC power is to use instantaneous power [21]. Instantaneous power can be obtained as the product of instantaneous voltage and instantaneous current. The result of this operation will be valid for any kind of loads, as it works with resistive, inductive, capacitive and even the modern harmonics rich nonlinear DC loads. It can be obtained from:

\[
P_{\text{inst}} = VI
\]

where

\[
V = V_m \sin \omega t
\]

\[
I = I_m \sin (\omega t - \varphi)
\]

Real power, or also called active power can obtained from instantaneous power average:

\[
P = |V| \cdot |I| \cdot \cos \varphi
\]

where

\[
|V| = \frac{V_m}{\sqrt{2}}
\]

\[
|I| = \frac{I_m}{\sqrt{2}}
\]

Apparent power can be found by taking the product of the RMS voltage and total RMS current.

\[
S = V_{\text{RMS}}I_{\text{RMS}}
\]

Finally, power factor can then be calculated by dividing real power by the apparent power.

\[
\text{Power Factor} = \frac{P}{S}
\]

Although using current and voltage readings is the most accurate way to determine apparent power, it can also be calculated using only current readings and using an estimated \(V_{\text{RMS}}\) value in order to determine apparent power. This method can be seen systems like in [4] [22].

### III. System Architecture and Hardware

As it happens in many studies, the development was separated in two areas: one corresponding to the physical system and its assembly, and the other related with the software and algorithms. In this chapter the physical system will be addressed. The following set of requirements was established for the physical system:

- **Measurement:** Be able to measure house consumption and individual appliances consumption as well;
- **Wireless:** The system should be able to communicate in a wireless manner using easy to implement protocol;
- **Expandable:** It should be possible to increase the number of modules on the network;
- **Low cost:** System should have a competitive price when compared to products available on the market;
• **Renewable sources**: Be able to measure energy production from local sources like PV panels and wind turbines;
• **Actuation**: User should be able to turn off/on appliances locally or remotely;
• **Security and privacy**: system data should not be accessible to any unwanted entity;
• **Flexible**: System should be open, able to integrate new modules and changes to the existing modules or architecture.

A. **Architecture**

After settling with Wi-Fi for the wireless communications and considering all requirements, it was decided that the system should have a router hosting the Wi-Fi network and contain the following modules:

• **Main power meter**: module to be installed on the main electrical circuit breaker board to measure full house consumption;
• **Smart plug (Plug)**: interfacing module, that can be installed between the wall plug and an appliance to measure specific consumption data and control the appliance;
• **Server**: module responsible to process all the data, store measurements, host the HEMS platform and user interface.

![Fig. 1. Proposed system architecture](image)

B. **Plugs and main meter microcontrollers**

The Arduino Uno/Nano open source microcontroller board was the first option considered to implement the plugs and main power meter. This was due to its flexibility, low-cost and huge community that facilitates the development for this platform.

What makes these devices very flexible and easy to use is the existence of 14 digital input/output pins and 6 analog inputs (8 in case of NANO), for interfacing with multiple sensors and actuators. They include a USB port, which can be connected directly to the computer without the need of an external programmer (which is necessary on many other microcontrollers). These two Arduino models are powered by an ATmega328P microcontroller from Atmel, which comes with the Arduino bootloader pre-installed and makes it straightforwardly programmable by the Arduino IDE (Arduino Integrated Development Environment) [23].

The second microcontroller considered was the Wi-Fi capable ESP8266 from Espressif Systems. This recently developed microcontroller is sometimes used only as a communication module for Arduino boards. But since this chip comes equipped with its own general-purpose input and output (GPIO) pins and is more powerful in terms of processing than the Arduino it can be used as a standalone device.

The ESP8266 microcontroller comes in development boards in order to make it convenient for use on projects. There are an immense number of different development boards for the ESP8266, one of the most well-known and used types is the NodeMCU family of boards. These boards contain an USB-To-Serial chip, that makes it easily programmable, and a friendly pin interface that it compatible with most breadboards. NodeMCU although originally created with LUA language as first choice, can also be programed using Arduino IDE and C++ after some simple changes to the compiler, and use most Arduino libraries.

Due to the similar cost, superior processing power and having included Wi-Fi capabilities, NodeMCU board was used both on plugs and main power meter, in order to handle measurements and communications.

C. **Power measurement**

In proposed architecture, both plugs and main power meter should be able to measure power consumption. Based on size and price, it was decided that only the main power meter should use both current and voltage sensing while the plugs will only work with current measurement.

1) **Current sensing**: For current measurement two different sensor are used. The one used on the main meter is the SCT-013-000 from YHDC, a split core current transformer that can measure up to 100A RMS and be easily clipped into a main line due to its split core design. For the plugs its used a current sensor module based on the ACS712 from Allegro microsystems, an Hall effect based linear current sensor, that can measure currents up to 30A and has a small size.

2) **Voltage sensing**: Voltage is only measured on the main power meter, and it is done using an AC/AC power adapter, in this case it was used a 9VCA adapter from NIMO ELECTRONIC (ALM081).

3) **Power calculation**: Using the current and voltage sensing methods mentioned above, is possible to obtain apparent power when only current is measured; And when both variables are measured: real power, apparent power and power factor can be obtained. To calculate parameters mentioned we used the emonLib [24] an electricity monitoring Arduino library that uses instantaneous power concept.
D. Actuators

In this architecture, main power meter is only used for gathering data and does not require any actuation capabilities; on other hand the plug modules are required to cut power from connected appliances, so a relay is needed in order to do so. Since the plugs are capable of reading currents up to 30A, a relay also capable of handling that current was chosen.

E. Main power meter design

This module is based on a NodeMCU board connected to an AC/AC voltage adaptor and three CTs. The choice of using three CTs was done in order to make this module more flexible, since in a single-phase scenario, the two spare CTs can be used to measure specific breakers or production from a local source. Also, in the case of a three-phase household, the three CT’s can be used to measure full house consumption (apparent power).

Since the NodeMCU board only has one analogic input port, an external ADC (ADS1015) was required in order to be able to read the four distinct analogic inputs (Figure 2).

F. Plugs design

Plugs have a simpler design that comprises a NodeMCU board, an ACS712 module, 30A relay and finally a button to locally control relay state (Figure 3).

G. Server

The final element of this architecture is the server, which is responsible to aggregate the data from the main power meter and available plugs, store the relevant data, manage the HEMS and host the user interface and other relevant software. A Raspberry Pi 3 Model B was chosen, due to its capabilities, low price and large support derived from its large community.

IV. SYSTEM SOFTWARE AND LOAD MANAGEMENT

After addressing system architecture and hardware designs to fulfill all hardware requirements, the development of the system’s software and the creation of a web platform were necessary in order to implement a fully functional HEMS system.

The main functionalities identified for the software side were:

- **Web interface:** The user should be able to easily access the system, using a web interface locally or remotely;
- **Data visualization:** Data gathered from main power meter and plugs should be easily accessible;
- **Storage:** Acquired data should be stored in a database;
- **Actuation:** The user should be able to remotely control plugs state;
- **Automation:** System should be capable of autonomously control the state of each plug and reduce energy consumption;
- **System status and logging:** The user should be able to verify the status of each module and access system events, which will be logged.

A. Messages protocol

Having Wi-Fi as the wireless networking technology, a compatible light-weight messaging protocol had to be selected, in order to allow data from plugs and main PM reach the server. MQTT was chosen as a consequence of being stable and mature when compared to more recent protocols.

This protocol works on top of the TCP/IP and uses a “publish/subscribe” model. It also requires a central MQTT broker to manage and route messages among an MQTT network’s nodes, as shown in Figure 4.
B. Main power meter and plugs programming

Both plugs and main power meter can be programmed using Arduino IDE and use most Arduino libraries after minor changes. Arduino IDE is an open-source software that can be used to program various microcontrollers using a set of C/C++ functions.

The main power meter and plugs have similar software logic, both of them proceed first with measurements and power calculations using emon.Lib, and then publishing the results to a specific MQTT topic using JSON format, repeating this process every second. In the case of the plugs they are also subscribed to a topic regarding the relay, so that they can receive remote instructions to modify relay state.

C. Server programing

The server responsibilities, which is by far the most complex element of the architecture, are diverse and require the use of multiple software and tools. Its main functions are, to act as a MQTT broker, manage the HEMS, host the HEMS webpage interface, host the database, and act as virtual private network (VPN) server. The advantage of having all these functions hosted on the Raspberry Pi, is that this HEMS will be independent from internet and internet services, which means it will function as long as electric power is available, assuring a bigger protection of the system’s information. Before installing the required software, Raspberry Pi requires an operating system to operate. In the system of this thesis Raspbian, a Debian based Linux distribution, was installed.

1) MQTT broker: When it comes to brokers options for the Raspberry, Mosquito broker is the standard. This MQTT broker developed by Eclipse is lightweight and implements most recent MQTT versions 3.1 and 3.1.1. For these reasons, Mosquito is the MQTT broker used on this project.

2) Platform and user interface: In order to develop the HEMS platform, instead of writing code from scratch, Node-RED was used. Node-RED is an open source, flow-based programming tool built on Node.js, that simplifies the connection of hardware devices, cloud based systems, databases and APIs.

In Node-RED, programing is done on the browser by connecting “Nodes” that send messages containing objects to downstream nodes. A node might have a configurable default function or be fully user-programable, using JavaScript. Similarly, to Arduino IDE that has libraries created by the community, Node-RED offers users the possibility of using custom nodes created and shared by others.

MQTT messages processing: In order to receive messages from plugs and main power meter, Node-RED will act as an MQTT client and subscribe to relevant topics. Information received will then be used for real time visualization, to be stored into the database and to be analyzed by the load algorithm.

Node-RED can also send messages, e.g. when it’s necessary to control the relay state of a plug, a message is published to a specific MQTT topic, containing a binary value of either one or zero, in order to respectively turn it on or off.

User interface: To create the user interface webpage, Node-RED dashboard module was used. This module provides a set of nodes to quickly create a dashboard that can be easily operated by the user. Provided nodes vary from interactable nodes, e.g. a clickable button, dropdown list and text input, to nodes that only display information, e.g. chart, gauge, notification and even a template node that can be configured using HTML (Hypertext Markup Language).

D. Load algorithm

After having a fully operational HEMS, automation was the next functionality to be implemented. In this algorithm, two situations were addressed for applying automatic load control. One situation is when the user wants to set a maximum instantaneous power consumption (max load) with the objective of controlling house energy peak consumption. The other case is to take advantage of local energy sources like PV panels and wind turbines, by setting a production energy level (production threshold), above which he wants his appliances to run as much as possible using only local sources.
The two described cases are addressed by two modes that can be activated by the user. These modes will be further described as “Max load control” and “Solar saving mode”.

1) **Categories**: In this algorithm, appliances management is done based on category attributed to each appliance by the user. Appliances categories are created by the user, and classify appliances in terms of being interruptible or not and how flexible is the interruption.

   Apart from name and ID number, the user also has to set the following parameters that will be used by the algorithm:
   - **Interruptible**: user can decide if the category should be controlled by the algorithm;
   - **Maximum continuous down time**: maximum time the appliance can be continuously turned off;
   - **Minimum down time**: time the appliance should be kept off, if turned off by the algorithm;
   - **Maximum down time**: maximum time appliance should be kept off during selected down time span;
   - **Down time span**: number of hours the algorithm will search back in time for appliance state in order to determine if maximum down time was exceeded.

   The algorithm will turn off more regularly and for longer time appliances that have more flexible parameters, and less often and less time appliances with more restrictive time parameters.

2) **Algorithm description**: The developed real-time algorithm has a circular structure, and its implementation includes checking a set of conditions, in order to find if the appliance/device connected to the plugs should turned on or off.

   In the case of max load control, the algorithm is activated simply with user order. On other hand, in the case of solar saving mode, it requires the user order and also a local production power value that is higher than the threshold. In both modes, after activated, the algorithm will either try to increase or decrease energy consumption, depending on the values of consumption and production or max load. Figure 6 shows algorithm logic.

**E. Database**

The database used by the platform was created using SQLite and managed using phpLiteAdmin. Created database is divided into different tables that apart from storing information received from plugs and main power meter, also contain other important data that should be secure in case of server failure, like the different energy categories created by the user, system logs and devices information.

**F. External access**

A virtual private network (VPN) was created using OpenVPN software in order make it possible to manage all the services remotely.

**V. RESULTS**

In order to validate the proposed HEMS, described in the last two chapters, a testing version was assembled and installed in the household of the author of this thesis. This testing version was composed by prototypes of the distinct modules, including one main power meter module, three smart plug modules, a server and also a solar module.

After the installation of the prototypes, platform was configured in order to test HEMS functionalities.

**G. Installation**

The main power meter module was installed on the circuit breaker panel. Since in this case, the house had a single-phase
electricity contract, only one of the CTs was used to measure full house consumption and the other two CTs were used to measure consumption of distinct breakers.

The three plugs prototypes were enclosed inside appropriate boxes, in order to be handled safely. The plugs were then connected to a computer setup, a fridge and a heater. With these prototypes was possible to create appliances consumption profiles similar to the ones seen in other studies [25].

The server didn’t require complex assembly, since it’s only composed of a Raspberry Pi, memory card and a power supply which are easily connected. For its installation is only required a place with good Wi-Fi signal, so it was installed close to the house router.

Since the used home didn’t possess any type of local energy source, and it was necessary to test the algorithm for that scenario, a solar module was created in order to simulate one. The developed solar module consists of a NodeMCU board connected to a small PV panel that can generate a solar production profile.

A. User interface

The user interface dashboard, with the required functionalities was successfully created using Node-RED capabilities. Dashboard was divided in multiple tabs:

- **Home tab**: used for overview of all system modules (Fig 7);
- **Main tab**: used to see all information gathered on the main electrical board;
- **Plugs tabs**: used for reading plugs individual information and remotely control relay state. One tab was created for each plug;
- **Solar tab**: used for monitoring local generation;
- **Report tab**: used to search and compare information from the database presented on chart format;
- **Cost tab**: used to compare cost of distinct electricity tariffs;
- **Advanced tab**: used for checking system status, change plugs energy categories and to activate load algorithm;
- **Logs tab**: used to read logging messages;
- **Configuration tab**: used for changing some of the systems configurations, like electricity price and daily e-mail address.

B. Algorithm test

The algorithm was tested multiple times to observe if appliances would be correctly controlled for implemented load control scenarios. Since house inhabitants have a flexible schedule and weather patterns also affect consumption and production, it wasn’t possible to create a fully controlled scenario for the tests. So, it would not be possible to fully replicate all appliances usage and energy production in two distinct days. Nonetheless, by comparing consumption and production patterns of days that had both saving modes activated to days that didn’t, it’s possible to observe clear differences. During the time of the day without local generation, it’s possible to observe that the max load control is able to reduce house peak consumption but the effect of the algorithm is much clearer when solar production is high enough to activate solar saving mode. In this situation, it is evident that power consumption regulates according to local production, making it possible to use more efficiently produced energy.

![Fig. 7. Example of a tab of the user interface, in this case Home tab.](image1)

![Fig. 8. House energy profile (A) Test A: with both saving modes activated, (B) Test B: without saving modes activated.](image2)

Figure 8, shows the results of two tests, where dark blue curve is full house consumption, orange curve is local production, light blue curve is the consumption of a heater and green curve
is the consumption of a fridge. Test A was done with both saving modes activated, solar saving mode was set with a threshold of 400W and max load control with value of 1000W, and the only appliances operated by the algorithm were the fridge and heater. Test B was done as a control test with no power saving mode activated.

In Test A it is possible to see how the algorithm handles the transition from max load saving mode to solar saving mode. E.g. after 16:00 it can be observed that the algorithm has to turn off both loads as the energy production is reducing, and at 17:00 when production is under the 400W threshold, algorithm is able to turn on both appliances since the new limit is the 1000W of the max load control. Also during the time of the day with no production it can be seen how the algorithm shuts off one or both appliances when energy consumption reaches max load limit.

These examples help to demonstrate how the success of the algorithm will greatly vary based on conditions like house energy usage and weather. Nonetheless, other factors like the type of appliances used and the number of smart plugs should also be considered. Better results can be expected when larger number of plugs, connected to flexible loads, are being used and also when they are operating smaller loads, e.g. two heaters of 500W being controlled by plugs will achieve better results than a single heater of 1000W connected to a single plug.

C. Discussion

Since developed system operates in a reactive it will only reduce consumption a few seconds after the limit is exceeded, which might not be an option in some cases.

A system where all major appliances were capable of sending requests to the HEMS for a certain energy quota, instead of starting to consume instantly would solve this limitation. In this system, appliances would have to wait for HEMS response to their request before start consuming. This way the HEMS would be capable of turning off other appliances with less priority before answering the appliance which made the request.

The proposed improvements would avoid load limit being exceeded. However, it would require all major appliances to be capable of sending energy requests to the HEMS and delay or interrupt their operation or specific tasks when asked by the HEMS. A system of this type can’t be obtained using only plugs like the one developed in this thesis, and all the appliances would have to come pre-built with this system.

VI. CONCLUSIONS

In this thesis a Home Energy Management System (HEMS) was proposed, based on low-cost hardware and open-source software. The developed infrastructure offers a tool to monitor house energy consumption and production, while also controlling the state of appliances, in order to decrease the house energy bill and enable the use of locally produced energy in a more efficient way. A Wi-Fi network with MQTT messages was used to implement a communication system between a Raspberry Pi server and multiple NodeMCU based modules that measure house and appliances energy consumption as well as local energy production.

The Node-RED tool was used to create a HEMS platform capable of gathering and storing data collected by power meters, and implement a web interface that gives users the capacity to monitor their energy consumption habits and control their appliances.

The system open architecture and the use of Node-Red allow adding new modules to the system and new functionalities to the platform, making the developed HEMS extendable and upgradeable.

A real-time load control algorithm was created to implement automatic control over appliances, so that house peak consumption is reduced and the use of locally generated power is promoted. This algorithm controls appliances based on their categories and load limit set by the user, and taking into account variation of house energy consumption and generation.

HEMS modules were prototyped and the developed algorithm was tested in a real scenario, proving that the algorithm worked as intended. However, it should be noted that, results for a house are very dependent on appliances use and weather conditions, which affect the use of thermal appliances and energy generation.

Although the developed system isn’t able to avoid consumption from exceeding, momentarily, the maximum load value, set by the user or based on current value of energy production, this fact doesn’t have considerable impact in the electric bill. A future system where all major appliances request the HEMS an energy quota, before turning on, will be able to avoid exceeding the maximum load set, but this is still not possible with the appliances currently available on the market.

Some ideas for future work are: Use of energy desegregation techniques in order to identify appliances consumption based on the main house meter, avoiding the use of individual plugs; Developing forecasting tools for the platform in order to predict house and appliances consumption, local power generation and the electric bill; Create plugs capable of avoiding appliances from start consuming immediately, which allows the HEMS to decide to turn on the appliance immediately or to turn off, previously other appliances, ensuring the maximum load set is not exceeded.

VII. REFERENCES


