Evaluating benefits of adding intelligence to small-scale renewable energy systems

Tara Petric*†, Charlotte Dupont* and Franck Le Gall*
*Easy Global Market
1200 Route des Lucioles, 06560 Valbonne - France
Email: tara.petric@eglobalmark.com, charlotte.dupont@eglobalmark.com, franck.le-gall@eglobalmark.com
†Telecom Bretagne, Rennes, France
Email: tara.petric@gmail.com

***Abstract*—Areas missing wide electricity grid deployment require set-up of local and small scale grids (nano-grids) covering individual or group of houses. Such systems require a performance/cost optimisation. A simulation has been developed to evaluate management algorithms. A 15 % optimisation through the use of simple predictive algorithms is demonstrated.**

***smart energy*—Internet of Things, off-grid energy, optimisation, smart energy**

I. Introduction

More than 600 million people who live in sub-Saharan Africa have no access to electricity. This means that after dark, school children cannot read or study, clinics cannot refrigerate, businesses cannot grow.

Small scale renewable energy is an approach that has been getting much attention lately, with the products such as [1] and [2] already used by the inhabitants of sub-Saharan Africa. It is a complete system that consists of a renewable energy source and storage, and requires no additional electricity sources. More specifically it contains a battery and a solar panel as well as a controller with some computing power to manage the flow of electricity between the various devices. This kind of solution provides its users with their own renewable energy system that can power various electrical devices and appliances. It helps them to improve their way of life, save time, and even start their own businesses in their local communities, offering services previously unavailable or unaffordable.

Bringing intelligence to renewable energy systems is not a new concept. It is already known in the scientific community that scheduling and load-shedding can bring benefits to off-grid, renewable-energy powered systems. In addition, it can make sure that the energy runs longer, and that high priority devices are served, as well as energy waste minimized. It has been explored in research papers [3]–[11] as well as live installations [12], [13].

Even though some of these papers served as a theoretical base for work done in this paper, all of them focus on larger-scale systems that are supposed to serve either a building or a whole community, supporting high demanding appliances (such as washing machine or HVAC systems), and thus needing additional power generators (e.g. running on diesel). Furthermore, none of them freely provide methods or software that can be extended, altered or scaled to apply to this kind of small-scale system. As to our knowledge, in the scientific community, no work has yet been done on how to support and improve these small scale systems that are meant to serve individuals.

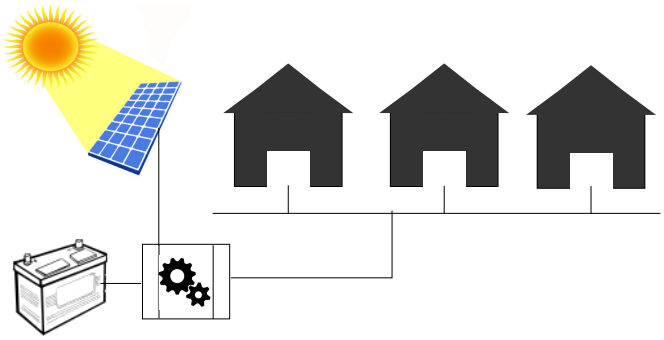

Fig. 1. The simulated scenario

With our work, we intended to change that. To experiment with these kind of solutions, we built an extensible and tunable software simulation. This simulation can be used to give insight into the improvements (in terms of serving more electricity requests without changing the equipment) that can be brought to the system. More specifically - the improvements obtained by introducing intelligence and smart scheduling, as well as making the state of the system more transparent to users. Using our simulation we are able to show, with statistical confidence, that a significant improvement can be achieved by these methods, making renewable energy a feasible option for more citizens of the world. We make this simulation freely available for use for scientific purposes, along with its complete documentation, and the tools used to analyze and visualize the results. In following sections the implementation of the simulation is detailed as well as the results that were obtained from it.

II. The simulation

The scenario showed on Figure 1 is modeled, where one system serves multiple African huts. Although the configuration can be easily altered to serve less or more houses.

978-1-5090-3843-5/17/$31.00

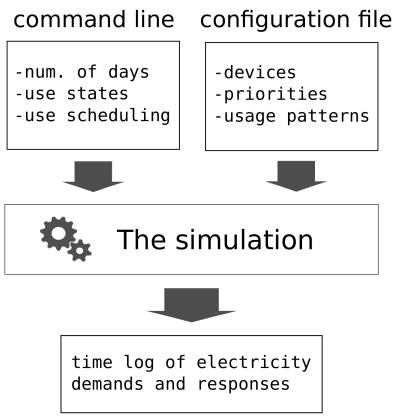

Fig. 2. The workflow of the simulation

The basic components of the simulation are:

- End devices
- Renewable energy storage
- Renewable energy source
- Controller

The end devices are meant to be devices with low-power requirements such as light bulbs, phones, tablets and a low-power fridge. As the renewable energy storage a sealed lead acid battery is used. As a renewable energy source a solar panel is used. The controller has computing power, and can communicate with all the end-devices and renewable energy loads and storage, and is in charge of controlling the energy flow between them.

Figure 2 shows how the simulation works. The user provides the configuration file as well as command line argument that will specify how the simulation will execute. Via the command line one can specify the number of ”days”, as well as the mode to run the simulation. The configuration file, on the other hand, specifies the type of end-devices, as well as their priorities and their usage patterns. The output of the simulation is a time log of electricity demands for each device and the electricity responses that were given by the system. Following is the detailed explanation of these elements.

A. The configuration file

All the end-devices are created based on the configuration file. For each entry it needs to be specified:

- The priority of the device
- In case of devices that are turned on by users at approximately fixed time, like the light bulbs: what is the time period that the device will be turned on
- In case of devices that are turned on by users at random times: when is the user able to turn it on (e.g. based on when he is home and when he goes to sleep)

Based on whether scheduling is turned on or not (via a command line argument) there are two distinctive ways for the simulation to determine when each of the devices will be turned on.

1) Random user demands

The base logic is that users put their devices to charge at random moments (i.e. when they are home, until they go to bed). This is achieved by random choice at runtime from the interval specified in the configuration file.

2) Scheduling of user demands

An option can be set to influence the time that the users will put their phones and tablets to charge. This corresponds to a real-life scenario where users get influenced by the information received from the system (i.e. variable energy cost). In this case, the time is picked from the specified interval, but with a weighted probability. This works in the following way:

Each period that the devices can be turned on (which is specified explicitly in the configuration file), is sliced into sub-periods for the sake of simplicity.These sub-periods were determined from experimental results, in order that, by picking the more favorable periods, the likelihood of all devices being served is maximized. The weights are given to each section, determining the probability that they will get picked in the following way:

- From the beginning of the interval (determined in configuration file) to 16h 80% probability
- From 16h to 19h 10% probability
- From 19h to 20h 7% probability
- From 20h till the end of interval (usually 23h) 3% probability

Sometimes its the case that the starting period set in the configuration file is after 16h - in that case the 80% probability is added to the 16h to 19h period. After the interval is picked with the given probabilities a random starting point is taken from that interval.

B. Running mode with states

It can be chosen, via the command line argument to have the system running with states. This will determine whether certain devices will be running or not, or if they will switch to low-power mode. The priorities of the devices, or, more exactly - how they will behave in each of the states of the system, are defined in the configuration file. The system can be in either NORMAL, WARN or CRITICAL state. These states are determined based on the estimates of the remaining energy at that point in time of the simulation. This is calculated based on the day of the year (which determines how much solar energy is available), as well as the SoC of the battery. It is also possible to run the system in stateless mode.

C. Details of implementation of components.

The simulation is written in the Python programming language [14], using the SimPy [15] library (to implement an agent based approach and provide time dependant analysis). Following is the description of simulation components.

1) Core

The Core module implements the components and responsibilities of the actual real-world controller (that keeps the whole system together and functioning), but also some simulation-specific components that are otherwise handled by real-life (e.g. the user energy demand patterns).

The Core includes the simulation clock implemented as a loop. This loop is repeated every n (command line argument) minutes (in terms of SimPy). So, each time: after deciding the state (if running in stateful mode), the controller goes through all of the devices that are supposed to be ON, and tries to give each one the requested energy.

How much energy to request is decided by the device, based on:

- Its pre-set behavior
- the current state of the system (NORMAL, WARN or CRITICAL)

The controller tries to satisfy the requests from solar energy, and then, if insufficient, from the battery. After this if there is excess energy (coming from solar, and not being used by the devices), it is used to charge the battery. If the battery is fully charged, or has reached its charging-current limit the energy is wasted.

2) Solar panel

Solar Panel behaviour is modelled with the help of PVlib [16] Python library. It uses modules from Sandia [17] (specific solar panel and inverter models) to generate solar data based on:

- time and date
- location (longitude and latitude)
- altitude
- wind speed
- solar panel surface tilt

3) End-devices

There are in principle two kind of devices:

- Charge-ables', that, after they are plugged in - they ask for energy for a certain period (as long as it takes them to charge). These devices can be plugged in anytime, depending on the user behaviour.
- 'Continuous' - they consume power from the time they are switched on until they are switched off (e.g. light bulb or fridge).

All devices have a preset-behavior via the configuration file. This behavior defines how do the devices act for different states of the system (whether they stay on, reduce power consumption or turn off completely, depending on their capability), implicitly giving them priorities, and thus influencing, in a distributed manner, the overall power consumption. Important to note is that the behavior only defines how the devices behave when they are ON. The decision of turning them ON is made by 'users' (for chargeables), or at a pre-configured time (for 'Continuous' devices). The behavior patterns of users are implemented to be either:

- random within limits (depending on the time they are at home and awake)
- highly influenced by the system (with higher probabilities on some time spans than others)

4) Battery

Battery has internal attributes such as capacity, voltage, State of Charge (SoC), loss (when charging and discharging) as well as maximum charging current (limiting the charging rate). In this type of cases, sealed lead acid batteries are the most widely used. We only use 12V batteries in the simulation, while we vary the capacities based on the models commercially available on the market. We estimate the maximum charging current as capacity*0.3, which roughly corresponds to the specification of existing models of these batteries. During charging, the maximum allowed charge current and the SoC of battery needs to be taken into account. When discharging, there is currently no limit for the current, the only limitation is the SoC of the battery. If there is not enough energy, nothing is taken from the battery.

III. Results

To obtain indicative results, we run the system in three modes. We repeated each run 100 times in order to get statistical confidence.

1) Bare-bones, without any states or priorities
2) With added states NORMAL, WARN and CRITICAL, that would cause the devices to reduce their energy consumption or shut down completely
3) With states and influencing users by scheduling the charging of their phones and tablets

To evaluate these results we use as the metric the ratio of satisfied power requests vs. the total number of power requests made (response/request ratio). Following is the discussion of the obtained results.

The simulation is run first in bare-bones mode, in order to construct the baseline against which further comparisons are made. From that baseline, trials are made to improve the response/request ratio by adding (a) states and (b) smart scheduling of user requests.

Adding states. The purpose of introducing states of the system is allowing prioritized devices to run longer by shedding the low-priority devices and reducing the energy consumption of dimmable devices (such as LED based lights). This also improves the overall response/request ratio because the system can run longer if devices are shifted to low-power mode when needed.

Scheduling the charging times. The previous results showed how much we can improve the functioning of the system by the system reacting to its state by adjusting the loads' power consumption. This is the reactive optimization of the system, when already problems are occurring. Interesting is to see how we can use smartness to prevent the problems from occurring in the first place. This can be done by calculating and predicting the effect of a load according to which time

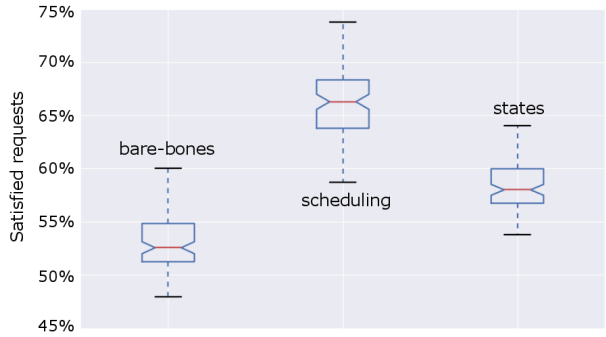

Fig. 3. All devices - ratios of satisfied requests when running the system in three different modes (100 samples per mode)

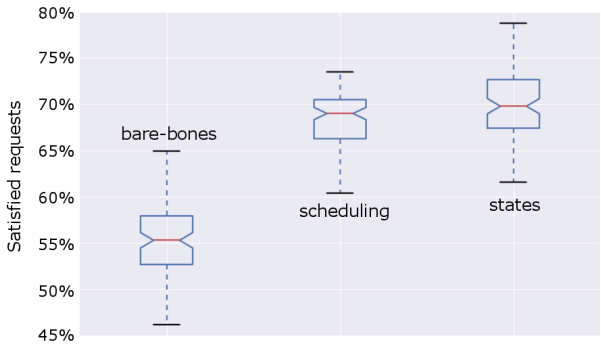

Fig. 4. Priority devices - ratios of satisfied requests when running the system in three different modes (100 samples per mode)

it is plugged in. This would influence the users to charge their smartphones and tablets at different time. We used experimental results whit varying time for choosing the right cut-off time (after which the users are less likely to plug in their phones, because of the suggestions of the system).

A. Result synthesis

The comparison of all results is shown in Figures 3 and 4.

This simulation was run with the following configuration: a single solar panel (100 W) and battery (17 Ah) as well as 15 end devices (5 tablets, 6 smart-phones, 3 light-bulbs and one low-power fridge). We can remark that the overall performance (for all devices) improves by about 5% when adding states, and even more drastically when adding scheduling of the devices with the ratio improving by about 15 %.

When looking only at high priority devices, we can see that introducing states brings a more significant difference (around 15 %). While, on the other hand, we cannot remark any improvement from adding scheduling (on top of states) here.

IV. Conclusion

Using our simulation we could get a better insight about the influences of the following things on the performance of the off-grid renewable-energy system:

- Automatic load-dimming and load-shedding depending on the self-known state of the system
- Influencing the users by suggesting to them when to turn on certain devices

The results show that there can be significant benefits brought by these points. The next step would be a physical implementation of the system to validate these results on a physical installation. There could also be additional benefits by working on the self-knowledge of the renewable energy system and better estimation methods, and more accurate user-suggestions based on machine-learning methods and possibly big-data analysis. Furthermore, the simulation can be easily extended to explore the case when more of these systems are interconnected and can share resources and loads, while taking into account the additional energy consumption of the controlling system.

Acknowledgment

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 687607 (WAZIUP).

References

[1] (2016). BBOX Plug and Play Solar Systems, [Online]. Available: http://www.bboxx.co.uk/ (visited on Dec. 21, 2016).

[2] (2016). Barefoot connect 3000, [Online]. Available: http://barefootpower.com/products/solar-home-systems/item/132-connect3000.html (visited on Dec. 21, 2016).

[3] B. Becker, F. Allerding, U. Reiner, M. Kahl, U. Richter, D. Pathmaperuma, H. Schmeck, and T. Leibfried, "Decentralized energy-management to control smart-home architectures," in *International Conference on Architecture of Computing Systems*, Springer, 2010, pp. 150–161.

[4] B. Becker, F. Kern, M. Lösch, I. Mauser, and H. Schmeck, "Building energy management in the fzi house of living labs," in *DA-CH Conference on Energy Informatics*, Springer, 2015, pp. 95–112.

[5] F. C.M. Colson M.H. Nehrir, "Algorithms for distributed decision-making for multi-agent microgrid power management," 2010.

[6] A. P. García, J. Oliver, and D. Gosch, "An intelligent agent-based distributed architecture for smart-grid integrated network management," in *Proceedings - Conference on Local Computer Networks, LCN*, 2010, ISBN: 9781424483877. DOI: 10.1109/LCN.2010.5735673.

[7] M. Pipattanasomporn, M. Kuzlu, W. Khamphanchai, A. Saha, K. Rathinavel, and S. Rahman, "Bemoss: an agent platform to facilitate grid-interactive building operation with iot devices," English, pp. 1–6, DOI: 10.1109/ISGT-Asia.2015.7387018. [Online]. Available: http://ieeexplore.ieee.org/articleDetails.jsp?arnumber=7387018.

[8] C. M. Colson and M. H. Nehrir, "Comprehensive real-time microgrid power management and control with distributed agents," *IEEE Transactions on Smart Grid*, 2013, ISSN: 19493053. DOI: 10.1109/TSG.2012.2236368.

[9] J. Schoeneberger, R. Duke, and D. Round, "Dc-bus signaling: a distributed control strategy for a hybrid renewable nanogrid,"

[10] D. Koß, D. Bytschkow, P. Kirti Gupta, B. Schätz, F. Sellmayr, and S. Bauereiß, "Establishing a smart grid node architecture and demonstrator in an office environment using the soa approach," in *2012 1st International Workshop on Software Engineering Challenges for the Smart Grid, SE-SmartGrids 2012 - Proceedings*, 2012, ISBN: 9781467318648. DOI: 10.1109/SE4SG.2012.6225710.

[11] B. Nordmand, "Optimizing device operation with a local electricity price," 2014.

[12] (2016). Institute AIFB - energy smart home lab/en, [Online]. Available: http://www.aifb.kit.edu/web/Energy_Smart_Home_Lab/en (visited on Dec. 20, 2016).

[13] (2016). Storage and cross-linked infrastructures - helmholtz association of german research centres. 00000, [Online]. Available: https://www.helmholtz.de/en/research/energy/storage_and_cross_linked_infrastructures/ (visited on Dec. 21, 2016).

[14] G. Van Rossum *et al.*, "Python programming language.," in *USENIX Annual Technical Conference*, vol. 41, 2007.

[15] K. Muller and T. Vignaux, "Simpy: Simulating systems in python," *ONLamp. com Python Devcenter*, 2003.

[16] J. S. Stein, W. F. Holmgren, J. Forbess, and C. W. Hansen, "Pvlib: Open source photovoltaic performance modeling functions for matlab and python," in *Photovoltaic Specialists Conference (PVSC), 2016 IEEE 43rd*, IEEE, 2016, pp. 3425–3430.

[17] D. L. King, *Photovoltaic array performance model*.