Smart Communities of Intelligent Software Agents for Collaborating and semantically interoperable Micro-Grids

Rocco Aversa[a], Beniamino Di Martino[a], Geir Horn[b], Svein Hallsteinsen[c], Salvatore Venticinque[a,*], Shanshan Jiang[c]

[a]University of Campania "Luigi Vanvitelli", via Roma 29, 81031 Aversa, Italy
[b]University of Oslo, P.O. Box 1080 Blindern, Oslo, Norway
[c]SINTEF, P.O. Box 4760 Sluppen, Trondheim, Norway
[*]salvatore.venticinque@unicampania.it

Abstract. CoSSMic was an European project that developed a multi-agent solution for smart management of shared energy by photovoltaic panels. Software agents implements collaborating consumer and producer devices negotiating energy over a peer-to-peer (P2P) overlay. The emergent behavior of the multi-agent system was an optimal schedule of energy consumption. This paper summarize main results of the project including techinques, open source technologies and data.

Keywords: Multi-Agent Systems, Smart Grid, Smart Energy

1 Introduction

The world is facing a climate crisis forcing us all to move away from fossil fuel to renewable energies. However, integrating a large fraction of locally produced and difficult to control renewable energy into the public electric energy grid is challenging [7]. There are many ways renewable energy can be generated, and the economical feasibility depends on local conditions. Wind and waves are often abundant in coastal regions, and most of the world's population lives in areas where insolation is abundant. However, a major obstacle to the profitability of photovoltaic (PV) systems is the misalignment of mid-day peak production and evening and night time energy consumption since this misalignment is normally compensated by expensive batteries.

The CoSSMic project has investigated how energy smart neighbourhoods could contribute to alleviating this problem. An energy smart neighbourhood means a neighbourhood where the buildings coordinate to maximize the self-consumption of the locally produced electrical energy and reduce consumption and feed-in peaks towards the public grid. The project has developed a prototype Cyber-Physical System of coordinated smart micro-grids where human users and intelligent devices collaborate to realize energy smart neighbourhoods, and a simulation facility where the system can be executed in a simulated environment to study the effects of such collaborative behaviour [11]. The core idea is to improve the management of green energy produced by photovoltaic panels by demand side management of electrical loads that can be shifted in time. The objective is to start the loads at times where they can run only on solar energy given the weather forecast predicted energy production. The monitoring and data collecting part of the system has been installed in a number of buildings for more than a year collecting detailed data about energy consumption and production, and this data has been used to simulate the effect of the coordinated scheduling of flexible appliances. The software platform has been implemented using open source technologies, and is available as open source. Different deployment models have been investigated, to exploit the flexibility of Cloud [2], while at the same time taking into account security issues.

2 Architecture

The CoSSMic architecture adopts a highly distributed agent based P2P approach where each consuming and producing device in the neighborhood is represented by an agent. Batteries are represented by a coupled pair of agents, one responsible for the charging and one responsible for the discharging. The agents of a neighborhood negotiate to adapt the consumption to the predicted production shifting loads in time inside constraints set by the inhabitants. This architecture has several advantages. It allows for easy creation and evolution of energy smart neighborhoods not requiring central organizational or computing support. It communicates and device failures. It implements a scalable solution partitioning of an exponential optimization problem. It avoids privacy concern confining private data inside each household. A household is considered as a microgrid and can be a building, a group of buildings or a part of a building. The household has a home gateway, which gateway executes the intelligence based on distributed computing and is also responsible for communication both with devices within the household and with other neighbors. Interconnected gateways represent collaborating microgrids that form neighborhoods. Each microgrid is an autonomous subsystem until it joins a neighborhood. The agents in a neighborhood make up a multi-agent system (MAS) and they communicate with each other directly, *e.g.*, a consumer agent can negotiate with any producer agent in the whole neighborhood. The MAS forms a P2P overlay to support communication and negotiation among agents as well as neighborhood management. The overall architecture for a microgrid is depicted in Figure 1. The *Graphical user interface* allows users to (re-)plan tasks, configure the system, set policies, preferences and constraints, monitor the energy usage of the household and status of the neighborhood, and see how the household contributes to and the benefits from participating in the neighborhood. The *Prediction* component computes forecasts for the PV production and the prices for electric power exchange with the public grid, based on third party services and knowledge about the house and its PV installations. The *Task manager* serves as the master agent of the multi-agent system negotiating the load scheduling. It creates producer and consumer

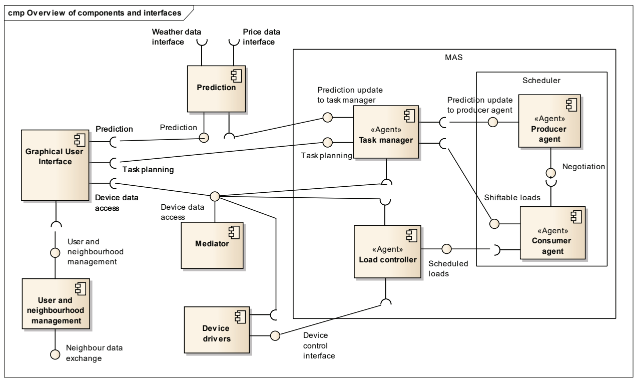

Fig. 1. Overview of components and interfaces

agents to represent the producer and consumer devices of the household, and manages a list of planned tasks for the household. The *Scheduler* implements distributed load scheduling and is responsible for negotiating load scheduling in the whole neighborhood. It consists of Consumer and Producer agents. The *Consumer agents* negotiate power delivery agreements with the *Producer agents* of the whole neighborhood and scheduling of the loads with the other households, and returns assigned start times for the loads. The *Load controller* executes the loads according to their schedule. The *Mediator* provides device management and data storage services. The *Device drivers* are responsible for the low-level integration of and communication with the various devices connected to the system. The *User and neighborhood management* component is in charge of keeping track of the users of the system and their roles and privileges, and the other households that are members of the neighborhood. A more complete description of the architecture is provied in [11].

3 Results and achievements

3.1 Platform

The CoSSMic technological platform has been conceived to run both on general purpose computers and on embedded systems like Raspberry. At lower level a number of device drivers have been developed using different technologies. JAVA and Python programs, but also bash scripts have been implemented to interface the heterogeneous devices installed at the trial sites. Examples of devices are smart meters, smart plugs or inverter of photovoltaic plants. Consuming devices like air conditioners, freezes, fridges, or washing machines installed in the household did not provide open interface and they have always been connected via smart plugs. Drivers implement different communication protocols and connect by heterogeneous interfaces. The CoSSMic mediator is based on

Emoncms[1], a powerful open-source web-app for processing, logging and visualising energy, temperature and other environmental data, developed in PHP. The original software has been extended by integrating new functionalities to the core modules, and developing new modules. In particular the new modules implement the CoSSMic graphical user interface; the installationn and management of smart devices; virtual devices, learning capabilities and the API for load management and communication with the multi-agent system. These new modules complement the API of Emoncms with the CoSSMic API.

The multi-agent system of the CoSSMic platform has been developed using two different technologies, Python and C++. All agents communicate by the XMPP protocol. A light XMPP server runs in each CoSSMic platform providing the needed communication services and to allow for the execution of a single instance of the platform. In this condition the schedule includes just the local devices and the energy exchange is optimized between local consumer and the household's PV plant, if it exists. Two instances of the CoSSMic platform communicate XMPP server-to-server connections. The SPADE2[2] Python library provides to the developers a set of API to develop agents based applications and a light XMPP server itself. One of the python agents is responsible to interface with the content management system. It uses the CoSSMic API and accepts RESTfull HTTP requests to receive from the mediator requests, which are forwarded to other agents. Consumer and producer agents are instead different threads of one process. They have been developed in C++ as a Theron[3] application. They implement the one-to-one energy negotiation described in Section 3.4 that produces, as emergent behaviour of the whoole neighborhood, the optimal schedule of consumptions. The platform has been published open source at *http://bitbucket.org/cossmic_release* as a set of public repositories, each one documented by its own wiki.

3.2 Semantic interoperability

The CoSSMic platform will include a semantic layer that will provide abstract description of the functionalities offered and the data exchanged by IoT enabled devices, appliances and sensors with the Cossmic software components by means of a semantic model realized through ontologies.

Actually IoT enabled devices, appliances and sensors have interfaces that often are proprietary and not standardised. To allow such devices to be seamlessly integrated within the CoSSMic infrastructure, it is necessary to translate their Application Programming Interfaces (APIs) into a homogeneous and interoperable form. This translation has to be specific to the protocol used by every device (and appliance and sensors), and in principle it needs one translator for each device to interface. Such translators will typically in the future be provided by the device manufacturers, or third parties can develop and sell them

[1] http://www.emoncms.org
[2] https://github.com/javipalanca/spade
[3] http://www.theron-library.com

to system integrators developing applications accessing such devices. CoSSMic is bootstrapping this ecosystem by developing such translators for the devices needed for the end-user demonstrations within the project. The main innovation in this activity is the definition of a semantic representation based on the available ontologies for the Smart Energy domain. Despite the multitude of available ontologies, the integration of these for heterogeneous sources of information and various domains is non-trivial. We are surveying available ontologies, and will select, evaluate and combine the most promising of these for the purpose of application integration. Furthermore we are integrating the existing ontologies by complementing them with additional information able to represent the APIs of the involved devices, appliances and sensors, in an agnostic way with respect to the vendor specific protocols and interfaces. Such integrated ontology will be the reference model to enable the definition of agnostic (non-vendor specific) APIs, and (hopefully automated) production of adapters and wrappers allowing a seamless integration and interoperability with the software layer of the CoSSMic platform, and exposing services to external platforms.

3.3 Learning capability

Learning capability allows the CoSSMic platform for modeling and prediction of consuming energy profiles of user's appliances. In particular different device categories have been designed, and for each of them a learning model has been defined. We defined single-run, those devices which have not a periodic behavior and are not usually interrupted, like washing machines and dish-washers. Continuously-run devices have instead periodic behaviors. In this case the internal controller periodically switches on the device according to other parameters like the temperature in case of freezers and heat-pumps. Finally electric-cars are considered as energy storages, which can be charged until they are full or the charge reaches a target level at a different power rate that is between a minimum value and a maximum one. The learning process is implemented as an automata the follows the status of the appliances. The automata input is the power consumed by the device. The transaction from a status to another one is triggered comparing the power value and a threshold that identifies the noise. The automata detects the start and the stop of the device, or can be moved to a waiting status if the run must be delayed. Other parameters are used, like the necessary time under the threshold to detect the switch off, are used to configure the learning process. For single-run devices the life-cycle of the learning automata allows to update of the consuming profile on the end of any run. Collected energy samples during the last runs are used. The learned profile is represented as a b-spline, that approximates the interleaved samples by a set of polynomial curves which minimize the mean square error, as it has been describe in [9]. In the case of continuously run device the lack of open interface to monitor temperature and other parameters, and the unfeasible interaction of the internal controller, motivated the design of a different learning technique. In particular the next working cycle is predicted to be equal to the previous one. With this assumption the switch on of the device can be delayed after the next predicted start time according to the scheduler indication. A maximum delay is allowed and also the daily cumulative delay cannot exceed a fixed threshold. The tuning of this parameters is another issue that has been investigated by experience, analyzing the collected measures. Devices are controlled by switching off and on the same smart-plug that provides energy information. These original learning models and their open source implementation, both as component of the CoSSMic platform and as tools for off-line analysis of collected data are results of the project. Such tools allow for analyzing time-series, for identifying automatically runs and for their supervised clustering to distinguish automatically programs and to filter fault detections.

3.4 Optimization model

The core of the system is a distributed multi-agent system [4, 3] where sofware agents collaborate to produce as emergent behaviour the optimal schedule of executions of users' appliances in accordance with constraints defined by the users[1]. One to one negotiations are used as a solution for the distributed optimization of device schedule [6, 13]. For this reason it can be defined as a virtual market for energy negotiation and brokering [5]. The CoSSMic approach consists of two steps described in the following sections.

When a load is submitted, a Consumer Agent is created in the system, and it will select a producer to provide the energy needed by the load. If this producer has sufficient energy according to its prediction to start the load between the load's earliest start time and its latest start time, it will assign a start time to the load. If not it will refuse the allocation, and the Consumer Agent will select another producer. It could happen that this new load will make a more optimal consumption of a producer's predicted energy production than the than its current set of load assignments. In this case the producer could cancel any of the previously assigned loads. Each of these rejected Consumer Agents will then have to select other producers to serve their needs. The selection process repeats until every load has assigned a start time from a producer. As the Consumer Agents act autonomously, this is a game where each play corresponds to a consumer selecting a producer, and the epoch of the game is the number of plays needed to have a new solution when the system is perturbed by either a new prediction or the arrival of a new load to the system. A particular set of assignments,*i.e.* bindings of loads to producers, is called a *configuration* of the game. The game will always converge to a configuration provided that the grid accepts to start any load within its allowed start time interval given by the load's earliest start time and its latest start time. This condition must be met even if one uses a grid model that is artificially limited in order to avoid peaks, meaning that peak avoidance cannot be guaranteed if there is no feasible schedule for the loads that have selected the grid as their producer under the maximum grid peak limit. Furthermore, the game is cooperative [10], because all the involved consumers and producers jointly tries to minimise the grid energy consumed by the neighbourhood.

A consumer's selection of a producer is carried out using a *variable structure stochastic automaton* (VSSA) [12]: A consumer has a probability vector $\mathbf{p} = [p_1, \ldots, p_n]^T$ with one probability for each producer, the grid inclusive. This vector represents a probability distribution with $\sum_i p_i = 1$. For each play, the Consumer Agent selects a candidate producer according to this empirical probability distribution, and this Producer Agent either accepts to provide energy to the consumer or reject it. Should the system be loaded, and all PV producers refuse the consumer, the grid is the only candidate producer and will be selected with probability one.

Classical scheduling originated in manufacturing disciplines and considers the problem of *assigning* a set of n jobs onto m machines. Each job j is assumed to have a known processing time on machine i. It should be noted that classical scheduling only implicitly considers the resources provided by the machine, *i.e.* the capacity of the machine is reflected in the time it takes to complete the job on that machine. The situation considered here is different in that the "machine" is a PV system that provides time variant resources, and the scheduling problem is to start the time variant "jobs" that are the assigned loads according to the resource availability on the "machine". The load profiles are *continuous* and once a load has started it will have to run to completion,*i.e.* the problem is a non-premptive single-machine scheduling problem. In contrast to classical scheduling problems, the PV Producer Agent may start two or more loads with overlapping execution periods if the predicted production profile allow this. In contrast to the combinatorial assignment problem, a *relaxed* form of the problem is considered here where it is possible to acquire additional resources for PV Producer Agent to supply the loads with energy since it can supplement with energy from the grid. This will guarantee that the problem has a solution, and transform the problem to non-linear programme to find the schedule that minimises the cost of the additional resources, *i.e.* the grid energy [8]. Each PV Producer Agent solves independently an optimisation problem to find the schedule for the loads assigned to it given its predicted energy production, and the distributed optimisation allows better scaling in the number of consumer tasks than a centralised optimisation problem.

3.5 Trials and data

The CoSSMic platform has been deployed at two different trials sites, in province of Caserta (Italy) and in City of Konstanz (Germany). In all installations the CoSSMic platform executed on Raspberry P2B or on Raspberry P3, using respectively the Linux distribution Raspbian Wheezy or Jessie. Delays and bureaucratic constraints limited the installation in province of Caserta to one private building, three public schools and a public swimming pool. Also the kind and the number of monitored devices were limited. A smart meter and a smar plug was installed in the private house and two or three smart-meters were installed in each public building to measure consumptions. Such devices communicate via a zigbee to a wi-fi gateway that allows for reading data of each meter using a Modbus protocol over TCP. The energy production by photo-voltaic plants was measured by using the web interface of the inverters, which were equipped with a network interface. The CoSSMic installations in Konstanz were under the responsibility of ISC (*International Solar Energy Research Center Konstanz*). Trials are more heterogeneous and include a greater number of devices and buildings. Trials include 4 industries, 2 schools and 6 private houses, In Table 3.5 a summary of trials information is provided. We collected data for more than one year in Konstanz and for a much more limited period in province of Caserta. The first column shows the trials id, while the second column contains the number of monitored devices. Then we have the starting date and the last date of the observation period. Finally we have the sum of monitored days and hours for each trial. They are not equals to the duration of the observation period multiplied for the number of devices because of downtime, voluntary disconnection of power grid in the school during nights and in weekends, because some devices have been installed later or because other kind of system failures or maintenance.

trial-id	**devices**	**from**	**to**	**#days**	**#hours**
ce01	2	Oct 20, 2016	Jan 11, 2017	1	18
ce02	4	Oct 17, 2016	Jan 27, 2017	738	17
ce03	3	Oct 26, 2016	Jan 11, 2017	145	13
ce04	2	Oct 19, 2016	Jan 11, 2017	96	6
kn01	3	October 13, 2015	February 8, 2017	1012	14
kn03	6	October 29, 2015	February 9, 2017	1707	14
kn04	20	October 13, 2015	February 8, 2017	8802	14
kn05	1	October 3, 2015	October 17, 2016	368	13
kn06	1	October 21, 2016	January 17, 2017	43	6
kn07	6	April 22, 2015	February 8, 2017	3332	11
kn08	5	April 1, 2015	February 8, 2017	2079	8
kn09	8	December 11, 2014	February 8, 2017	3959	13
kn10	9	October 3, 2015	February 8, 2017	4277	14
kn11	4	October 26, 2015	February 8, 2017	1768	12
kn012	7	October 24, 2015	February 8, 2017	2350	10

Table 1. Summary of trials results

3.6 Simulation and Emulation Tools

The deployment of CoSSMic software on real trials for testing and evaluation purpose introduces a number of drawbacks. On one hand the support of the user to run the devices or to reproduce some relevant conditions for testing or evaluation purpose is needed. However the user is already annoyed during the usual utilization because of faults and limitations that characterize a research prototype. On the other hand, even if only real trials can be used for collecting data, the existing installations provide limitations in terms of number of devices, heterogeneity of devices, number of households and not-deterministic conditions for reproducing testing condition. However for testing purpose the platform can be configured to read data collected in the past, and to write them in the system as the devices would running at the current time. The configuration allows for selecting which devices must be emulated and for each device a different starting date-time can be set. This allows both to emulate the execution of a trials in the past, but also to let it run in laboratory with additional real devices, or virtual ones. In fact application which simulate virtual devices have been developed at the beginning of the project to test the platform without real installations. They are also available as an open source package.

A simulation tool was designed to overcome the limited representativeness of our real trials, finding out more about how the collective and individual benefits of the approach depends on the configuration of the neighbourhood, the accuracy of weather forecasts, the price models of the energy providers, and so on. In this way the analysis of the collected data in the trials could be complemented with data coming from a simulator, where we can replay the observed user and device behaviors, varying a number of other factors, such as the configuration of the neighborhood, the number of PVs, the capacity of the storage systems, the frequency of weather forecasts updates, the price models of the public grid, etc. In addition, using the simulation approach, it could be possible to investigate how the CoSSMic distributed system scales with increasing number of households and devices in a neighborhood. Since the main goal of the simulator is to generate more data for the evaluation stage of the project, the key design constraint of the tool has been to obtain a model that exactly reproduces (i. e. reusing the developed software components) the main steps of the real prototype software such as: the creation of the shift able loads; the negotiation and optimization stage using a distributed algorithm; the production of the scheduled loads. The simulator is based on the discrete-event simulation (DES) model where the system appears as a discrete sequence of events in time. Each event has marked with a timestamp and produces a change of state in the system.

4 Conclusion

This paper provided an overview about the results achieved by the research activity of the CoSSMic project. We presented an innovative system architecture for energy smart neighbourhoods that has been implemented by an open source prototype. Data collected from trial installations in 17 buildings where the monitoring part of the prototype has been installed for up to more than a year and collecting detailed data on local energy production and use. The development activities provided as a foreground also open source tools facilitating the execution and observation of the prototype implementation in a simulated environment. Other results include the analysis of regulations, cost models and tariffs for the electric energy sector and their relationship to CoSSMic energy smart neighbourhoods. The proposed P2P solution demonstrated to be scalable, by partitioning an exponential optimization problem, but still providing good improvements in terms of optimality. Future work aimt at improving the degree of desired stability for further involving the users without annoying them and the engineering of the platform that would enable the road-map designed for exploitation.

Acknowledgements

The project Collaborating Smart Solar-powered Micro-grids (CoSSMic) has received funding from the European Unions Seventh Framework Programme for research, technological development and demonstration under grant agreement no FP7-SMARTCITIES-2013-608806.

References

1. Amato, A., Aversa, R., Di Martino, B., Scialdone, M., Venticinque, S., Hallsteinsen, S., Horn, G.: Software agents for collaborating smart solar-powered micro-grids. vol. 7, pp. 125–133. Springer Heidelberg (2014)
2. Amato, A., Aversa, R., Ficco, M., Venticinque, S.: Cossmic smart grid migration in federated clouds. pp. 103–108. IEEE Inc. (2016)
3. Amato, A., Di Martino, B., Scialdone, M., Venticinque, S.: An agent-based approach for smart energy grids. vol. 2, pp. 164–171. SciTePress (2014)
4. Amato, A., di Martino, B., Scialdone, M., Venticinque, S.: Multi-agent negotiation of decentralized energy production in smart micro-grid. Studies in Computational Intelligence 570, 155–160 (2015)
5. Amato, A., Martino, B., Scialdone, M., Venticinque, S.: A virtual market for energy negotiation and brokering. pp. 162–168. IEEE Inc. (2015)
6. Amato, A., Martino, B., Scialdone, M., Venticinque, S.: Distributed architecture for agents-based energy negotiation in solar powered micro-grids. Concurrency Computation 28(4), 1275–1290 (2016)
7. CIRED Working Group on Smart Grids: Smart grids on the distribution level - hype or vision? cired's point of view. Tech. rep. (05 2013)
8. Geir Horn: Scheduling time variant jobs on a time variant resource. In: Zdenek Hanzálek, Graham Kendall, Barry McCollum, Premysl Sucha (eds.) The 7th Multidisciplinary International Conference on Scheduling : Theory and Applications (MISTA 2015). pp. 914–917. Prague, Czech Republic (Aug 2015)
9. Horn, G., Venticinque, S., Amato, A.: Inferring appliance load profiles from measurements. Lecture Notes in Computer Science 9258, 118–130 (2015)
10. Imma Curiel: Cooperative Game Theory and Applications - Cooperative Games Arising from Combinatorial Optimization, Theory and Decision Library C: Game Theory, Social Choice, Decision Theory, and Optimization, vol. 16. Springer, Berlin Heidelberg (1997)
11. Jiang, S., Venticinque, S., Horn, G., Hallsteinsen, S., Noebels, M.: A distributed agent-based system for coordinating smart solar-powered microgrids. In: Proceedings of SAI Computing Conference 2016. pp. 71–79. IEEE (2016)
12. Kumpati S. Narendra, Mandayam A. L. Thathachar: Learning Automata: An Introduction. Prentice Hall (May 1989)
13. Tasquier, L., Scialdone, M., Aversa, R., Venticinque, S.: Agent based negotiation of decentralized energy production. Studies in Computational Intelligence 570, 59–67 (2015)