Energies **2014**, *7*, 3355-3382; doi:10.3390/en7053355

OPEN ACCESS

energies

ISSN 1996-1073

www.mdpi.com/journal/energies

Review

Reviewing Microgrids from a Multi-Agent Systems Perspective

Jorge J. Gomez-Sanz [1],*, Sandra Garcia-Rodriguez [1], Nuria Cuartero-Soler [1] and Luis Hernandez-Callejo [2]

[1] Faculty of Computer Science, Complutense University of Madrid, Madrid 28040, Spain; E-Mails: sandragrodriguez@ucm.es (S.G.-R.); ncuarter@ucm.es (N.C.-S.)

[2] CIEMAT: Centro de Investigaciones Energéticas, Medioambientales y Tecnológicas, Avenida Complutense 40, Madrid 28040, Spain; E-Mail: luis.hernandez@ciemat.es

* Author to whom correspondence should be addressed; E-Mail: jjgomez@fdi.ucm.es; Tel.: +34-91-394-7644; Fax: +34-91-394-7547.

Received: 10 February 2014; in revised form: 5 May 2014 / Accepted: 8 May 2014 / Published: 22 May 2014

Abstract: The construction of Smart Grids leads to the main question of what kind of intelligence such grids require and how to build it. Some authors choose an agent based solution to realize this intelligence. However, there may be some misunderstandings in the way this technology is being applied. This paper exposes some considerations of this subject, focusing on the Microgrid level, and shows a practical example through INGENIAS methodology, which is a methodology for the development of Agent Oriented systems that applies Model Driven Development techniques to produce fully functional Multi-Agent Systems.

Keywords: microgrid; smart grid; multi-agent systems (MAS); INGENIAS; intelligence

1. Introduction

Following Farhangi [1], when comparing a Smart Grid with the conventional Power Grid at the distribution level, the following extra elements can be highlighted: a sensor and metering network, network nodes with computation capabilities, switches or actuators that allow the grid setup to be changed and the capability of plug in or plug out new devices.

However, adding such elements to any Power Grid does not necessarily make the grid smart. New control algorithms are required that coexist or integrate with standard power management mechanisms.

A first idea could be defining management rules for each computing node and indicate, in each role, which actions (switching on an element, for instance) should be executed depending on the current state of the network, which is determined through the sensor network. This idea is not so simple:

- Plug in or plug out new elements in a grid, and their proper management, needs the control system to be aware of what it is connected and which services it could provide. New energy storage systems or new generators can also be considered. However, there can be a wide variety of devices to consider such as new sensors or new computing nodes.
- Sensors and other metering systems are not always reliable. A sensor can fail, it could be incorrectly installed or could provide different observations. The intelligence system should deal with these issues. Scaling the problem to hundreds of sensors just makes it harder, but processing data in time to take adequate decisions may not be feasible. Even if decisions related to safety and protection are taken directly in the involved devices by specialized hardware, the need of quickly processing (in time) sensor data to take adequate decisions is still a priority.
- Some elements or entire sections of the grid could stop working. The control system should be able to manage these kinds of situations by proceeding in a proper way to preserve the stability of the grid.
- The Power Grid can face warning situations and therefore automatic protection mechanisms should be considered as well.

Thus, the literature on this subject transmits the concerns of the authors about sensor integration, proficiency in the data management, awareness, computing distribution, coordination among devices and so on. Any solution to be implemented in a grid that tries to make it smart should provide answers to how such capabilities could be incorporated in a coherent way.

Traditionally, when people talked about "electrical grid" they used to refer to the interconnected transmission system. However, the term "Smart Grid" has been more oriented to the entire electrical system including generation, transmission and distribution. Regarding the distribution system, several efforts target the increase of manageability and efficiency by dividing the smart distribution grid into sub-systems. Such sub-systems are called "Smart Microgrids", or just "Microgrids", and consist of energy consumers and producers at a small scale and are able to manage themselves [2]. Examples for Microgrids may be, for instance, villages, industry sites, university campus, *etc*. Furthermore, a Microgrid can either be connected to the backbone grid or to other Micrograds, or it can run disconnected to the electricity grid in island mode.

Since the distribution system is considered as the largest and most complex part of the entire electrical system [3], an extensive literature is focused on Smart Grids located at this level. For this reason, we want to contribute to this research by identifying the most recurrent problems within these kinds of grids. Furthermore, we suggest an improvement of the multi-agent control system, designed by Pipattanasomporn *et al.* [4], for managing Microgrids.

A first hypothesis of this paper is that an important part of Smart Grid researchers thinks that agent research is a field capable of providing preliminary solutions to the above mentioned issues. Smart Grid-related papers are frequently found, which do assume, like Farhangi [1] and Amin and Wollenberg [5] surveys, that intelligent agents are the right technology for Smart Grids. In fact, the IEEE

Power Engineering Society Intelligent Systems Subcommittee created the group the IEEE Power and Energy Society Multi-Agent Systems Working Group (MASWG) to identify and disseminate the details of key technical challenges for the proper use of the technology within the power engineering domain.

Since the initial assumption seems to be valid, the paper focuses on a second hypothesis: the use of agent technology made so far in Microgrids is not following the major agent oriented development trends. The fact that agent technology is relevant does not imply that the problem is solved. Actually, the agent research community has been holding debates to determine which way is the best one to apply this concept in current grids. These concerns have been dealt in part with the advent of Agent Oriented Software Engineering (AOSE). Nowadays, a developer can find many methodologies that could be applied to real problems and that are supported by software tools. The use of such methodologies to model the agent oriented part of a Microgrid could improve their management and the better realization of the Microgrid concept.

This paper contributes with a discussion on how agents can be applied to build Microgrids in Section 2. It enumerates some principles that can aid a developer to apply the agent paradigm in a more effective way. Then, it illustrates the idea with an enhancement made on an existing work from Pipattanasomporn *et al.* [4] in Section 3. This work was chosen because for being a highly cited example of the application of agent technology to Microgrids. The experiment was made using INGENIAS (INGENIería de Agentes Software or its equivalent Software Agents Engineering) Agent Oriented Methodology [6] as mean to achieve a different Microgrid than the original work. A first attempt is shown in Section 3 where it does not incorporate any Microgrid simulation. An expanded version, where a Microgrid is simulated, is presented in Section 4. Afterwards, there are some conclusions extracted from this research.

2. Intelligence in a Microgrid

To begin with, addressing the intelligence in a Microgrid implies deciding first when one Microgrid will be considered intelligent. The work from Brooks [7] can answer this: "the intelligence is in the eye of the beholder". Brooks studies how research in Computer Science has influenced in the understanding of what Intelligence means for biological systems. Through observation of dynamic environments and how intelligence could occur in those places, the author concludes that artificial intelligent entities are intelligent because they behave, with respect to an external human observer, accordingly. However, the way this behavior is obtained does not necessarily require complex reasoning engines. As a proof of concept, Brooks [8] built his Subsumption architecture. This architecture was a set of layers made of individual automaton.

Therefore, assuming there is a Microgrid that could behave intelligently according to some external observer, the next step is identifying what behaviors current literature considers as intelligent ones. The way to achieve such behaviors is the subject of the rest of the section.

Farhangi [1], and Amin and Wollenberg [5] state that the intelligence is reached with the incorporation of agents to the grids. This is not an isolated fact. A quick inspection of keywords in Google Scholar returns 19,600 documents having the term Smart Grid, whereas Smart Grid and agent together returns

13,500 hits, which is rather significant. Therefore, there is a concern that agents and Smart Grids are highly related, though how these agents do provide this intelligent behavior is still to be determined.

The agent term seems to be used in a different way from journals specialized in agents, such as Journal of Agent Oriented Software Engineering or Journal of Autonomous Agents and Multi-Agent Systems. The term agent is used as an equivalent of actor, like an abstraction of an entity that does things in an intelligent but undetermined way. There are works, like Farhangi [1], and Amin and Wollenberg [5], supporting Agents, but they do not tell or point out what an agent is for them. Moreover, they do not contain references to published works in conferences or journals specialized in this topic.

The IEEE Power and Energy Society MASWG studied this problem too. They issued two reports which were written as summaries of existing agent research results rather than guidelines for the application of Multi-Agent Systems (MAS) to the Power Grid. The first one [9] justified the interest of agent technology for Power Grids, concluding that a MAS can be used either as a way of building robust and flexible hardware/software systems or as a modeling approach. Also, the report indicated intelligence is realized by the incorporation of known AI techniques. MASWG point at areas where MAS can contribute such as distributed control, modeling and simulation (e.g., energy markets), protection, and monitoring and diagnosis. The second report [10] is more relevant to this paper since it deals with techniques and tools that could allow engineers to use MAS. The second report starts with an analysis of the FIPA (Foundation for Intelligent Physical Agents) standards for communication and interoperability. It puts special emphasis on using an ontology for Power Grids, more precisely the one previously mentioned. In theory, this ontology could facilitate the understanding of different MAS in a Power Grid. The design of the MAS, which is the focus of our contribution, has also a dedicated section, but smaller than the previous ones. The authors in [10], do not provide detailed recommendations but references to some AOSE methodologies. The also use some words about agent architectures (agent anatomies in the paper) and recognize JADE [11] as the main agent platform implementation. The report also says that there is insufficient information to support a recommendation of any particular agent anatomy. Furthermore, MASWG seems not to have had any activity since 2011, so no additional research could be found on this topic. Anyway, the results provided by the two reports, in our opinion, are not clear guidelines or recommendations of how to apply MAS in a Power Grid. For instance, blueprints of specific and recommendable configurations of MAS for Power Grids are missing. However we consider that the two reports are a very relevant starting point. Our contribution could be considered as a continuation of this effort since we show how we apply AOSE solutions to a particular problem and discuss the differences regarding the original version.

Some of the participants of the MASWG, later on, issued a new report [12] that focused on the performance of existing agent based solutions for Power Grid control. They briefly refer to AI techniques for power flow management, AI techniques for voltage control and AI techniques for restoration. They also consider how intelligent systems can be used to realize a Smart Grid architecture. More precisely, they consider distributed and robust deployment but also Self-Management through condition monitoring and selective devolved control. We share these concerns too, but again, we feel a critical analysis of current MAS solutions needs to be done in the context of Smart Grid. A system providing the features cited in papers like [12] are not necessarily made following trends in agent research or conventions in

AOSE. We are not defending that the latter the better, but just that they are the result of a deep discussion on how agent features could be designed.

In the following lines we do a quick review of some works that propose the use of MAS as a solution, focusing on the problem of Microgrids management. As it can be observed, engineers are mostly using an agent platform, such as JADE, to develop a solution. As we discussed at the beginning of Section 2.1, using an agent platform does not mean it is being used properly. Just as if a programmer uses an object oriented language does not imply object oriented programming is being made. The following cases are examples of this argument.

Pipattanasomporn *et al.* [4] is a highly cited work which contains meaningful references to agent research and tries to translate those concepts to Microgrids, focusing on the energy distribution level. They propose Intelligent Distributed Autonomous Power Systems (IDAPS) that pursues to control a blackout. It is developed using ZEUS [13] which has graphical environment that facilitates the creation of such systems. The Microgrid is modeled in Matlab/Simulink and connects the agents as actuators. The intelligence, from an external point of view, consists on a mechanism for transition from connected mode to island mode in case of a blackout. However, this paper does not regard the scalability of the solution and how agents do interact to solve problems. In fact the control is embedded in a single agent which is the one that takes the decisions.

Xiao *et al.* [14] also suggests to combine Matlab and ZEUS to realize a MAS controlled Microgrid. They propose a hierarchical control system to replace a SCADA (Supervisory Control And Data Acquisition) based system, both are frequently centralized solutions. The hierarchy that Xiao *et al.* defend creates different levels of responsibilities: a distribution master station level, a distribution substation level and a distribution terminal level one. Authors follow the ZEUS steps in a more explicit way than [4] and deploy the system in a distributed way across several computers. However, the solution resembles to Pipattanasomporn *et al.* [4] at the lower level. There is a Distributed Energy Resource (DER) agent and a Load agent and Substation agent. They play similar roles than the [4] approach. There is also a coordination protocol among levels but it is not detailed. Moreover, some designing artifacts of the solution are missing in the proposal. We find again that the expected intelligence of the system is not discussed. The experiment implements a straight deployment with few agents. Unfortunately, there is short information that allows to reproduce the experiment.

Oyarzabal *et al.* [15] addresses a Microgrid management system built using agent technologies. Authors use JADE platform to develop the system, but they are not using a disciplined approach. They identified up to ten agent types being each of them responsible of a specific part of the system. This is a design that seems to follow a familiar organization of roles in the Microgrid. Unlike others, here there seems to be a concern for identifying different agent types but it does not address a behavior beyond as specific protocol implementation. Each agent implements part of one algorithm for the control of the system but there is no concern for the role of intelligence in this kind of approaches. The work succeeds in the experimentation to prove that a JADE based implementation of such systems could work at least in terms of scalability. However in this solution agents do represent mainly a division of labor. The benefits of agent technology in this case are similar to those ones of other distributed programming technologies. CORBA (Common Object Request Broker Architecture), for instance, could have been used to distribute responsibilities in a similar way.

Jiang [16] introduces an agent based control framework for DER Microgrids. One of the reasons for using agents is that these kind of Microgrids are in favor of energy sources distribution, and therefore its control should be distributed as well. The author suggests three kind of agents: an agent representing the energy source, a second one that represents the energy storage and a third one to represent a load. In this case, no agent oriented technology is cited or how the inherent features of the agents are relevant in the experiment. This kind of contribution is frequent; for instance we can find them in agent conferences, such as the "International Conference on Autonomous Agents & Multiagent Systems" (AAMAS), which are not the most relevant ones.

However, it is possible to use agents just as a convenient concept for thinking about the problem. Afterwards, this concept can be realized through agents or any other alternative technology. This is the case of Ramchurn *et al.* [17] that studies a strategy for reducing peaks of demand on behalf of consumers by using agents. However, the authors do not cite the technology used to execute the experiments. Therefore, it is hard to evaluate the relevance of agent research in the experiment itself. We also found a short design in these approaches, but the way the agent technology is used cannot be evaluated properly.

Kouluri and Pandey [18] focus on Microgrid control using agents too. JADE is the chosen agent platform. The authors identify three kind of agents: the control agent, the DER agent and the load agent. There is also an flow chart representing the activities in the system. These activities are coded within the abovementioned agents. We assume that this flow chart summarizes the expected intelligence of the system. In this case, this chart permits to issue the appropriate orders to each agent in order to handle the blackout. In the experiments, agents are connected to a Simulink instance, which realizes the Microgrid simulation. Simulink provides the input to the agents and also executes the orders generated by them. Like in [4] authors discuss the ways that the system can make the transition from island to connected mode. Despite the interest of the contribution, there are no details of specific agent features. Like other similar papers, we think that this kind of experiment proves the middleware capabilities of JADE, mainly. Other more inherent agent features, such as peer-to-peer communication, the use of ontologies, autonomy, or mobility, are missing.

Dimeas and Hatziargyriou [19] present a MAS for the control of a Microgrid which is an evolution of a previous work [20] and it is implemented in JADE. The proposal distinguishes between three levels: the distribution network operator (DNO) and market operator (MO), at the level of medium voltage and coordinating different Microgrids; the Microgrid central controller (MGCC), which acts as mediator between the Microgrid and the network/market; and the Local Controllers (LC), which controls DERs, production and storaget units, and some local loads. The market side part, influences the production in a Microgrid taking into account that energy could be bought or sold. There are roles identified for agents with specific responsibilities. However, within a Microgrid, there will be only one MGCC and several LCs. In the system, there will be only one DNO, MO, but several MGCC (one per simulated Microgrid). Authors compare a previous solution [20], where control in the Microgrid was centralized in the MGCC, against another that delegates control to the LCs and coordinates them through auctions at the market level [19]. Authors use auctions as main mechanisms to decide what to do, *i.e.*, the main mechanism to produce the desired intelligent behavior. Authors also use ontologies to facilitate inter-agent communication, though its benefits are not explored in detail. The coordination issues between the Microgrid and the market are studied, too, though there is no formal solution for this issue, only ideas of how to deal with it.

Kumar Nunna and Doolla [21] propose a MAS architecture for modeling the electricity market and implements a prototype using JADE. Agents are distributed in two levels: field level, to manage individual micro-grids to match the supply and demand, and market level, to cater the trading among the Microgrids and with the grid. The authors propose the matching of production and the demand through auctions implemented as FIPA auction protocols over the JADE platform. Simulated Microgrids are capable of storing/releasing electricity to the grid. There is a scheme of the relevant agents and, unlike others, authors show test cases involving more than one Microgrid. The solution is not completely decentralized since there is only one market, with its corresponding agents, that launches the auctions. In fact, it is this market the one that coordinates the bids and decide what to be produced/consumed.

In [22], a decentralized control architecture for Microgrids is presented. Even though the authors describe the system as a MAS, its architecture is not introduced. Authors introduce two scenarios where the Microgrid has to adapt to new conditions, like loss of solar resources. JADE platform is used to implement the communication interface between agents and the Microgrid, which is simulated with MATLAB/Simulink. The expected behavior is a complex one, close to what one may consider "intelligent". According to the text, the complexity arises because of the the variations of prices (both sell and buy) of energy in the utility market. This variation enforces that a Microgrid, sometimes, prefer to sell energy or to buy it. Also, the solution is a self-organizing one where there are negotiations at several levels. The proposed solution is based on a series of individual utility functions that realize the decision making process. These ideas seem close to the results agent research pursues. However, no system blueprints are provided. The benefits of having used JADE are not discussed either.

Logenthiran *et al.* [23] present a MAS for real-time operation of a Microgrid. The multiagent system consists of several distributed generator agents, load agents, a renewable energy source agent, a storage system agent, a microgrid manager agent, a schedule coordinator agent (SC Agent), a demand side management agent (DSM Agent) and other administrative agents. Comparing to other approaches, SC Agent and DSM Agent can be highlighted. The SC Agent executes a two-stage scheduling: day-ahead and real-time scheduling while the DSM Agent performs load shifting before the day-ahead scheduling and reduced the load in real-time whenever it is necessary and possible. As in many other works, a design of the MAS is not presented although JADE platform is used to implement the system. The number of instances of each agent is limited as well. Load agents are expected to be found in undetermined numbers. However, the paper only mentions one SC Agent.

In conclusion, we can observe that the cited works utilize an agent platform implementation to develop their MAS, being JADE the most used (see [15,18,19,21–24]). However, we consider that the reviewed works do not make an extensive use of the key features of agent technology in the context of Microgrids:

- In most cases, there is one instance of each agent. Hence, authors do not consider any interaction between similar agents. As we will show later, the possibility of having more than one instance of an agent may affect positively the performance of the system in the case of multi-agent systems [25].
- Agents do have little choice or the choices are not evident in the approach. Many of the cited papers consider the autonomy of the agent as a relevant feature. However authors are not making

explicit the alternatives that each agent has got or even considering that an agent refuses to produce electricity, for example.

- Decentralization is done through distribution. There is a control centre which can be allocated in different places, but there is just one control element. Decentralization happens when the control is distributed along the different existing agents. A trivial case would be having more than one control element. The reviewed literature is mostly centralized eventhough proposed MAS are distributed. There is a tendency towards having a distinguished agent that controls or coordinates everything.
- Agent to agent relationship is often a client-server one. The Peer to peer approach, which is more natural to agent technology, is not present in the reviewed papers. By using JADE, any agent can address any other agent. However in the reviewed papers agents can address other specific agents and only in a very particular way . Furthermore, in the selected papers, agents play just one role. This reduces the flexibility that could allow a purer MAS solution.
- There are no blueprints. What authors introduce as design is usually a graphical depiction (custom made) of key elements in the system together with annotations. These graphics are not useful for detailing an agent based system and there are already methodologies for capturing and specifying MAS since, at least, 2002 [26].

2.1. Reflections on the Use of Agents

The transition to new technologies implies having discussions about what approach works better. It happened with the transition from unstructured to structured programming (which eliminated the GOTOs statements among others) and the transition from Modules to Objects. The latter is particularly relevant to this problem because it leads to a book [27] about how Object Oriented Programming was wrongly used. It could be considered that this book inspired Wooldridge and Jennings [28] to apply the same principles to the world of agents. Wooldridge and Jennings identified some frequent approaches which they considered bad practices or, at least, issues to consider. Among them, there are the following stances:

- Believing that agents can solve any problem;
- There are too many agents or not enough ones in the solution;
- Agents are oversold: “the design is good just because it has agents”;
- Becoming dogmatic about agents or an agent evangelist: “agents can do anything!”;
- Designers do not know why they are using agents. In fact, other technology may suit better the problem.

The first and the last points are specially significant because of how easily is to fall in one of them. They are not technological mistakes but a result of the attitude of the developer. As a guideline for developments and to prevent falling into one of those traps, the key could be to use software engineering. Gómez-Sanz [29] recommends three basic principles:

- Agents are programs at the knowledge level. Newell [30] say that “an agent is a program that processes knowledge”. Moreover an agent satisfies the rationality principle. This principle states an agent which chooses those tasks that allow the Agent to reach its goals. Choosing goals, finding ways to achieve them and determining how the agent can create new knowledge are all the elements that, together, make an agent something different.

- An agent is a container for Artificial Intelligence solutions. The main text book in Artificial Intelligence, Russel and Norvig's [31], presents agents as the wrapper for algorithms created within classic AI. The kind of intelligent behavior and the techniques to achieve it can be obtained in the AI literature. This is useful when a single agent is the one to show intelligence. However, when the combination of individual behaviors is the goal, classic AI is less useful. Distributed AI is another line and can be regarded as the beginning of agent research. Relevant agent based systems have rarely only one agent, they usually have many.
- The Hunhs and Singh's [25] test assumes that adding a new agent to an agent oriented system will alter its global behavior in some way. The improvement of the system performance depends on the agent and its purpose. The test precisely proposes to replicate instances of the existing agents trying to find these improvements. It also suggests having an utility function to be defined before starting the test in order to measure this fact. When there is none may be due to the fact that the agents are not aware of the existence of others or that any benefits are reached from interacting with them. In some way it means Agents are not behaving as they should.

3. The Discipline of the Development of Multi-Agent Systems

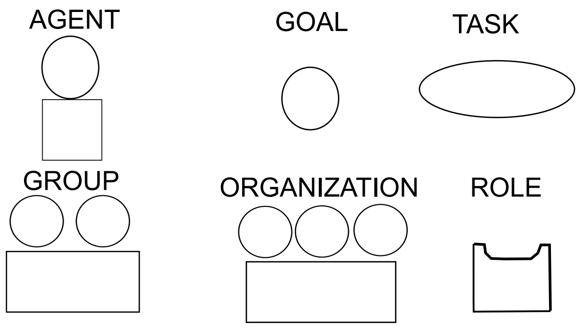
Agent Oriented Software Engineering (AOSE) intends to create a development discipline and normalize the use of software Agents. AOSE is gradually incorporating Model Driven Development practices. This implies that the developer must specify the problem using some modeling language and then conceive how this model is transformed either into other models or into executable code. As a result, current AOSE methodologies can produce full MAS and/or provide an advanced starting point for the development.

However, the use of AOSE methodologies does not guarantee that bad habits disappear. Many of the issues pointed out in [28] (see Section 2.1) are caused by the attitude of the system designers rather than the technology itself.

INGENIAS [6] is an Agent Oriented Methodology and it is supported by the INGENIAS Development Kit [32]. It produces JADE [11] Agents and it is the most widespread platform for agent development. Hence, it is possible to take advantage of existing software developed for JADE. INGENIAS has its own notation for representing MAS (see Figure 1). INGENIAS can switch or mix this notation and UML notation with stereotypes. The latter is frequently used in documents for people with previous experience in this methodology.

In this paper, INGENIAS is applied to a case obtained from a frequently cited conference paper introducing IDAPS [4]. The refactored version will be named INGENIAS IDAPS or I2DAPS. It illustrates a few modifications to the original work that show a different, more agent-like, approach to get similar results to the original work. It is not our purpose to show I2DAPS is better than IDAPS. It is an example of what could have been done if ideas from Section 2.1 had been used.

Figure 1. Meaning of some icons used in the system specification with INGENIAS.



There were four agents per Microgrid in the original work:

- User Agent. A gateway for interacting with the user.
- DER Agent. It registers a control agent, communicates with other DER agents, senses and reacts to measurements, controls the associated DER, informs of power production and receives requests from the user agent.
- Control Agent. It registers elements from the Microgrid, monitors the system, analyses the situation in the system, stores data about the system, sends signals to the main circuit breaker when grid failures are detected and provides the user a graphical representation.
- Database (DB) Agent. It registers information that is produced in the system.

IDAPS used ZEUS [13] to create the system. ZEUS was a reputed WYSIWYG ("What You See Is What You Get") editor for MAS that was discontinued many years ago. Anyway its quality is still outstanding. Therefore it could be said that IDAPS was built using solid agent oriented tools. ZEUS was also the name of a body of recommendations on how to use the tool to build systems. IDAPS paper does not elaborate on the application of ZEUS but there are some decisions which were not justified such as stating a DB should be seen as an agent. ZEUS also underlines the importance of defining the deployment but this information was not included in the paper beyond citing there was one instance of each agent type. Hence, there are two elements to be evaluated that could lead to important differences: the way the agents are used and the way the results are presented.

The refactored I2DAPS is introduced first in Figure 2.

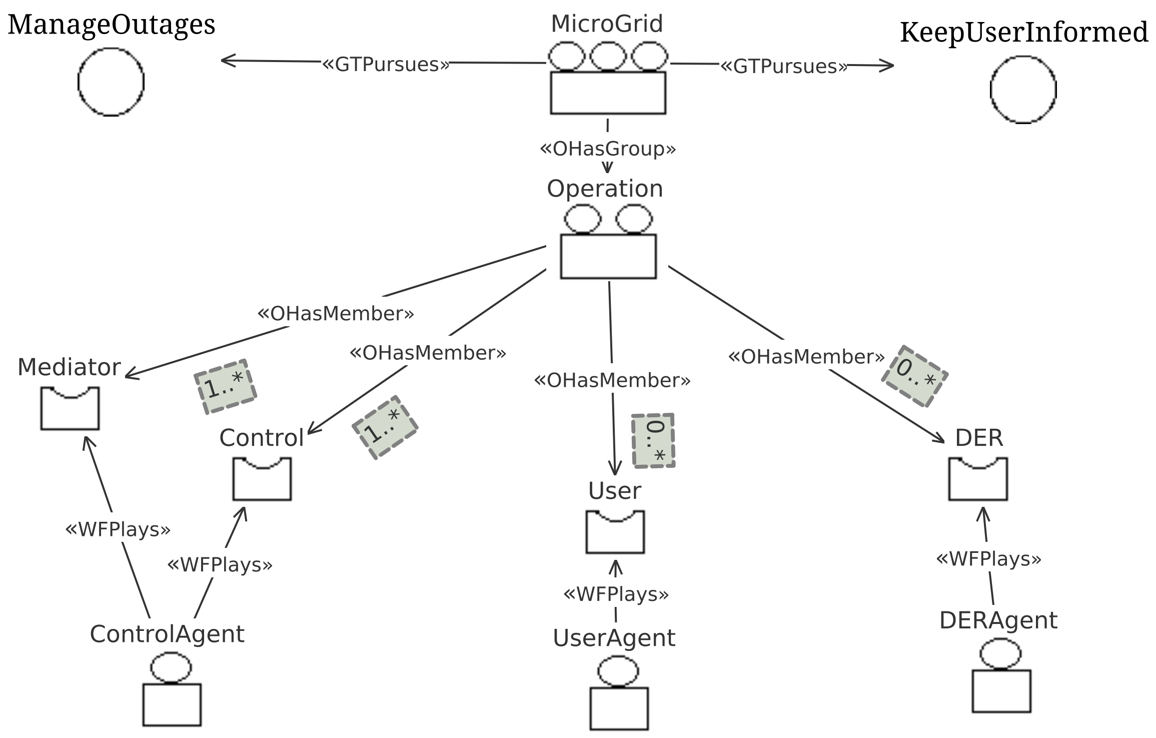
It uses INGENIAS notation and identifies the roles of the agents in the system together with the context of these roles. The most meaningful changes are:

- In IDAPS, there was no first class representation for the Microgrid. Agents were not declared as belonging to a Microgrid either. They just existed and it was the actual programming what made them aware there were other Agents belonging to the same Microgrid. By associating these Agents to the group and organization, this issue becomes transparent for the reader. The organization will be the Microgrid and the distinct activities involved into its operation left to specialized groups.
- We have added an Operation group to distinguish between the different business and management processes that a Microgrid requires. Other groups may realize other aspects, like Billing.
- Roles have been included in the specification. ZEUS methodology does include the concept of role but the paper describing IDAPS is omitting that part. A role is an aggregation of tasks, goals

and information that can be assigned to different Agents so that we achieve an important degree of reuse. In our refactored solution, we distinguish between Control, User and DER roles. Agents will play these roles: all of them, some or just one. This gives the designer more freedom for defining what kind of Agents will exist. There is also one more role, the Mediator. This role was included to achieve a more peer-to-peer solution and represents the agent that promotes the obtaining of consensus as it will be introduced in the next section.

- The DB Agent of the original paper has been eliminated. It did not seem to be a relevant agent if the definition from Section 2.1 was used. A DB does not need to satisfy the rationality principle; in INGENIAS, for instance, the DB fits better as a resource used by some other existing agents or associated to any group. If it is associated to the group, all agents belonging to it will have access to the DB.
- There are explicit goals. The new system defines goals the system must achieve. The first goal is to keep the user informed of what is happening inside the Microgrid. The second goal is to manage blackout and electricity outages. System actions and roles are related to these two goals, so their purpose is explicit.

Figure 2. Agents of I2DAPS associated to an organization. (DER means "Distributed Energy Resource").

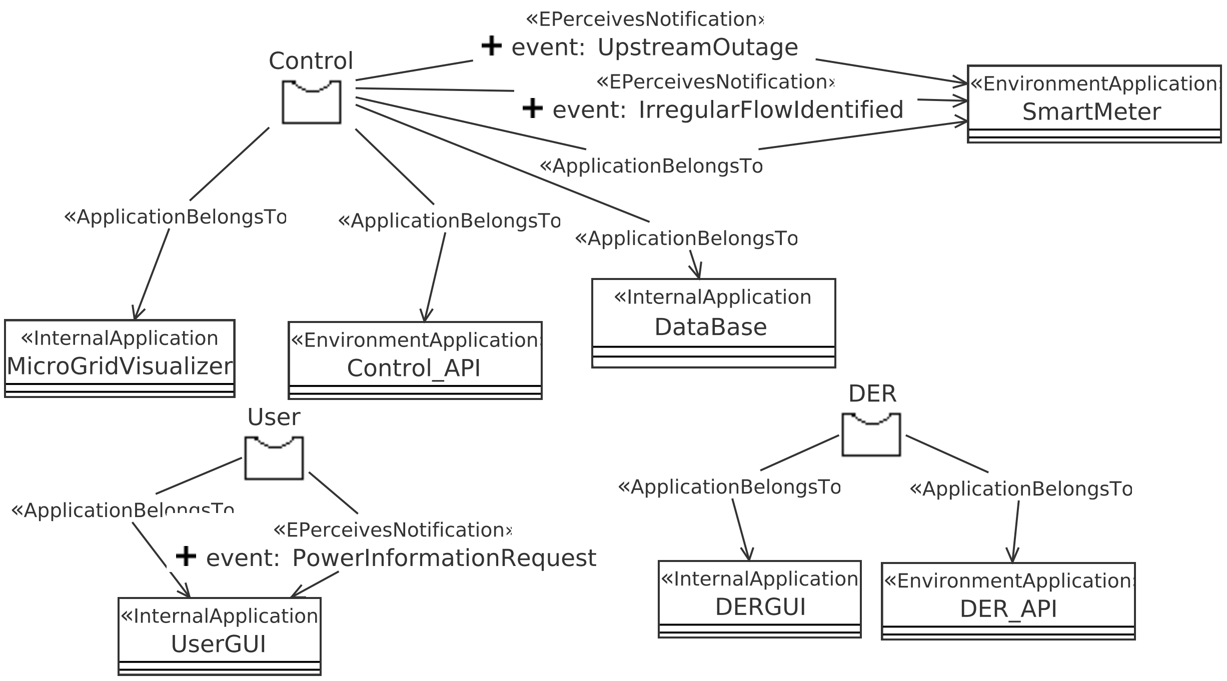


These ideas are presented in Figures 2 and 3. Figure 2 represents the Microgrid with the organization Microgrid. Being an organization, it is granted the entities can subscribe/unsubscribe from it. Moreover the diagram establishes that the Operation group must be formed by exactly one agent playing the role Control, any number of Agents playing the role of User and any Agents playing the role of DER. For each role, the same diagram identifies three different Agents. In this case, we assume that the Control

Agent is playing two roles. This enables this agent to act as a mediator or as a controller. This is another example of flexible design which was not found in papers from Section 2.

Figure 3 explains the resource allocation per role, something that at the IDAPS paper was made in plain text. These resources are grid control libraries (e.g., for activating circuitbreakers), DER libraries (for accessing DER infrastructure), visualizers for the user, meters and some GUIs (Graphical User Interfaces). The previous elements may provoke some interesting situations in the grid that should be controlled such as the existence of an upstream outage or the detection of a low quality of the electricity flow.

Figure 3. Resources associated to system roles. (API: "Application Programming Interface"; and GUI: "Graphical User Interface".)



As a key difference from works cited in Section 2, there is no need to represent as Agents something that it is not. This is the case of the above mentioned DB, but there are others as the way the agent interacts with the DER or any other hardware element.

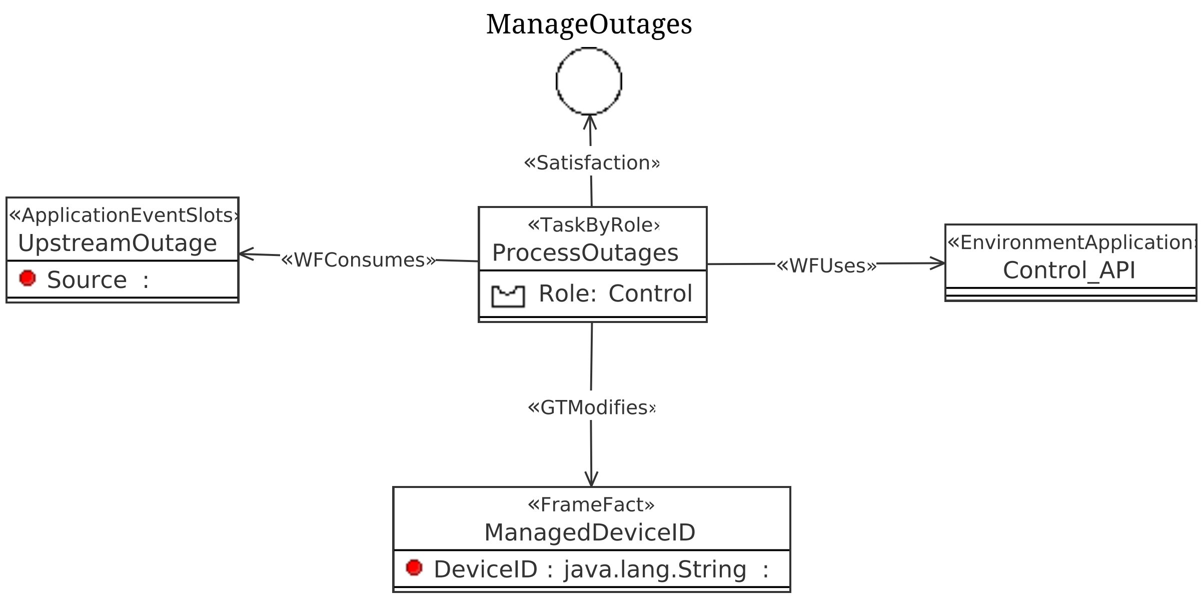
Readers can find links between this solution and the hierarchical approach from [14]. The definition of groups helps to define controlled scopes for the agents to interact. The use of these groups will be more evident in next section where a deployment with multiple groups is used.

IDAPS proposes three basic functions: registration/deregistration of devices into the Microgrid, managing blackouts and informing the user on DER status.

The first is a function already provided by the organization in INGENIAS. IDAPS is not dealing with the situation whether the registration implies something else besides annotating there is one more element. This information may be useful elsewhere, for instance, to take later decisions. In any case, organizations in INGENIAS already provide subscription mechanisms and the organization information can be used to limit the scope of agent interactions (see Section 4).

The second is represented in Figure 4. According to Pipattanasomporn [4], there is a single agent in charge of managing the blackouts through the clever use of circuitbreakers. We translate it as a single task that, given a blackout event detected by low level hardware, it proceeds to create an island and isolate from the main lines. The triggering event is the upstream outage that is produced by the applications from Figure 3, in particular, by the SmartMeter. The task assigned to the control role uses the control API (library) to enable the circuitbreakers. The ManagedDeviceID element informs the tasks which device is assigned to this agent for its management. Each agent can be assigned to different devices so that we need to make explicit this assignment. In addition, it is made explicit that this task contributes to the achievement of the goal Manage outages, already identified in Figure 2.

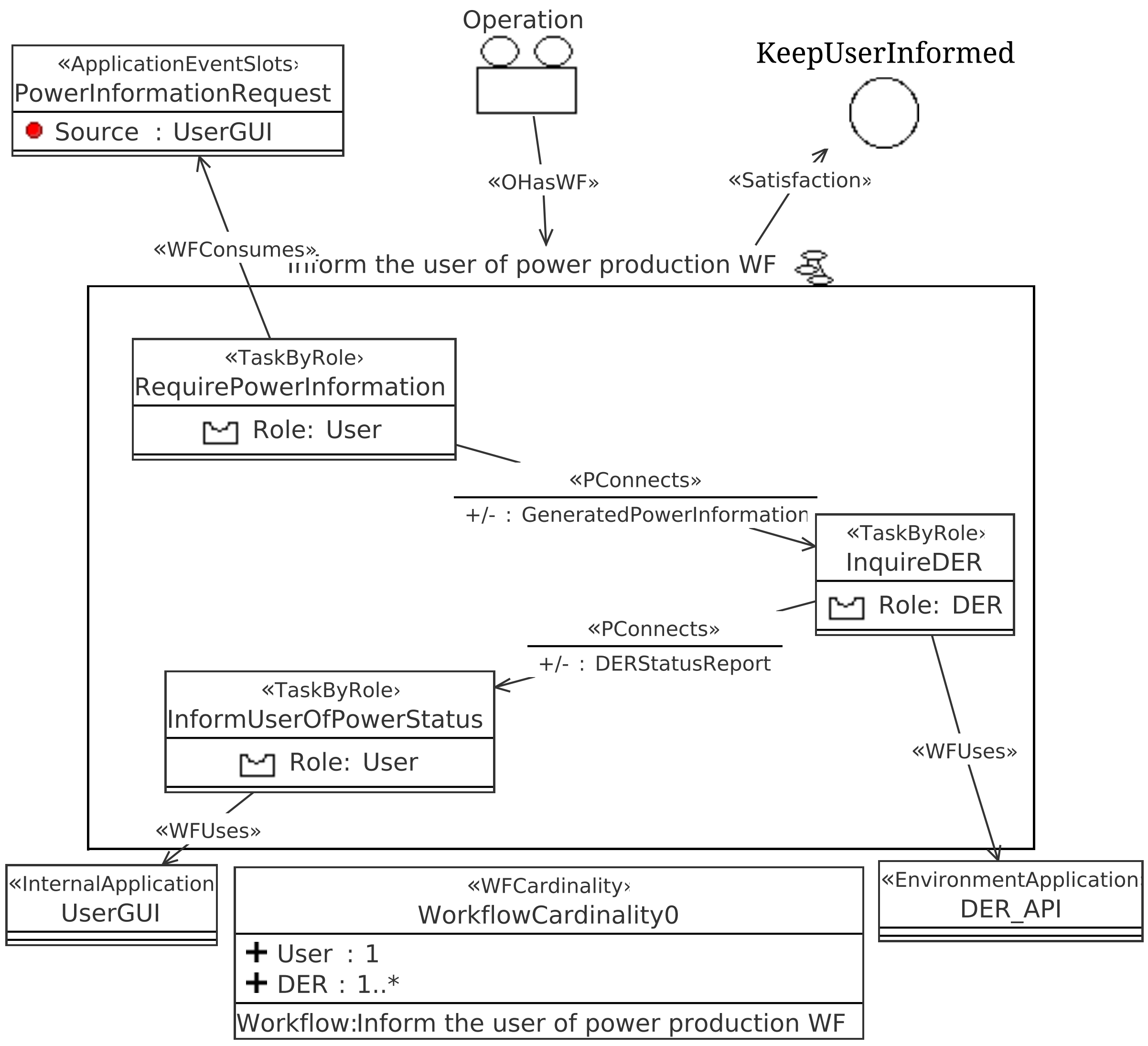
Figure 4. Blackout management task.



The third function implies the collaboration of several agents to inform the user of the current status of the grid. This is necessary to attain the goal keep the user informed identified in Figure 2. For this issue we use an INGENIAS workflow definition in Figure 5. The notation includes interconnection of tasks within the workflow pointing which element is produced by one task and which is consumed by the next task in the sequence. The "+/—" symbols mean that the referred piece of information is created, "+", by the previous task and consumed, "—", by the next task. Roles responsible of each task are declared together with the tasks.

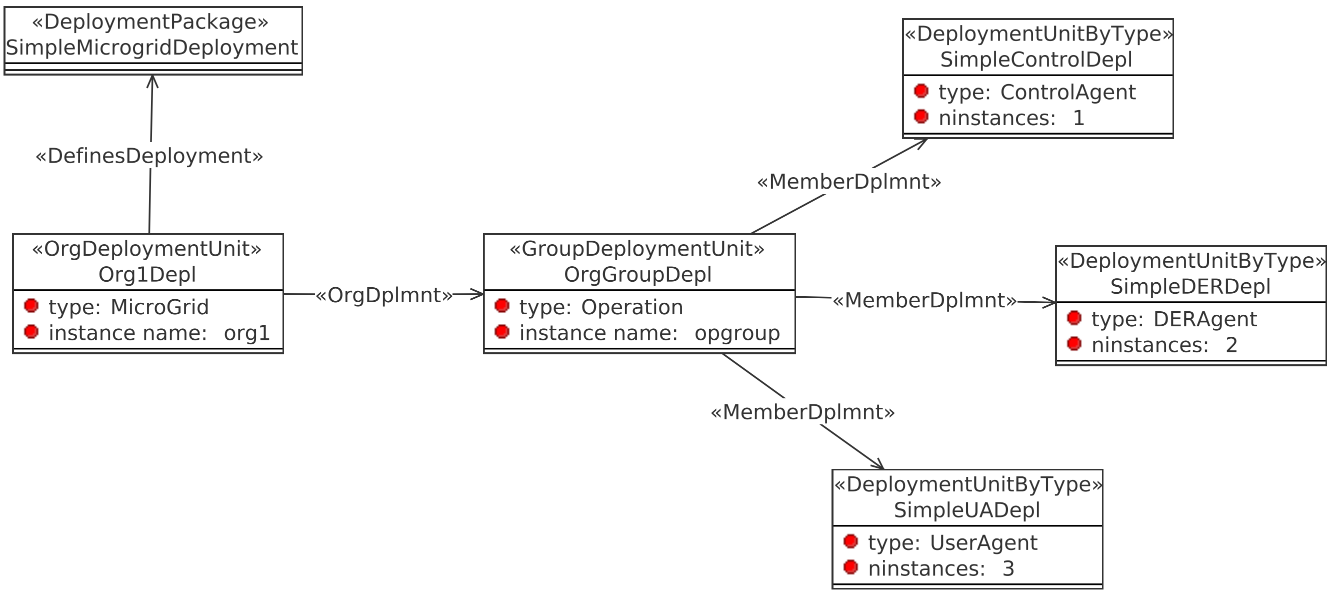
The workflow is initiated by the user for requesting more information (see task require power information and Figure 3 to identify the source of this event). This information is obtained from the DER (see task inquire DER). Once it is collected, the information is presented to the user through a GUI (see task inform user of power status). Furthermore the workflow has as scope the operation group type (see Figure 2). Since there can be several group types in the same Microgrid, this constrains the communication to agents belonging to the same group. The number of agents involved in this workflow is not fully determined until runtime. There is one user involved and at least one DER (see entity workflow cardinality). The default behavior implies that one user will ask all DERs within the same Operation group. The workflow pursues the goal keep user informed too. This goal is also associated by default to the different tasks that belong to this workflow.

Figure 5. Workflow (WF) for informing the user on the current status.



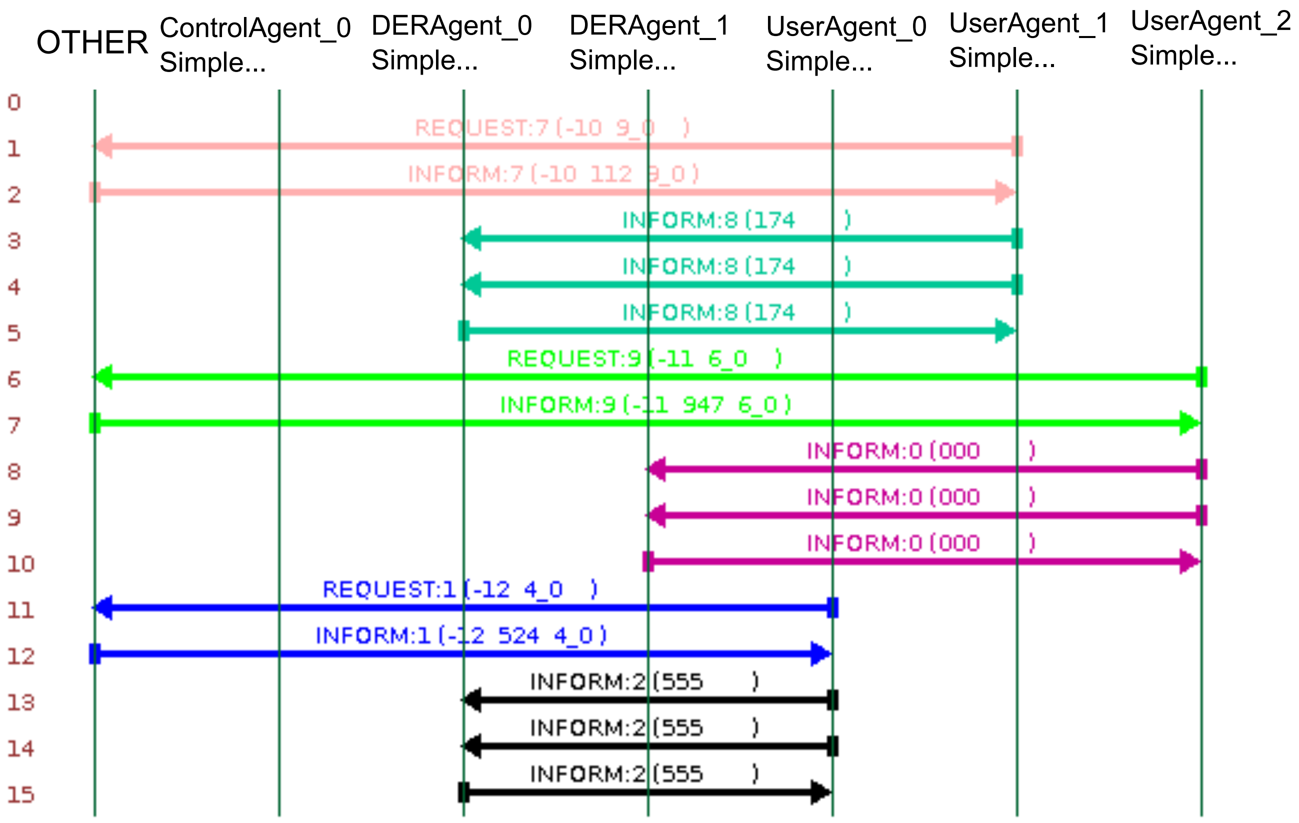
The deployment of the Microgrid was not considered in Pipattanasomporn [4]. Figure 6 shows how it would be done in the case of INGENIAS. The central element is a Deployment Package which represents an executable that launches a specific set of agent instances. The deployment represents the intended structure of the organization. In this case, there is one operation group with one Control Agent, two DER Agents and three User Agents.

Figure 6. Initial deployment description of the system.



Once designed, in INGENIAS diagrams are automatically processed and fully functional agent systems are produced. The generated system already implements the designed tasks and is capable of executing the expected protocols as shown in Figure 7. The development is managed using the Maven tool [33] which allows, with a minimum installation, to address the development in a efficient way.

Figure 7. JADE Sniffer showing how the communication among agents is taking place.



The execution and message interchange among generated agents is represented in Figure 7. Produced agents are JADE ones where JADE platform is the most widely used. Moreover, generated agents can reuse JADE tools like the sniffer. The sniffer intercepts all messages interchanged among agents and presents sequence diagrams such as those of Figure 7. The numbers on the left indicate the sequence order; the columns, the different senders/receivers; and the labels over the arrows, the message type. This initial version can be downloaded from the site [34] and github [35]. The figure shows how each user agent is querying DER agents in the system to obtain a status report, like in the original work. Looking at the message number, on the left, queries are messages 3, 4, 8, 9, 13 and 14; and answers are messages 5, 10 and 15. The apparent duplication of queries is due to the way inter-agent conversations are automatically coded in INGENIAS.

3.1. Evaluating I2DAPS

So far, a humble translation of Pipattanasomporn's work [4] has been introduced. The straight application of INGENIAS can bring some benefits with respect to the original IDAPS work which will be detailed below:

- This representation of the agent concept is more evident when introducing the deployment (see Figure 6). Reminding the principles introduced in Section 2.1, INGENIAS forces the developer to think how many instances of each agent should exist. In the next section, this example will go one step further and detail how each instance can be specified.
- There are blueprints. The representation of the agent elements in the system and their interdependencies are more explicit, which facilitates explaining others what the system does. For each agent, there is an account of the capabilities of each of them. Each capability is connected to a system goal. Most of the concepts in INGENIAS are connected to goals in one way or another. Since developers identify system goals, it is easy to justify the convenience of one element with respect the original specification requirement. It is a matter of finding how the element, like a task, is connected to the system goal, which is a form of requirement traceability along the design. Also, this brief analysis has led to changes in the conception of the system. Elements like a DB have not been considered as agents.
- Agents in the system can play more than one role. Figure 2 contains an agent playing two roles. This enables the agent to participate more in the system. In addition, it increases its capabilities and, potentially, the flexibility of the system. After all, the agent can play several roles at the same time. As it will be seen in the next section, this leads to situations where an agent can try to interact with itself because it is playing both the roles of client and server at the same time.
- The decoupling between what an agent is and what is not is clearer in INGENIAS than in the reviewed works from Section 2. The concept of application serves well to indicate how agents interact with non-agentified elements. Furthermore it prevents the overuse of agents when they are not required, for instance the case of converting a DB into an agent.

However, the example of I2DAPS has a serious flaw: it is disconnected from a Microgrid. Despite this fact, the example illustrates the potential of the approach and that initial assumptions made in

Pipattanasomporn [4] could have been different if an Agent Oriented Methodology, such as INGENIAS, had been applied.

Despite avoiding pitfalls cited in Section 2.1, it is not clear if the system, even refactored, is showing an intelligent behavior. The most promising part is the one dedicated to avoiding blackouts propagation. Though it is an important feature it could be discussed if, for an external observer, to avoid a blackout is a sign of intelligence.

4. Integrating Microgrid Simulation into I2DAPS

To address the issues raised in Section 3.1, this extension works on the agent principles from Section 2.1. It explores the effect of replication of single agents such as DER Agents and Control Agents. According to Hunhs and Singh's [25] test, the incorporation of new agents of the same kind should lead to an improvement of the functionality. Note that some replication was made in the previous section with little apparent impact.

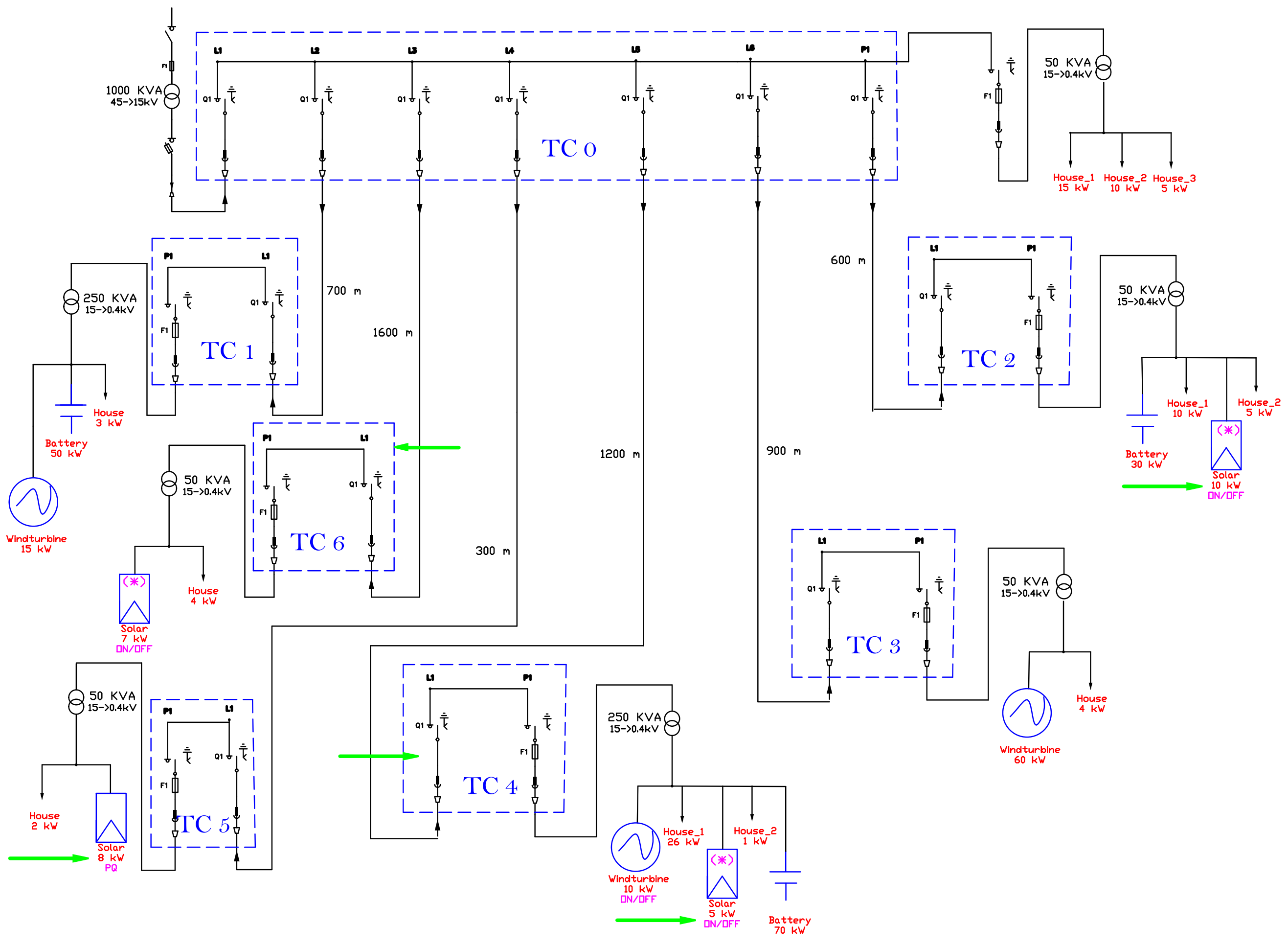
In this section we focus on Control Agents. Replicating them means that the system becomes more tolerant to system failures. However, it is not possible to add more control elements lightly and therefore this section will try to provide a preliminary solution to the mentioned problem.

Another improvement over the previous approach is the connection of the system to a simulator, just as the original work. Pipattanasomporn used Matlab while the second one will be based on GRIDLAB-D [36], a power grid simulator that also is able to simulate Microgrids. The simulator had to be modified to act as a real time simulator capable to receive orders in runtime and a new Java layer was incorporated. These improvements facilitated a greater connectivity to the INGENIAS system. The Microgrid to which agents will connect is introduced in Section 4.1. The design variations that will allow interconnectivity are presented in Section 4.2.

4.1. Case Study Microgrid

For experimental purposes, a radial Microgrid with distributed generation and batteries has been designed. Its structure is shown in Figure 8. The grid is made of six transformation centres ($TC_1...TC_6$) connected in medium voltage (15 kV) to a central one (TC_0); the latter receives the energy in medium voltage from the substation. A transformer of 1000 kV A at the entry of the substation receives the energy in high voltage (45 kV) from an external supplier and transforms it to a medium one. Furthermore, each TC_i represents a transformation centre where the energy is converted and driven from medium to low voltage lines and viceversa (maximum kV A supported by transformer and line distances are also shown in the figure).

Figure 8. Microgrid to be controlled by I2DAPS.



On the other hand, the Microgrid has several elements all connected to low voltage lines such as energy suppliers (wind and solar generators), batteries and consumers. Power consumption or generation is specified in each consumer and supplier. In case of batteries, maximum charging/discharging power (kW) is provided. Note that the generators can be automatically controlled by the agent system. Each generator specifies its control mode which can be ON/OFF or full control of P and Q.

The solar generators of TC_2, TC_4 and TC_6 are not permanently connected to the grid so that they would work as plug-in elements. The connections of those elements with the Microgrid is represented with "(*)" in Figure 8. Moreover, the transformation centres or elements where agents have been connected are signalized with a green arrow.

For an easier comprehension of the grid, its final elements are described in the following Table 1.

Table 1. Elements specification.

TC_i	Transformer	Name	Control	Consumption (kW)	Generation (kW)	Fase	Distance to transf. (m)
TC_0	50 kV A	House_1	-	15	-	ABC	222.5
		House_2	-	10	-	ABC	498.1
		House_3	-	1	-	ABC	424.5
TC_1	250 kV A	Windturbine	-	-	15	ABC	6
		Battery	PQ	50	50	ABC	20
		House	-	3	-	ABC	424.5
TC_2	50 kV A	Solar	ON/OFF	-	10	A	18
		Battery	PQ	30	30	ABC	19
		House_1	-	10	-	ABC	42.3
		House_2	-	5	-	ABC	51.5
TC_3	50 kV A	Windturbine	-	-	60	ABC	6
		House	-	4	-	ABC	110.5
TC_4	250 kV A	Solar	ON/OFF	-	5	A	80
		Windturbine	ON/OFF	-	60	ABC	38
		Battery	PQ	30	30	ABC	20
		House_1	-	26	-	ABC	61.5
		House_2	-	1	-	ABC	123
TC_5	50 kV A	Solar	PQ	-	8	A	34
		House	-	2	-	ABC	231
TC_6	50 kV A	Solar	ON/OFF	-	7	A	25.3
		House	-	4	-	ABC	145

4.2. Extending I2DAPS

The connection of agents to the Microgrid is done through the resources depicted in Figure 3. These resources are coded as mediators with the Microgrid using remote method invocation utilities. Then, the agents can deliver orders to the Microgrid in real time or obtain measurements from the sensors associated to it. This action implies few changes to the specification since it is only needed to add the new programming code to some specific elements.

In order to evaluate the role of the agents in the Microgrid from Section 4.1, the infrastructure introduced in the previous section is enhanced with new agent instances. The interactions among these new instances are analysed below.

Figure 9 introduces a new deployment where two groups are depicted whereas Figure 6 presented one. The main difference is that agents associated to one group will only see those ones from his same group if the developer wants it. Each group has a control agent instance and at least one DER agent. Group subgrid TC_2–TC_4 have DER agents connected to TC_2 (Solar 10 point in Figure 8) and TC_4 (Solar 5 point in Figure 8). Group subgrid TC_5 and TC_6 has less elements. Control agents are attached to TC_4 and TC_6. As a difference with Figure 6, this time agents are individually initialized.

The two groups determine two different scopes that will be more evident when explaining Figure 10. They limit the visibility of User and DER agents when performing certain operations.

Figure 9. New I2DAPS deployment for the Microgrid from Figure 8.

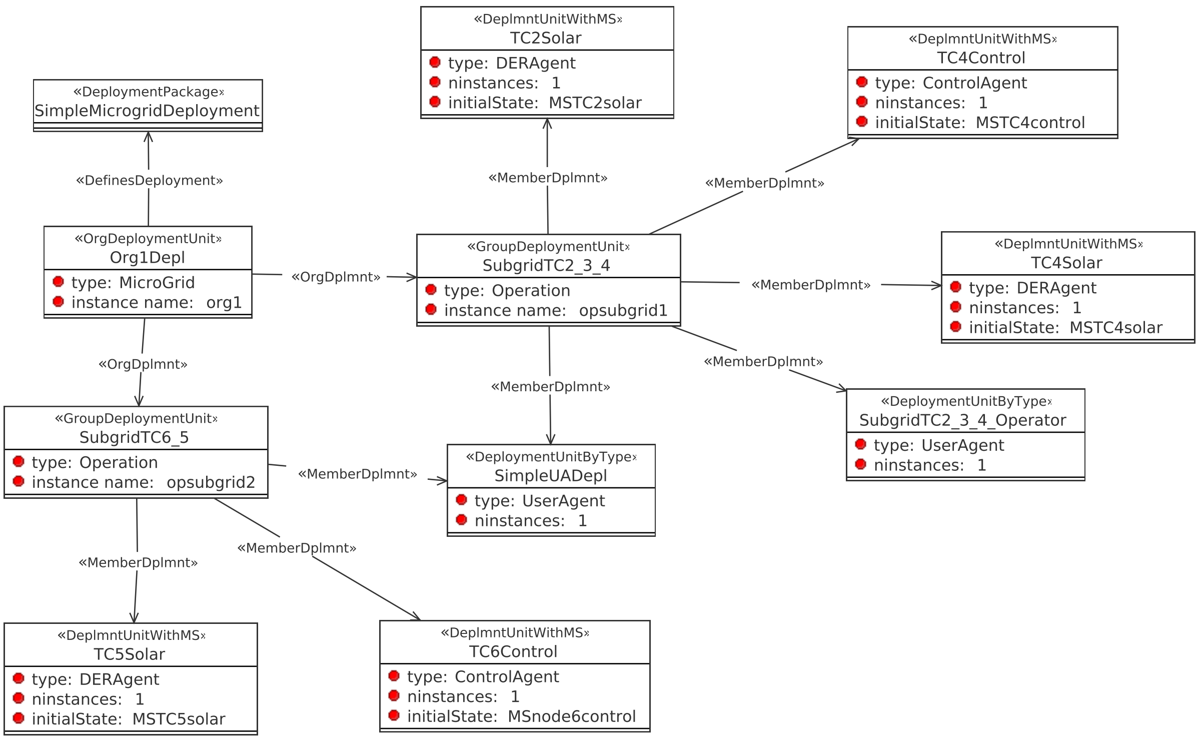
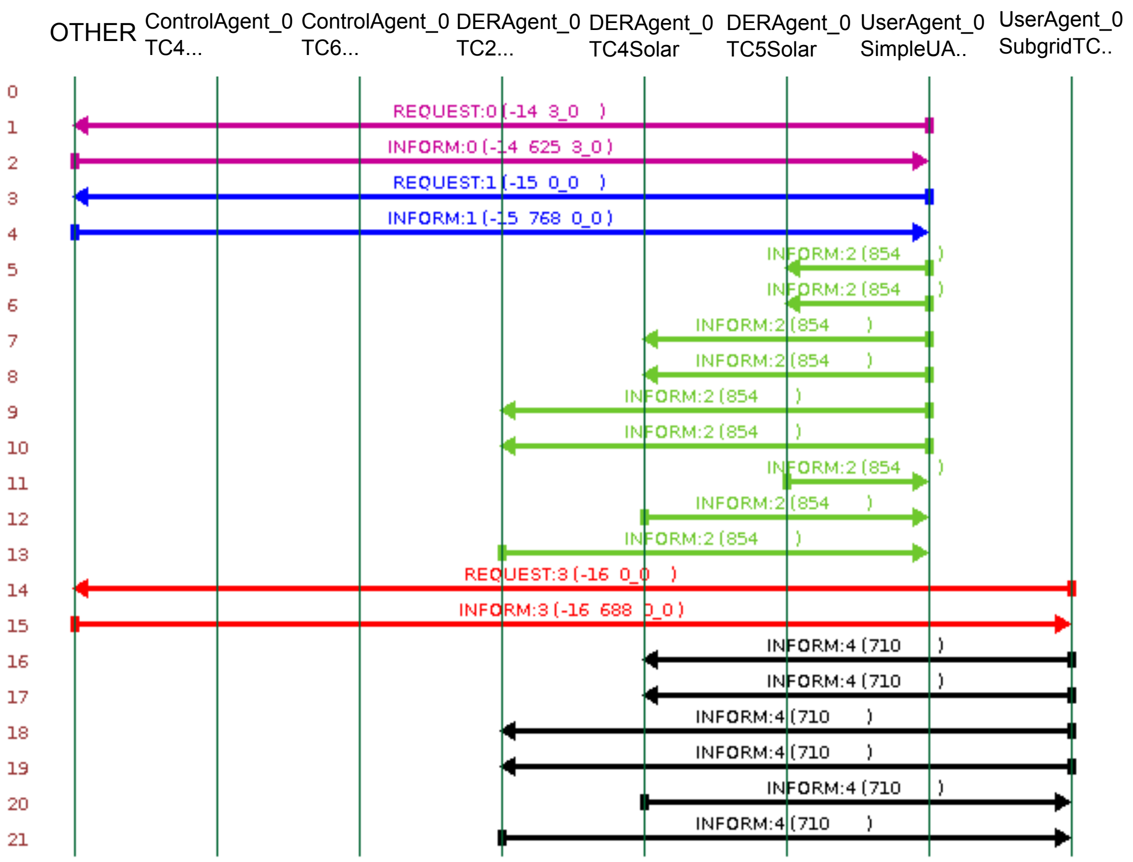
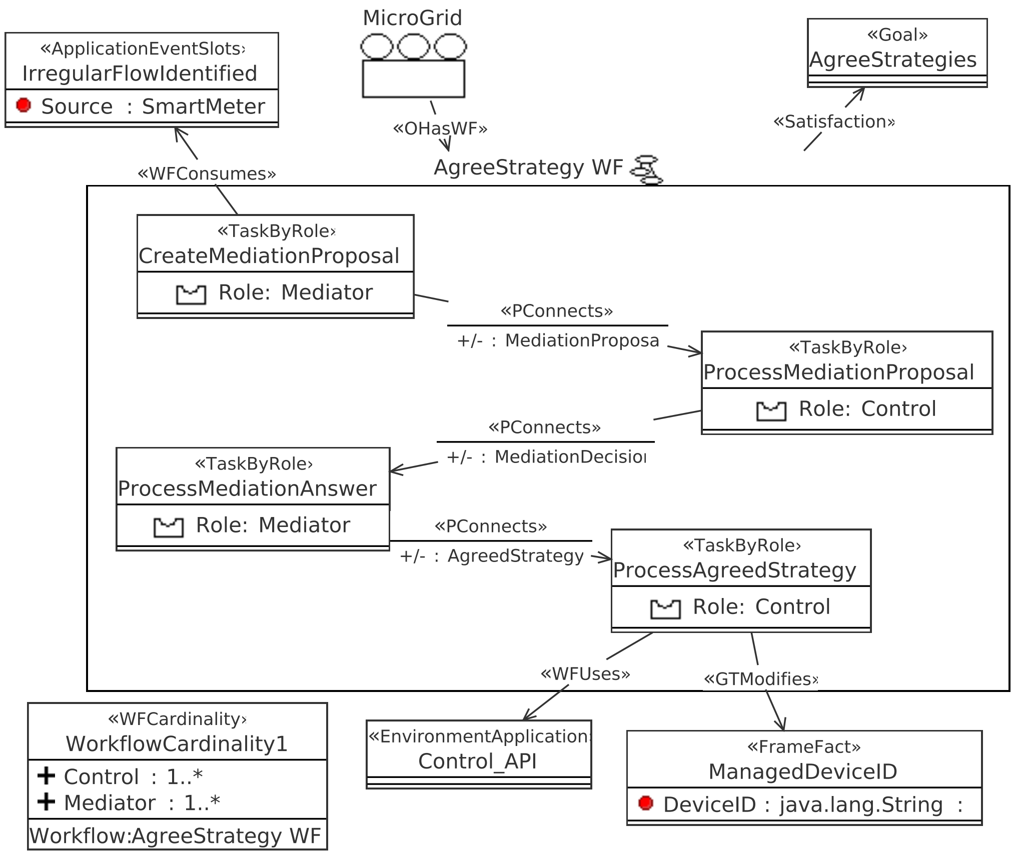

«DeploymentPackage» SimpleMicrogridDeployment
«DefinesDeployment»
«OrgDeploymentUnit» Org1Depl
type: MicroGrid
instance name: org1
«OrgDplmnt»
«GroupDeploymentUnit» SubgridTC6_5
type: Operation
instance name: opsubgrid2
«MemberDplmnt»
«DeplmntUnitWithMS» TC5Solar
type: DERAgent
ninstances: 1
initialState: MSTC5solar
«DeplmntUnitWithMS» TC6Control
type: ControlAgent
ninstances: 1
initialState: MSnode6control
«DeplmntUnitWithMS» TC2Solar
type: DERAgent
ninstances: 1
initialState: MSTC2solar
«GroupDeploymentUnit» SubgridTC2_3_4
type: Operation
instance name: opsubgrid1
«DeploymentUnitByType» SimpleUADepl
type: UserAgent
ninstances: 1
«DeplmntUnitWithMS» TC4Control
type: ControlAgent
ninstances: 1
initialState: MSTC4control
«DeplmntUnitWithMS» TC4Solar
type: DERAgent
ninstances: 1
initialState: MSTC4solar
«DeploymentUnitByType» SubgridTC2_3_4_Operator
type: UserAgent
ninstances: 1

Figure 10. Interaction among User Agents and DER Agents limited by the group membership as described in Figure 9.



Continuing the idea of replication of agents, Figure 11 does make an interesting point: what happens if there were more than one control agent in the grid? The system would become a chaos if the control is not properly partitioned. For this reason and to deal with the decentralization of control, we suggest a solution inspired in previous work [37]. The idea is to perform rounds of inquiries to the different controlling agents using a mediator role to reach an agreement.

Figure 11. New workflow for arranging agreements among control agents.

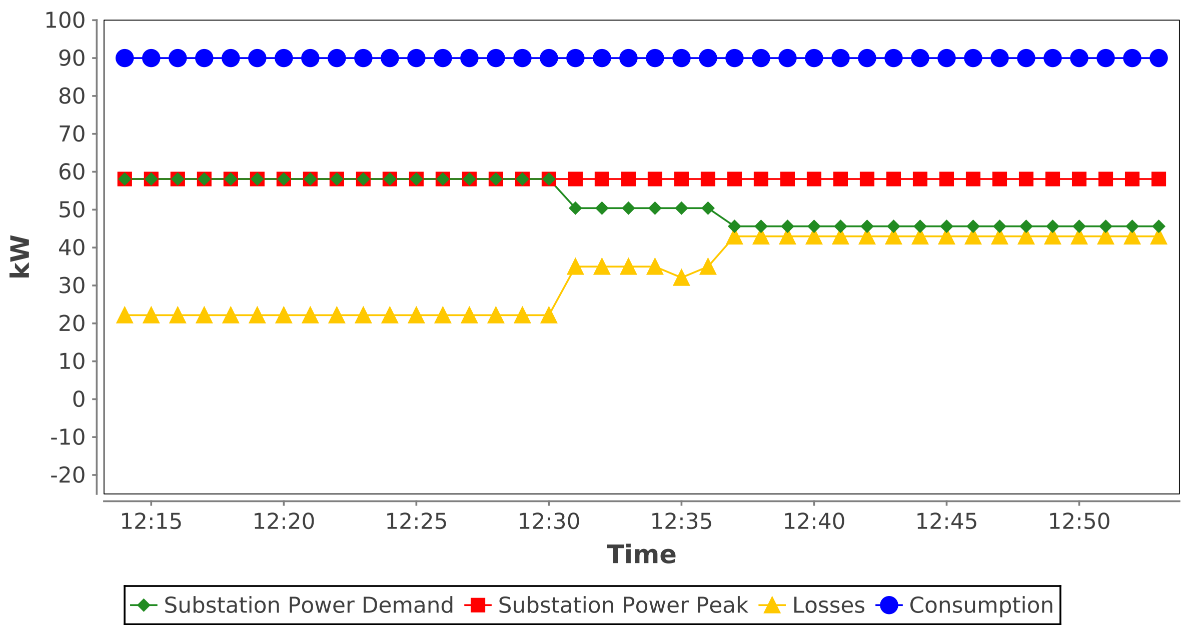


This agreement workflow is showed in Figure 11. Control agents will now also play the role Mediator and Control at the same time (see Figure 2). Hence, given two control agents, both can start collaborating with each other with any combination of roles. This new workflow will permit control agents to interchange strategies and enact a chosen one. The mechanism is independent of the number of agents. It also will be sensible to the number of control agent instances, which ought not to collide and give conflicting orders to DER agents. Hence, any two control agents can try in any order to achieve an agreement acting as peers. This time the scope of the workflow is the organization so that two control agents, even if they are assigned to different operation groups, will be able to perform this workflow. The cardinality of the workflow (see entity workflow cardinality in Figure 11) is also 1 to *N*. There is one mediator but at least one control agent.

This sensibility is explicit in Figure 10, which is a more elaborated version of Figure 7. The figure shows how the scope of each user agent is different when contacting DER agents. One of the agents is belonging to two groups so it can retrieve descriptions of DER agents belonging to both groups too. Since the other user agent has access to only one group, using properly the organizational concepts it is possible to limit the number of acquaintances of one agent.

In the implementation, the GRIDLAB-D based Microgrid simulation will be started into another process. DER agents, when starting, will plug to their corresponding element in the grid and request a POWER ON command to the element. As a result, the GRIDLAB-D will take into account the energy production of the DER. Figure 12 shows the Microgrid measurements taken at the main substation of the Microgrid. When DER agents are created and connected to the Microgrid the substation demand drops. It is coherent since incorporated agents are energy providers. Nevertheless, there is an increase of the distribution and transformation losses.

Figure 12. Measurements at the Substation when DER agents connect and enable the corresponding DER resources.



The influence of control elements can be seen only when there is any alarm. In this case, the alarm is the existence of an Irregular Flow Identified event generated by a Smart Meter (see Figure 3). Figure 13 shows such interaction. The interaction is arranged with all existing Control Agents in the Microgrid. Involved agents are Control Agent for TC_4 and Control Agent for TC_6. The second is acting both as leader and as subordinate. It works as leader when contacting other control agents and gathering proposals to solve the current issue. It works as a subordinate too when telling other control agents which strategy has been finally decided. In this case, the leader has to notify itself, since it plays both roles of mediator and control. Playing two roles in the same conversation means the same agent can play as message deliverer and receiver. This can be observed in lines 5, 6 and 7 of the Figure 13. In those cases, there is an arrow pointing at itself. Whatever the case is, the intention along the scenario is switching on associated devices in TC_6 and TC_4. This capability of adding elements to conversations while they are added to the system is one of the inherent capabilities of the agent technology.

The result of the interaction is presented in Figure 14. The agreed strategy consists in keeping generators on and discharging available batteries. Once again, the demand from the substation is reduced and there are some important losses in the system. The latter may be the case of a poorly designed Power Grid with less than adequate transformation centres. We use such elements to justify the activation of DER resources closer to the demand, specially when DER are not photovoltaic or wind based.

Figure 13. Workflow among Control Agents to deal with an extra demand.

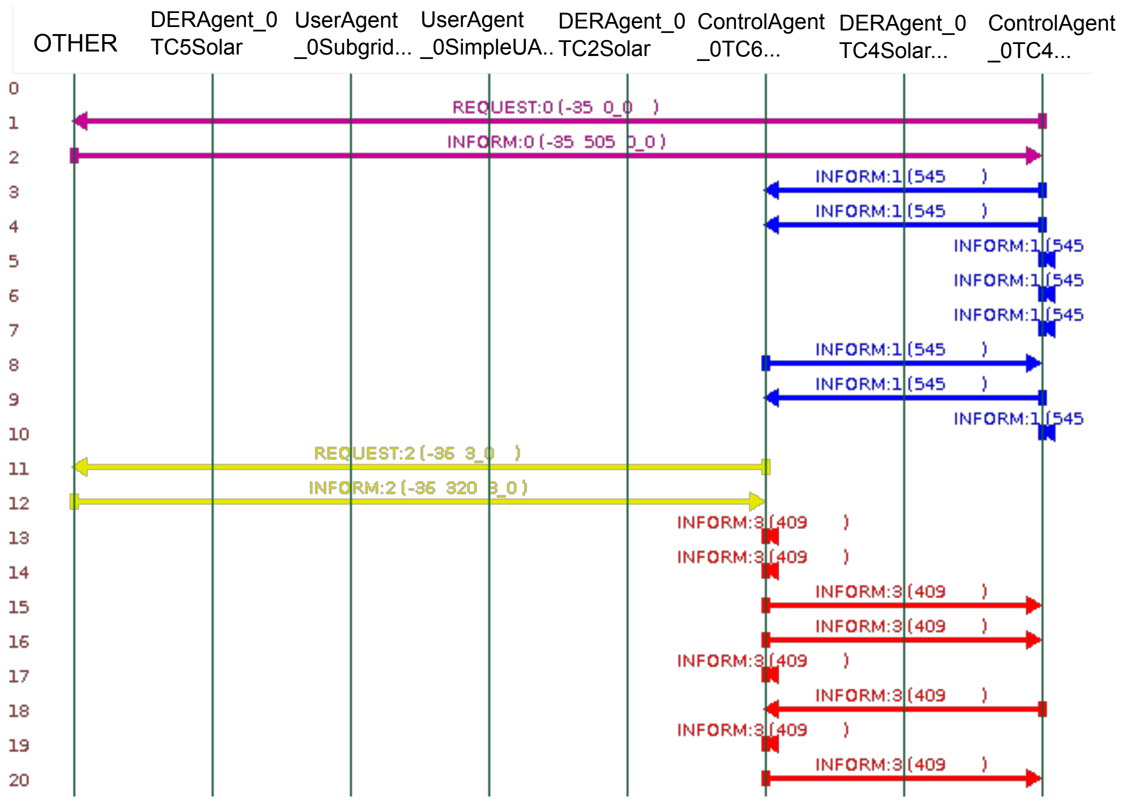
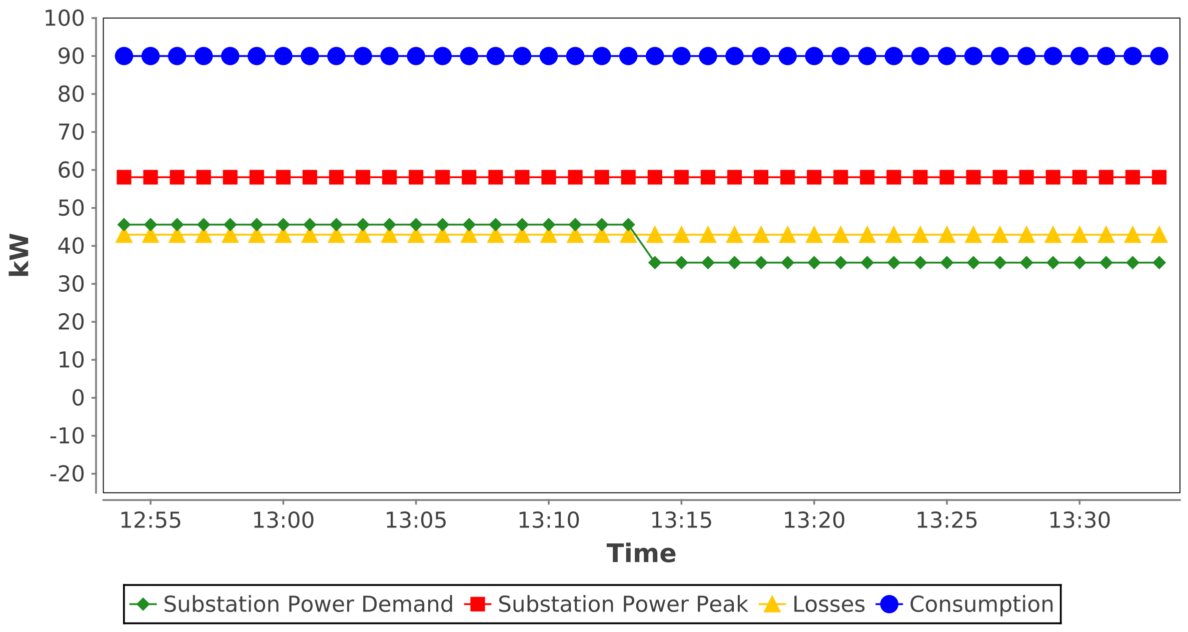


Figure 14. Demand from the substation when control is enabled.



4.3. Criticism of the Results

This new version is more complete than the previous one. Agents are connected to a power grid simulator and some charts could be elaborated with the results. Moreover, the visibility of the agents made possible to create distinct areas of influence while existing in the same Microgrid.

The intelligence of the resulting system is easier to observe than the last time but not absolutely complete yet. A real time operation of the simulated Microgrid may help for this task. If, during a long simulation, the resulting Microgrid behaviour showed improvements against the just turn everything on strategy, it would be a good evidence. Another hint could be to provoke failures in the network and check that the system is still capable of operating.

However, the agreement workflow (see Figure 13) allows the coexistence of different control agents in the same Microgrid, or even in the same group. As long as there is an agreement in the strategy and the willingness to comply with the agreements, the system should tolerate a number of control agents. In addition, this makes the system more fault tolerant. Control agents can fail and, still, the system could re-arrange remaining control agents and reassigning controlled devices if the agreement workflow were properly modified.

To conclude, thinking about the interaction among instances of the same agent was an educating experience. Agents can participate in the same conversation using different disguises, which can be an useful for combining capabilities in the same entity.

5. Conclusions

In this paper, we have reviewed some relevant research works about agents in Microgrids. Even though they apply agent technology, the resulting systems show some inconveniences: authors only consider one instance of each agent type; agents are not really having decision capabilities; there is distribution of components but little decentralization; agent interaction is a client-server one instead of peer-to-peer; and, most important one, there are no precise blueprints that explain what the MAS design is.

The role of agents in Smart Grids is not such evident for an agent researcher and, for this reason, we suggested in this work some guidelines to follow when designing an agent system to manage Microgrids. These guidelines are based on research trends pointed out in the agent research community.

In order to illustrate a practical example about how to carry out these recommendations, a specific highly cited work applying agent technology (IDAPS) to Micrograids [4] was evaluated and an alternative design was proposed (I2DAPS). The new design explores features of agent technology not addressed by the original authors. Among the changes performed over IDAPS, the most remarkable are:

1. Incorporation of the Organization concept to address the Microgrid subscription of agents. Also to illustrate how the different aspects of a Microgrid could be encapsulated within a single entity, the Microgrid Organization. Concretely, the paper includes the operation aspect, though not all power grid operation issues are considered.
2. Functionality reuse by means of roles. The original work simply assigned responsibilities to the agents whereas I2DAPS applies roles as mean to aggregate functions and associate them to different agents.
3. The not-everything-has-to-be-an-agent principle was applied and a clear distinction was made between what deserved to be considered an agent what did not. A DB in the original work was one agent and I2DAPS redefined it as non-agent element to be controlled by other agents.

4. Peer-to-peer philosophy was applied to realize cooperation between control elements. The example addressed several control elements which had to agree in a strategy to manage the network. The solution was an open one in the sense that new control agents could be incorporated to the agreement discussion.
5. In order to study the new system, agents were connected to a Power Grid simulator. Some charts with the tested Microgrid measurements and the dialogs among agents were included as well. Results showed how I2DAPS worked according to the previous description.

In addition, these benefits were represented in a way which is not usual in Microgrid papers that use agent technology. We applied INGENIAS to show how a blueprint of a MAS could be. INGENIAS and other agent oriented methodologies are solutions proposed by the agent research community to address an agent oriented development.

Even some benefits from the designed perspective have been identified in this work, there is still more experimentation to carry out such as simulating in a real environment or provoking different failures in the network in order to study how the system responds. Therefore, a performance study of the resulting solutions needs to be made. Though we show some experiments, more intensive ones are required to evaluate the impact of the new designs we propose. The performance of MAS systems has been already studied, but, as we justify in our study of existing works, we need to ensure before that the MAS technology is used as something more than a convenient middleware. An interesting experiment would be comparing IDAPS performance against I2DAPS, for instance.

Acknowledgments

To the project MIRED-CON IPT-2012-0611-120000, funded by the Ministry of Economy and Competitiveness. This project aims to bring new capabilities to conventional Microgrids in order to obtain improvements in billing and reducing energy distribution losses.

Author Contributions

The author Jorge Gomez-Sanz designed INGENIAS system for the experimentation, reviewed the state of the art and evaluated the individual contributions regarding the current agent research. The authors Sandra Garcia-Rodriguez and Nuria Cuartero-Soler carried out the integration of INGENIAS system within the GRIDLAB-D simulator power grid, designed the tested Microgrid, evaluated the experiments, and reviewed and corrected the document. Finally, Luis Hernandez-Callejo evaluated the Microgrid design, discussed the impact of agent research in terms of Microgrid control and advised about the state of the art.

Conflicts of Interest

The authors declare no conflict of interest.

References

1. Farhangi, H. The path of the smart grid. *IEEE Power Energy Mag.* **2010**, *8*, 18–28.

2. Sobe, A.; Elmenreich, W. Smart Microgrids: Overview and Outlook. In Proceedings of the "Smart Grid" Workshop of the ITG INFORMATIK 2012 Conference, Braunschweig, Germany, 16–21 September 2012.
3. Hassan, R.; Radman, G. Survey on Smart Grid. In Proceedings of the IEEE SoutheastCon 2010 (SoutheastCon), Concord, NC, USA, 18–21 March 2010; pp. 210–213.
4. Pipattanasomporn, M.; Feroze, H.; Rahman, S. Multi-Agent Systems in a Distributed Smart Grid: Design and Implementation. In Proceedings of the 2009 IEEE/PES Power Systems Conference and Exposition, PSCE '09, Seattle, WA, USA, 15–18 March 2009; pp. 1–8.
5. Amin, S.; Wollenberg, B. Toward a smart grid: Power delivery for the 21st century. *IEEE Power Energy Mag.* **2005**, *3*, 34–41.
6. Pavón, J.; Gómez-Sanz, J. Agent Oriented Software Engineering with INGENIAS. In *Multi-Agent Systems and Applications III*; Marík, V., Pechoucek, M., Müller, J., Eds.; Springer: Berlin/Heidelberg, Germany, 2003; Volume 2691, pp. 394–403.
7. Brooks, R.A. Intelligence without Reason. In *Computers and Thought, IJCAI-91*; Mylopoulos, J., Reiter, R., Eds.; Morgan Kaufmann: Burlington, MA, USA, 1991; pp. 569–595.
8. Brooks, R.A. A robust layered control system for a mobile robot. *IEEE J. Rob. Autom.* **1986**, *2*, 14–23.
9. McArthur, S.; Davidson, E.; Catterson, V.; Dimeas, A.; Hatziargyriou, N.; Ponci, F.; Funabashi, T. Multi-agent systems for power engineering applications—Part I: Concepts, approaches, and technical challenges. *IEEE Trans. Power Syst.* **2007**, *22*, 1743–1752.
10. McArthur, S.; Davidson, E.; Catterson, V.; Dimeas, A.; Hatziargyriou, N.; Ponci, F.; Funabashi, T. Multi-agent systems for power engineering applications—Part II: Technologies, standards, and tools for building multi-agent systems. *IEEE Trans. Power Syst.* **2007**, *22*, 1753–1759.
11. Bellifemine, F.; Poggi, A.; Rimassa, G. JADE: A FIPA2000 Compliant Agent Development Environment. In Proceedings of the Fifth International Conference on Autonomous Agents, Montreal, QC, Canada, 28 May–1 June 2001; pp. 216–217.
12. Davidson, E.; Catterson, V.; McArthur, S. The Role of Intelligent Systems in Delivering the Smart Grid. In Proceedings of the 2010 IEEE Power and Energy Society General Meeting, Minneapolis, MN, USA, 25–29 July 2010; pp. 1–6.
13. Nwana, H.S.; Ndumu, D.T.; Lee, L.C.; Collis, J.C. ZEUS: A toolkit for building distributed multiagent systems. *Appl. Artif. Intell.* **1999**, *13*, 129–185.
14. Xiao, Z.; Li, T.; Huang, M.; Shi, J.; Yang, J.; Yu, J.; Wu, W. Hierarchical MAS based control strategy for microgrid. *Energies* **2010**, *3*, 1622–1638.
15. Oyarzabal, J.; Jimeno, J.; Ruela, J.; Engler, A.; Hardt, C. Agent Based Micro Grid Management System. In Proceedings of the 2005 International Conference on Future Power Systems, Amsterdam, The Netherlands, 16–18 November 2005.
16. Jiang, Z. Agent-Based Control Framework for Distributed Energy Resources Microgrids. In Proceedings of the 2006 IEEE/WIC/ACM International Conference on Intelligent Agent Technology, IAT '06, Hong Kong, China, 18–22 December 2006; pp. 646–652.

17. Ramchurn, S.D.; Vytelingum, P.; Rogers, A.; Jennings, N. Agent-based Control for Decentralised Demand Side Management in the Smart Grid. In Proceedings of the 10th International Conference on Autonomous Agents and Multiagent Systems, International Foundation for Autonomous Agents and Multiagent Systems, AAMAS '11, Richland, SC, USA, 2011; Volume 1, pp. 5–12.
18. Kouluri, M.; Pandey, R. Intelligent Agent Based Micro Grid Control. In Proceedings of the 2011 2nd International Conference on Intelligent Agent and Multi-Agent Systems (IAMA), Chennai, India, 7–9 September 2011; pp. 62–66.
19. Dimeas, A.; Hatziargyriou, N. Operation of a multiagent system for microgrid control. *IEEE Trans. Power Syst.* **2005**, *20*, 1447–1455.
20. Dimeas, A.; Hatziargyriou, N. A Multiagent System for Microgrids. In Proceedings of the 2004 IEEE Power Engineering Society General Meeting, Denver, CO, USA, 6–10 June 2004; Volume 1, pp. 55–58.
21. Kumar Nunna, H.; Doolla, S. Multiagent-based distributed-energy-resource management for intelligent microgrids. *IEEE Trans. Ind. Electron.* **2013**, *60*, 1678–1687.
22. Colson, C.; Nehrir, M. Comprehensive real-time microgrid power management and control with distributed agents. *IEEE Trans. Smart Grid* **2013**, *4*, 617–627.
23. Logenthiran, T.; Srinivasan, D.; Khambadkone, A.; Aung, H.N. Multiagent system for real-time operation of a microgrid in real-time digital simulator. *IEEE Trans. Smart Grid* **2012**, *3*, 925–933.
24. Dimeas, A.; Hatziargyriou, N. Agent Based Control of Virtual Power Plants. In Proceedings of the International Conference on Intelligent Systems Applications to Power Systems, ISAP 2007, Niigata, Japan, 5–8 November 2007; pp. 1–6.
25. Huhns, M.N.; Singh, M.P. Agents on the web: The agent test. *IEEE Internet Comput.* **1997**, *1*, 78–79.
26. Caire, G.; Coulier, W.; Garijo, F.J.; Gómez-Sanz, J.J.; Pavón, J.; Leal, F.; Chainho, P.; Kearney, P.E.; Stark, J.; Evans, R.; *et al.* Agent Oriented Analysis Using Message/UML. In *Agent-Oriented Software Engineering II*; Wooldridge, M.J., Weiß, G., Ciancarini, P., Eds.; Springer: Berlin/Heidelberg, Germany, 2001; Volume 2222, pp. 119–135.
27. Webster, B.F. *Pitfalls of Object-Oriented Development*; M & T Books: New York, NY, USA, 1995.
28. Wooldridge, M.; Jennings, N.R. Pitfalls of Agent-Oriented Development. In Proceedings of the Second International Conference on Autonomous Agents, Minneapolis, MN, USA, 10–13 May 1998; pp. 385–391.
29. Gómez-Sanz, J.J. The Construction of Multi-agent Systems as an Engineering Discipline. In *Engineering Societies in the Agents World VII*; O'Hare, G.M.P., Ricci, A., O'Grady, M.J., Dikenelli, O., Eds.; Springer: Berlin/Heidelberg, Germany, 2006; Volume 4457, pp. 25–37.
30. Newell, A. Reflections on the knowledge level. *Artif. Intell.* **1993**, *59*, 31–38.
31. Russell, S.; Norvig, P. *Artificial Intelligence: A Modern Approach*; Prentice Hall: Upper Saddle River, NJ, USA, 2010.
32. INGENIAS Development Kit. Available online: http://ingenias.sf.net (accessed on 16 May 2014).
33. Maven-Apache Foundation. Available online: http://maven.apache.org (accessed on 16 May 2014).
34. Agents for Energy-GRASIA Reseach Group. Available online: http://grasia.fdi.ucm.es/energy (accessed on 16 May 2014).

35. GitHub. escalope/i2daps. Available online: https://github.com/escalope/i2daps (accessed on 16 May 2014).
36. GridLAB-D Simulation Software. Available online: http://www.gridlabd.org/ (accessed on 16 May 2014).
37. García-Magariño, I.; Gómez-Sanz, J.J.; Pérez-Agüera, J.R. A Multi-Agent Based Implementation of a Delphi Process. In Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS 2008, Estoril, Portugal, 12–16 May 2008; Volume 3, pp. 1543–1546.