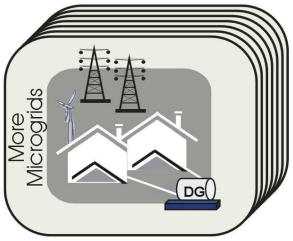


Advanced Architectures and Control Concepts for

MORE MICROGRIDS

Contract No: PL019864

WORK PACKAGE B

DB3: Decentralized control concepts

Final Version

February 2008

Document Information

Title: Decentralized control concepts

Date: 10-02-2008

Task(s):

Coordination:	Aris Dimeas	adimeas@power.ece.ntua.gr
Authors:	Nikos Hatziargyriou	nh@power.ece.ntua.gr
	Aris Dimeas	adimeas@power.ece.ntua.gr
	Spyros Hatzivasiliadis	spyrosxatz@gmail.com
	J. Jimeno	joseoyar@labein.es
	J. Oyarzabal	jjimeno@labein.es

Access:

__	**Project Consortium**
__	European Commission
X	PUBLIC

Status:

__	For Information
__	Draft Version
__	Final Version (internal document)
__	**Submission for Approval (deliverable)**
X	Final Version (deliverable, approved on…)

Table of Contents

1. Introduction

The objective of this WP is to develop control strategies based on centralized and fully distributed technologies and compare them to each other. The large opportunities provided by the wide application of next-generation ICT technologies, especially communications infrastructure will be investigated.
Scope of this part is the short presentation of the new control approach adopted within MORE MICROGRIDS project. This control approach may support several aspects of the Microgrid operation and is based mainly in the Multi-Agent System (MAS) technology. This control scheme introduces the idea that all the main decisions should be taken locally, being though in coordination with the other actors. The ability of coordination implies the usage of a high level language from the actors and consists of two main parts. The first part is that although the decision is local, it has taken into account the conversation or the negotiation between the actors. The second part is that a certain degree of high level coordination or monitoring is inevitable.

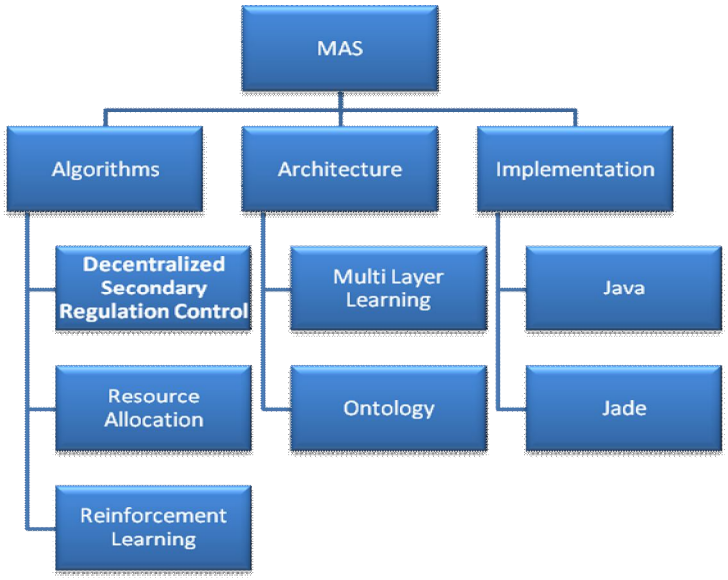


Figure 1 .The development of the MAS system.

Furthermore, one of the great advantages of the Multi-Agent Technology is its ability to deal with the complexity the Microgrid operation introduces. On the other hand, the centralized control approach reaches the limits of scalability and computation ability to deal with the complexity the Microgrids introduce. The ability to deal with the complexity leads also to the extended usage of all the operational advantages of the Microgrid. Since this is one of the main goals, a simple example can be given.

The work has three main parts as presented in figure 1.

More specific:

Algorithms:

Certain algorithms where presented that allow the actors to coordinate and converge in a Multi Agent environment. The algorithms presented in the deliverable take into

account the special characteristics the Microgrid operation introduces. We should take into account that the environment of the Microgrid is stochastic. For example, no one can predict accurately the actual consumption of a load. This requires the development of an algorithm that should take into account the unpredictability of the system.
Furthermore, there is another issue that forms the stochastic nature of the environment. This is the autonomy of the agents taking into account the fact that they collaborate. The autonomy is a fundamental characteristic of the MAS technology that is delivered from the distributed control and expresses the ability not to exchange all the information but only the necessary.
LABEIN presented algorithms aimed to efficiently operate a Microgrid. The objectives of a Microgrid Management System are to economically operate the different resources in the Microgrid while maintaining a proper quality of supply for the users connected to it. Other drivers such as minimizing emissions and performance optimization could be also taken into account. More specific **LABEIN** provided algorithm for **Decentralized Secondary Regulation Control**.
NTUA focused on providing algorithm that will allow the system to coordinate and perform the various operations. The algorithms are trying to support the system taking into account the stochastic nature of the environment. More specific NTUA provided a **resource allocation algorithm based on symmetric assignment** which allows the DG units and loads to Economically Dispatch the Load. The second algorithm is based on **Multi-Agent Reinforcement Learning** that allows the agents to create after negotiation a schedule for future actions.

Organisation

NTUA provided a system that includes several operations for the Microgrid. Therefore suggested a model where the MAS is organized in three levels. All the agents associated directly with the control of the production units or controllable loads belong to the ***Field Level***. These agents directly communicate and control a production unit or a load and may be organized in MAS according to physical constraints of the system. Each of these MAS has also an agent that has the responsibility to communicate with other higher level MASs in order to cooperate with them. These MAS belong to the ***Management Level***. Finally, these MASs may form larger MASs in order to participate in the ***Enterprise Level***. Furthermore advanced architecures that allow the operation of large amount of agents is introduced as presented in Figure 1.3.3.
Furthermore both **LABEIN** and **NTUA** worked in providing ontologies and using the extended capabilities of Jade in coordination of the MAS system.

Implementation:

Both **LABEIN** and **NTUA** implemented their systems in a common agent development platform called Jade. JADE (Java Agent Development Framework) is a software development framework aimed at developing multi-agent systems and applications conforming to FIPA standards for intelligent agents. It includes two main products: a FIPA-compliant agent platform and a package to develop Java agents. JADE has been fully coded in Java and an agent programmer, in order to exploit the framework, should code his/her agents in Java, following the implementation guidelines described in this programmer's guide. This guide supposes the reader to be

familiar with the FIPA standards1, at least with the *Agent Management* specifications (FIPA no. 23), the *Agent Communication Language,* and the *ACL Message Structure* (FIPA no. 61).

JADE is written in Java language and is made of various Java packages, giving application programmers both ready-made pieces of functionality and abstract interfaces for custom, application dependent tasks. Java was the programming language of choice because of its many attractive features, particularly geared towards object-oriented programming in distributed heterogeneous environments; some of these features are Object Serialization, Reflection API and Remote Method Invocation (RMI).

Next in Part A of this report the contribution of NTUA is presented and in Part B the contribution of Labein.

PART A. Contribution of ICCS

Scope of this chapter is the presentation of a new control approach that may support several aspects of the Microgrid operation and will focus mainly in the Multi-Agent System (MAS) technology. As it will be stated next, this control scheme introduces the idea that all the main decisions should be taken locally, being though in coordination with the other actors. The ability of coordination implies the usage of a high level language from the actors that consists of two main parts. The first part is that although the decision is local, it has taken into account the conversation or the negotiation between the actors. The second part is that a certain degree of high level coordination or monitoring is inevitable.

Furthermore, one of the great advantages of the Multi-Agent Technology, as it will be described, is its ability to deal with the complexity the Microgrid operation introduces. On the other hand, the centralized control approach reaches the limits of scalability and computation ability to deal with the complexity the Microgrids introduce. The ability to deal with the complexity leads also to the extended usage of all the operational advantages of the Microgrid. Since this is one of the main goals, a simple example can be given. Let's consider the main grid operation without the market services initially, for the sake of simplification. The key element of the main grid is the central SCADA system that collects measurements and data from the Network Substations as well as the Power Stations. After some state estimation, load flow and economic dispatch calculations the main SCADA sends set-points to the Power Stations.

This schema is far more complicated than it looks, since in the Power Station there are several specialized engineers supported by sophisticated software tools and local SCADA systems in order to control and monitor it. Similar is the status in the main substations.

If market operation is also included, at least two other concepts or services should be added in the previous example: the Power Producer Company and the Distribution Company. The Power Producer owns one or more Power Stations and sells energy to the Power Market. The Distribution Company buys energy from the Power Market and sells it to its consumers.

The conclusion is that the operation of a large grid is not only very complex but also that some principles of distributed control and local-distributed intelligence have already been applied if we take into account the several engineers (distributed intelligence) across the sub-stations or the power stations (distributed control).

The Microgrid, although being a simplified model of the main grid, a part of the main grid's complexity is still present in it, while some new features are also introduced. Therefore, at least the level of decentralization adopted in the main grid should be present. However, this is still not enough due to the true nature and the special requirements of the Microgrid. The owner of a Distributed Generator (DG) does not have as a primary goal to produce power in order to sell it or just cover the demand (non liberalized networks). The DG tries to meet several other tasks such as local heat demand satisfaction. Additionally, it is by no means designed to adopt the role of a large power station. Similar is the situation with the controllable loads.

Therefore, the scope of this chapter is to identify first the control services that the Microgrid operation requires and afterwards to describe how the Decentralized

Control Concept and more specifically the Multi Agent System Technology may provide a realistic solution for the required services.

2. The environment of a Microgrid

As it was mentioned in the introduction, in order to understand the basic functionalities of the Microgrid control, it is necessary to identify the main control services that the Microgrid needs. The main goal of this sub-chapter is to point out that processes like market participation, voltage support, load shedding or local heat demand satisfaction, are services that the control system should manipulate almost equally.

2.1 The Microgrid electrical operation as a process

Future Microgrids will comprise several types of production or storage units as well as controllable loads. The Microgrid, being a small model of the main grid, includes all the main grid's operations as well as its technical challenges in a simplified version. In this document every task or operation of the control system will be referred to as a service to the system, taking into account that the control system we are describing may control field level equipment (e.g. a switch) or even high level processes (e.g. Market participation).
Every control system provides services in order to manipulate the various processes of the system and to achieve its tasks or goals. A service could be the minimization of the total consumption within a group of units, the provision of reserve to the grid or the management of a battery bank. The various services have two main characteristics:

1. Local or global. The battery management is a local procedure but the consumption minimization of a group of units is not.
2. Contradictory. Consumption minimization is contradictory to reserve provision. The control system should decide the hierarchy of the services' satisfaction.

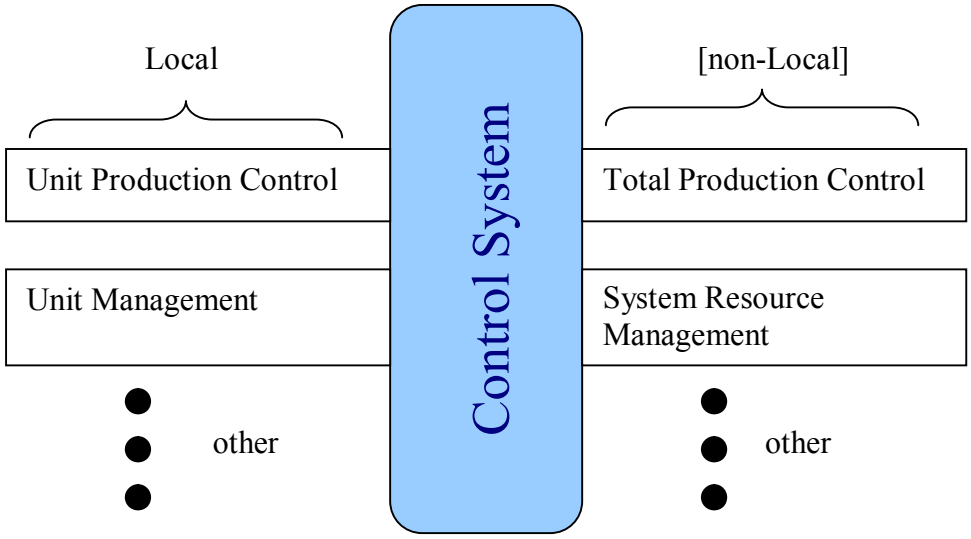


Figure 1. The services provided by the control system.

The two categories mentioned above, also reveal the main problem with Microgrid control: the complexity of the system. Figure 1 generally presents some of the several operations that should be adopted for the electrical control of the system. As an example we could consider two of these operations: the control of total production and the control of a specific unit. Although the similar control process in the main grid would have been described as central, it is actually not, since the central SCADA does not take into account local information (e.g. temperature of a boiler) in order to decide the set point for a specific Power Station. The central SCADA has knowledge of only two values, frequently updated by the Power Station personnel: the Maximum and the Minimum production capability. Thus, it sends a set point between these two values. Several people may however urge that the DG units are very simple and therefore simple rules and models can be inserted in the central control system. This is wrong if we consider the operation of hundreds of DGs (scalability) the system becomes significantly complex or even infeasible to solve (computational complexity).

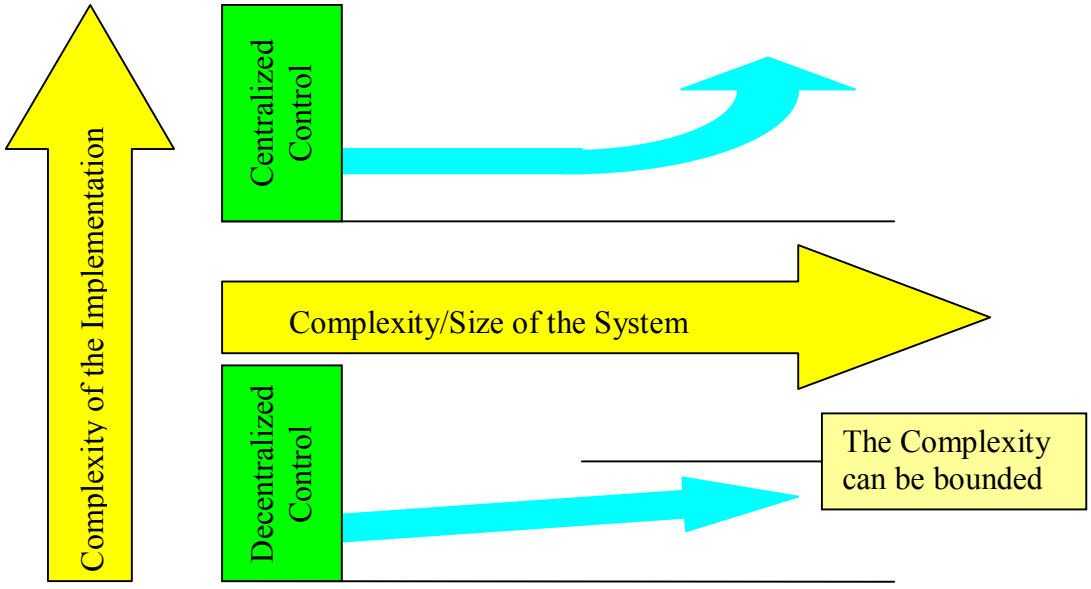


Figure 2. One of the main goals of the decentralized control is to deal with complexity.

The final conclusion about the control operation of the Microgrid is that the main challenge is not to deal with complex sub-systems (e.g. large power stations) but to deal with a complexity derived by the aggregation of several simple sub-systems as well as with several different services. Therefore the Decentralized Control approach actually emulates the operation of the larger system where several controls are distributed and thus is able to keep the complexity bounded as presented in Figure 2.

2.2 The Market Environment

One key aspect of the operation of Microgrids is the Market participation. This is one real challenging part and one of the main goals of the introduction of this technology. The owners of a distributed resource (DG) or a controllable load expect to have benefit from using them. Similar is the situation regarding all the entities that participate in the market.
However, one major issue should be clarified first. There are two steps in the establishment of the Energy Market environment. The first step is to define the market model and all the rules that should be respected by the participants. The second step is to implement the Market Model in the real system.

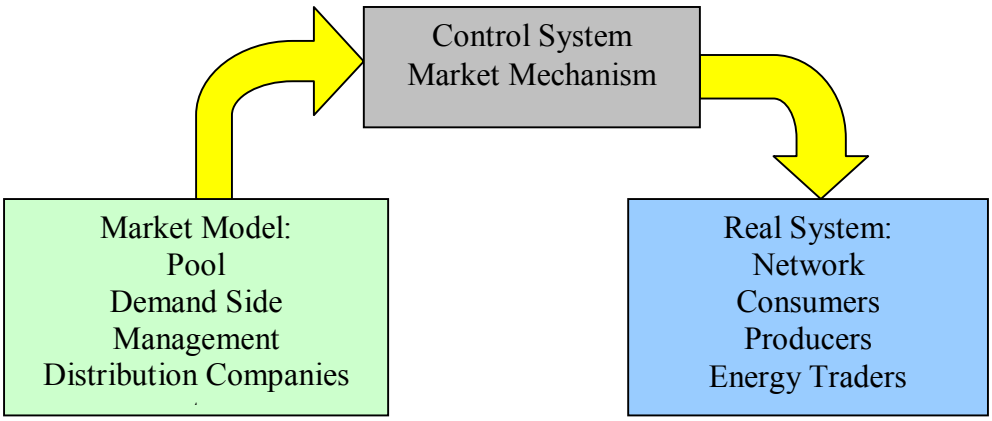


Figure 2. The decentralized control provides a solution to a given market model.

This is a critical issue since there might be confusion about the role of the control system, either centralized or decentralized. The role of the control system is to find the correct solution according to the existing Market Model. It should be stressed that the control scheme does not introduce a Market Model, but it is trying to find the optimal solution for a given model. Furthermore, depending on the capabilities of the control system, the ESCos may implement more complex Market Models as well as provide Services to the clients. In other words, a sophisticated control system may provide tools to the ESCos to support more complex services for his customers.
Another comment is that the Control System and the Market Model should identify the special characteristics of the real world in each case. This means that the complex Market Mechanism of a main grid focused on large production units and big consumers cannot be applied in the Microgrids. Although it is not the scope of this document to analyze the issues regarding the market operation, some key elements should be considered in the operation of the control system. One of these key elements is that e.g. Market participation is a tool for increasing the benefit of the system as shown in Figure 3.
The owners of a DG or a controllable load do not have as a primary motivation the income that could be gained by the optimal participation of their unit in the Energy Market. Instead, their goal is to optimize their operation (as a whole Microgrid or as single units), taking into account that there are also other needs such as heat demand.

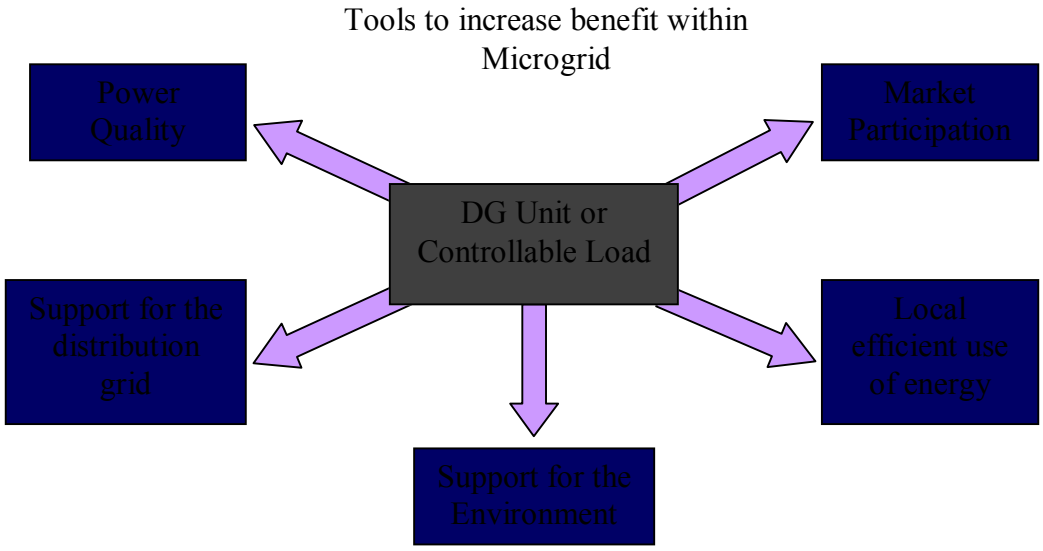


Figure 3. Market participation among others is a tool to increase the benefit of a Microgrid.

The control system should be able to identify the special needs in each case and to use the Market Services in the most beneficial way. The balance between the local needs and the Market participation should be found in each case separately, as presented in Figure 4.

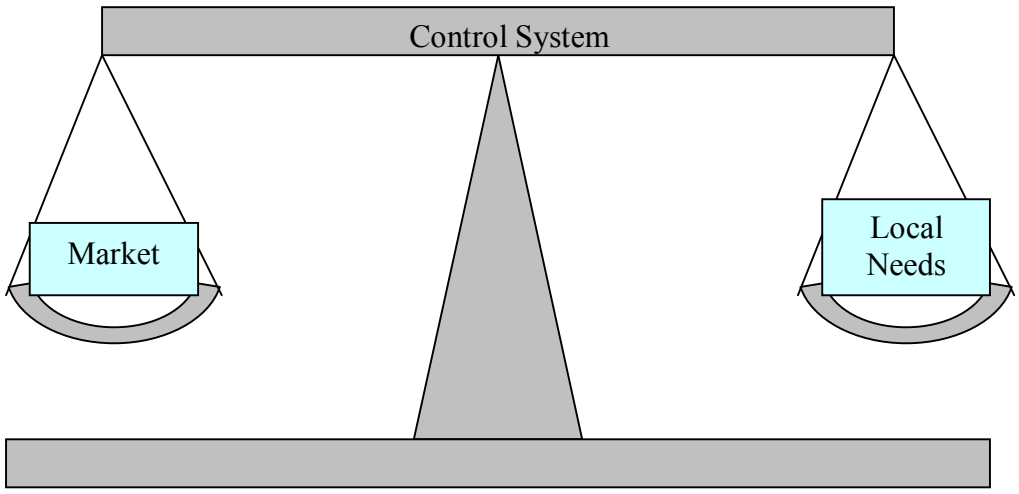


Figure 4. The control system should be able to balance between market participation and local needs

In this document an exact market model cannot be described since none exists. A main Market scheme can be given though. This scheme is presented in the next figure:

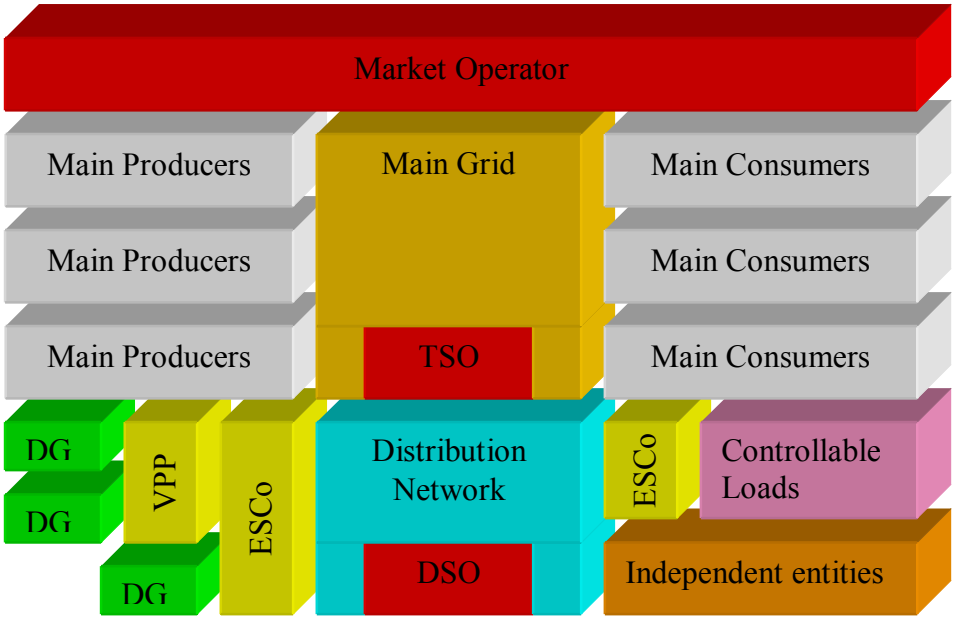


Figure 5. The generic scheme of the Market Model

In the generic Market model adopted here, Microgrids belong to the Distribution Network Level and are participating in the Market actively via ESCo companies. There might also be other entities (Independent Entities) that occasionally should be taken into account. These could be uncontrollable loads or production units that do not belong to a Microgrid.
The key element here is that the DG units may interact directly with the ESCo or form a Virtual Power Plant (VPP). The scope is not to investigate what is best but to provide a control scheme flexible enough to include complex forms.

3. Multi Agent System Theory

This chapter introduces a quite new theory in the area, not only for Microgrids control but also for Power System control or management. The theory is called Multi Agent System (MAS) technology. MAS is actually the evolution of the distributed control scheme as well as the Object-Oriented design. From these ancestors it keeps two main characteristics:

1. A problem is solved by splitting it into several smaller sub-problems
2. The abstraction of information, meaning the data, information, processes and knowledge that are not required in higher levels of decision processes are kept private.

Beyond these two points another significant characteristic is introduced. This is the local intelligence. The intelligence, although it is a quite fuzzy concept, is expressed mainly by the ability to take decisions autonomously as well as by the ability of high level communication with other agents. Next in this sub chapter, a more detailed description of the agent theory is presented, starting with the definition of the agent itself.

3.1 What is an agent?

There are several books and papers that may have presented the basic theory of agents. Scope of this text is not to provide an introduction of agent theory but to present the agents from the point of view of a Microgrid control system. For different types of applications, even within the power system area the concept, agents may be different, more abstract or fuzzy.
The main element of a MAS is the agent. An agent is a physical entity that acts in the environment or a virtual one without physical existence. In our case, an agent with physical entity is a microsource or a controllable load and a virtual one a piece of software that coordinates the agents. The cooperation of several agents forms a society called MAS. The basic element for this cooperation is the high level communication ability among the agents.
An agent is capable of acting in the environment, meaning that the agent is capable to change its environment by its actions. For example, an agent that controls a storage unit and decides to store energy, rather than to inject it, alters the decision and the behavior of other agents.
Agents communicate with each other and this is part of their capability of acting in the environment. For example, agents controlling microsources communicate with the Market Operator and the other agents, in order to negotiate in the internal Microgrid market.
Agents have a certain level of autonomy. This means that they can take decisions driven by a set of tendencies without a central controller or commander. The autonomy of each agent is related to its resources, e.g. the available fuel, in case of a production unit.
Another significant characteristic of agents is that they have partial or none at all representation of the environment. Each agent only knows the state of the unit or the load it controls; it can be however informed via conversation with the other agents about the status of the neighbouring system.

Finally, an agent has a certain behavior and tends to satisfy certain objectives using its resources, skills and services. One skill could be the ability to produce or store energy and a service could be to sell power in a market. The way that the agent uses its resources, skills and services defines its behavior. As a consequence, the behavior of each agent is formed by its goals. An agent that controls a battery system aiming to provide uninterruptible supply to a load has a different behavior than a similar battery system, whose goal is to maximize profits by participating in the energy market. The concept of the behavior is a significant part of the agent technology and is further analyzed in the next section.

3.2 Agent vs. Intelligent Agent

Several issues regarding the agent technology are confusing and the main cause for this situation is the lack of a strict description of what an agent is. Any entity or device that has one or more of the characteristics mentioned in the previous section can be considered an agent.

For example, an under-frequency relay may be an agent:

- it has partial representation of the environment
- reacts autonomously according to its goals
- possesses skills

According to the literature this type of entity can be considered as a reactive agent who is just responding to stimuli from the environment. Another category indentified in the literature is the cognitive agent. This type of agents have the ability to perceive the environment, in other words they have intelligence. However, still the definition of an intelligent agent is quite fuzzy since there is no formal definition about what intelligence is.
Therefore, the authors adopt in this text some main attributes that describe an intelligent agent for Microgrid control. These are:

1. Memory in order to acquire knowledge of the environment. The memory is one fundamental element of the intelligence and the ability to **learn**.
2. Ability to perceive the environment. The internal modeling and representation of the environment should be detailed enough in order to support the decision making process.
3. Ability to take decisions according to his memory and the status of the environment and not just to react.
4. Ability for high level communication. The agents should have the ability to exchange knowledge and use the communication as a tool to proceed with complex coordinated actions.

It is not the purpose of this section to further analyze what is artificial intelligent and what is not. However, it is critical to distinguish the architecture presented here from MAS based on reactive agents.

3.3 MAS and Agent Communication

Two or more intelligent agents that cooperate form a Multi-Agent system. It is beyond of the scope of this chapter to provide a detailed description on how the MAS is formed and what are the internal rules, however some critical points should be provided.
The main point is the cooperation between the agents. This is a basic element of the MAS structure. There is no need for the development of this type of control if the participants within the Microgrid do not adopt the principle that through the collaboration they will increase not only the overall but also their own benefit. Furthermore, as described in the previous section, this collaboration due to the numerous local or non local services is very complex. On this account, one of the main goals of this document is to provide a model/method on how to cope with this complexity and keep it bounded as presented in Figure 3.
The next point is that this cooperation is infeasible without the presence of a high level communication language. The communication among the agents should be distinguished from the common communication systems which are based on the theory of Shannon. In this section we are not concerned about how a message with 255 characters will be encoded, modulated and transmitted from point A to point B. The core of the agent communication is the content of the message. The content of the message is based on the formal Agent Communication Language [ACL] which among others has two main characteristics:

1. formal structure like the syntax in human language
2. ontology which is similar to the vocabulary of the human language

This kind of a high level language is necessary in order to support the fundamental operations of the agent, which are the collaboration and the intelligence.

4. The coordinated decentralized control system based on MAS

4.1 Introduction

The main simple principle regarding the use of MAS in the control of Microgrids is clearly illustrated in Figure 6.

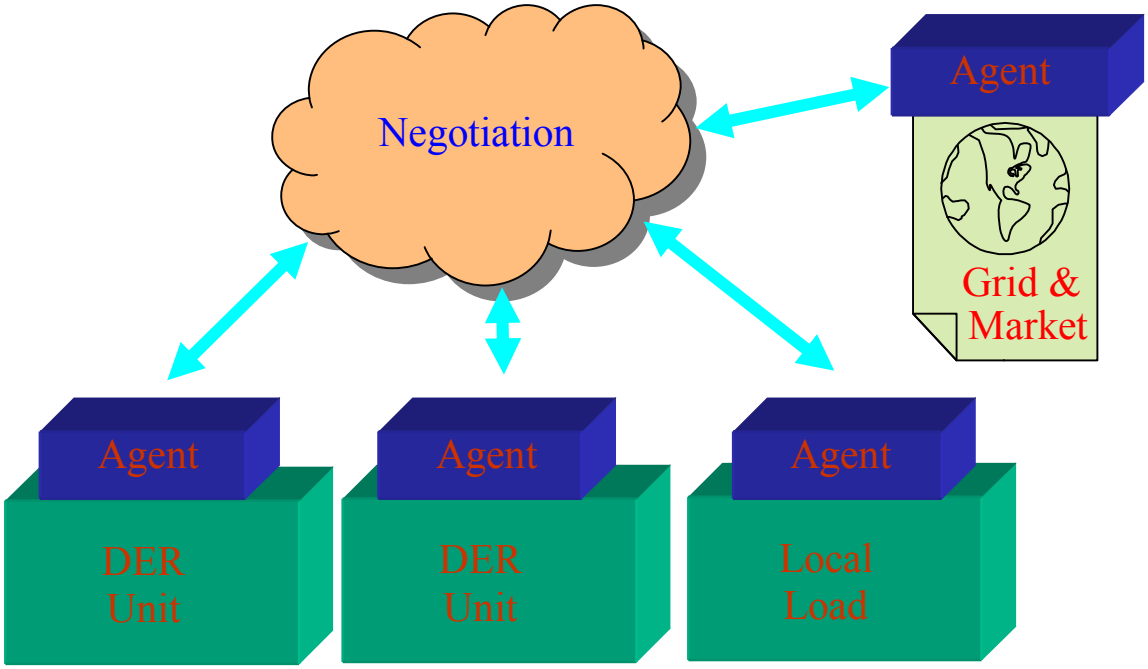


Figure 6 The MAS control scheme.

According to this scheme, an agent is attached on every DER unit and on every controllable load. Furthermore, an agent who interacts with the ESCo or the DNO (distribution network operator) announces prices or commands for load shedding etc. "Coordinated decentralised control", as the title of this section denotes, is the core of this implementation. The agents take decisions according to guidelines given set by the ESCo (e.g. price schedules) or the DNO (e.g. commands to shed load). By no means, are the agents able to act freely according to their own will.
This architecture has certain issues that need to be clarified:

1. How are the various services organised within the MAS?
2. What kind of algorithms can be used for the various services and how do they interact with each other ? How can intelligence be used?
3. How can such an architecture be applied with several actors?
4. How can such an architecture be applied in a real field?

In the following sections, the authors will describe the solutions they adopted in their implementations.

4.2 Service organization

This question refers to a critical issue regarding the MAS operation. As mentioned before, the operation of the control system incorporates several services that may be local or global. The key issue in the design of the system is to strictly identify the services and classify them.
The authors adopted a very simple but very effective model for this type of classification called Multi-Layer Learning, introduced by [1] for the control of a team of robots who participate in a soccer match.
Based on this approach, each agent controls one unit of the Microgrid, for example a battery bank, a wind turbine or a controllable load and has the following characteristics:

- A capability for certain actions, such as production of energy or operation of a switch.
- A variety of behaviours, according to which decides what the next action should be.
- Some resources such as diesel fuel level or battery state of charge, depending on the type of the unit.

According to the new architecture, the behaviour of the agent is categorised into three levels as shown in table 1.
In this table, behaviours for communication purposes or other strict software engineering tasks are omitted in order to simplify the description. The main idea behind this classification is that the behaviours are grouped depending on the effect on the environment.

Level	Agents	Behavior	Example
1	1	Single	Battery management
2	Many	Multi Agent	Resource Allocation
3	Many/All	Team	Market Participation

Table 1. Control levels of the MAS behaviours.

The first level includes all the actions and decisions that are necessary to control and manage **locally** the unit. For example, in the battery management operation, if the agent detects a low state of charge (SOC), he decides to stop the injection of energy to the grid. This is an action that was decided locally without asking for permission from other agents. On the other hand, the decision of starting the charging operation is not local and for this reason the decision should be taken on a higher level.
The second level includes simple tasks that should be completed by more than one agent. For example, after the clearing of the market, the Microgrid receives an order to inject certain kWh in the grid and possibly also feed the local loads. The decision for the level of production for each unit can be taken after a small internal auction as described next.

1.Peter Stone and Manuela Veloso. Team-Partitioned, Opaque-Transition Reinforcement Learning. In Minoru Asada and Hiroaki Kitano, editors, RoboCup-98: Robot Soccer World Cup II, Springer Verlag, Berlin, 1999. Also in Proceedings of the Third International Conference on Autonomous Agents, 1999

The highest level includes all the services that refer to the ultimate goal of the system, which in our case is the optimisation of the Microgrid benefit. An example for this level is the market participation.

The operation of the various levels can be clearly illustrated in the example of Figure 7, which refers to the Market Participation. In the first step, the MAS, after negotiating with the ESCo, should receive a schedule for Power Production and Power Consumption that also includes prices. The negotiation and the decisions regarding the participation of the Microgrid in the Market belong to the higher level of control which constitutes the team behaviour.

Afterwards, the agents should decide how to realise the schedule. Two schedules are created, one for production and one for consumption. In this way, it is obvious that there are now two sub problems that do not include all the agents. The production schedule includes only the DG units and the consumption schedule only the controllable loads. Focusing on the production schedule, the DG units should decide on how to share fairly the requested production. After the negotiation, each DG has a separate schedule and the control process moves to the lowest level, i.e. is the local level.

In the local level, every DG unit should accomplish its schedule taking into account its special characteristics and status. It is obvious that the special characteristics as well as the status information were taken into account in the previous negotiation. For example, a battery unit would not make any bid if it has no sufficient kWh stored.

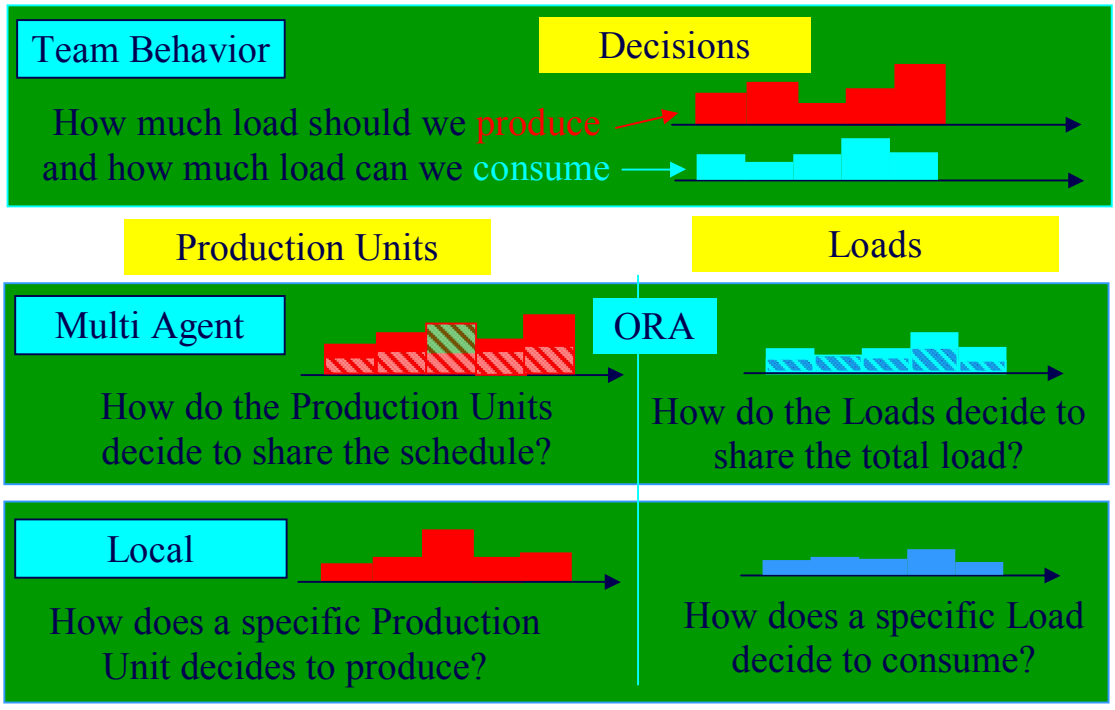


Figure 7. The Multi Layer Learning

It is obvious that several questions arise about the collaboration of the agents according to the Multi-Layer Learning approach and definitely the first refers to the type of algorithms used for the negotiation.

5. Algorithms and Services

The question of the previous chapter was referring to the type of algorithm used within the various levels of the MAS. This means that we should describe how the agents reach to a conclusion. It is obvious that the full list of these algorithms cannot be enumerated since they are infinite, if we take into account that each algorithm may have several variations.
However a classification may be done to the algorithms:

1. Deterministic
2. Stochastic

We should take into account that the environment of the Microgrid is stochastic. For example, no one can predict accurately the actual consumption of a load. This requires the development of an algorithm that should take into account the unpredictability of the system.
Furthermore, there is another issue that forms the stochastic nature of the environment. This is the autonomy of the agents taking into account the fact that they collaborate. The autonomy is a fundamental characteristic of the MAS technology that is delivered from the distributed control and expresses the ability not to exchange all the information but only the necessary. It is not the scope of this document to analyse this characteristic, since there is a lot of discussion on this topic among the researchers [1,2], however an example may be given: in the example described in the previous section regarding the Multi-Layer Learning, the DG starts to produce taking into account that the other units will try to fulfil their obligation to the schedule. But the DG agent is not sure if they will actually fulfil their obligations and what they will finally do. This kind of behaviour of the agent is necessary in order to avoid mass data exchange and extremely complex algorithms that require extensive processing power.
The other type of algorithms is the deterministic. Several problems do not have the stochastic parameter. For example in the previous section the DG unit had to share the production schedule. The production schedule, the prices as well as the internal cost of the various units are strictly defined and so the solution should be unique. For this type of problems the agents are called to find this unique solution.
Finally, a general comment concerns the time effort needed in order to find a solution. In a real system, it is generally accepted that a good solution found in limited time is acceptable and more preferable than an optimal solution that takes, however, extended time within the time horizon of the problem.

Next two examples of algorithms will be presented

1 Multi-Agent Systems for Power Engineering Applications—Part I: Concepts, Approaches, and Technical Challenges McArthur, S.D.J.; Davidson, E.M.; Catterson, V.M.; Dimeas, A.L.; Hatziargyriou, N.D.; Ponci, F.; Funabashi, T.; Power Systems, IEEE Transactions on Volume 22, Issue 4, Nov. 2007 Page(s):1743 – 1752 Digital Object Identifier 10.1109/TPWRS.2007.908471

2 Multi-Agent Systems for Power Engineering Applications—Part II: Technologies, Standards, and Tools for Building Multi-agent Systems McArthur, S.D.J.; Davidson, E.M.; Catterson, V.M.; Dimeas, A.L.; Hatziargyriou, N.D.; Ponci, F.; Funabashi, T.; Power Systems, IEEE Transactions on Volume 22, Issue 4, Nov. 2007 Page(s):1753 - 1759Digital Object Identifier 10.1109/TPWRS.2007.908472

5.1 Symmetric assignment

The core of the algorithm applied is based on an auction algorithm for the solution of the symmetric assignment problem. This method provides maximization of the internal benefit of the system. The symmetric assignment problem is formulated as follows:

Consider n persons and n objects that should be matched. There is a ***benefit*** a_{ij} for matching person i with object j. In the presented application, the benefit for each person is his revenues for obtaining object j, i.e. an agreement for producing a certain amount of energy. The main target is to assign the persons to objects and to maximize the ***total benefit***:

$$\sum_{i=1}^{n} a_{ij} \quad (1)$$

The ***price*** p is an algorithmic variable that is formed by the bids of all persons and so expresses the global desire. The prices of all objects form the price vector. These prices should not be confused with the market prices. Furthermore, the difference between the benefit and the price is the ***actual value*** of an object for a specific person. The actual value for a specific object is different for two persons, since it is related to the benefit. At the beginning of the iterations, the price vector is zero and so the actual value is equal to the benefit, although variations of the proposed methods use initial non- zero values for faster convergence.

In order to clarify the above terms, we consider an example with two objects and two persons that belong to a larger set of persons and objects. The first person has a benefit vector $\{a_{11},a_{12}\}=\{10, 9\}$, the second one has benefits $\{a_{21},a_{22}\}=\{7, 10\}$. Taking into account only the benefits, the first person has higher benefit for the first object and the second person for the second object. If we assume a price vector p={1, 8}for the two objects, the actual values for the two persons are {9, 1} and {6,2}.Both players desire the first object more, since both have greater actual value for it than for the second, however the second person has greater benefit for the second object, than for the first. It can be said that the benefit represents local information for each person and the price vector global information for the whole system. The price for an object increases until at most one person wants it. Increasing the price of an object is an indication that there is another person that desires this object, too.

The auction algorithm used calculates the price vector p, in order to satisfy the ε-complementary slackness condition suggested in [1,2]. The steps are described next:

At the beginning of each iteration, the ε-complementary slackness condition is checked for all pairs (i,j_i) of the assignment. The j_i is the object j that person i wants to be assigned to. So the formulation of this condition is:

$$a_{ij_i} - p_{j_i} \geq \max_{j \in A(i)} \left\{ a_{ij} - p_j \right\} - \varepsilon \quad (2)$$

1 D. P. Bertsekas and D. A. Castanon, "A Forward/Reverse Auction Algorithm for Asymmetric Assignments Problems", Computational Optimization and Applications 1992, Vol. 1, pp. 277-297.

2 D. P Bertsekas, “Auction Algorithms for Network Flow Problems: A Tutorial Introduction”, Computational Optimization and Applications, 1992, Vol. 1, pp. 7-66.

A(i) is the set of objects that can be matched with person i. This inequality has two parts: α_{ij}-p_j is the actual value of object j for person i , as described before. The right part refers to the object that gives maximum value to person i minus ε. ε is a positive scalar, added in the bid of each object, in order to avoid possible infinite iterations in case two or more objects provide maximum benefit to the same person, as will be explained later.

If all persons are assigned to objects, the algorithm terminates. Otherwise, a nonempty subset I of persons i that are unassigned is formed. Similarly, the nonempty subset P (j) is formed by the available objects. The following two steps are performed only for persons that belong to I.
The first step is the bidding phase, where each person finds an object j which provides maximal value and this is:

$$j_i \in \max_{j \in A(i)} \{a_{ij} - p_j\} \qquad (3)$$

Following this, the person computes a bidding increment

$$\gamma_i = u_i - w_i + \varepsilon \qquad (4)$$

u_i is the best object value

$$u_i = \max_{j \in A(i)} \{a_{ij} - p_j\} \qquad (5)$$

and w_i the second best object value

$$w_i = \max_{j \in A(i), j \neq j_i} \{a_{ij} - p_j\} \qquad (6)$$

According to the previous equations, the bidding increment is based on the two best objects for every person. The price of an object rises, if there are two or more bids for it and the price increment is the larger bidding increment between the bids. It is obvious that, if the scalar ε=0 and the benefits for the first and the second best object are the same, then γ_i=0 and this leads the algorithm to infinite iterations. The ε scalar ensures that the minimum increment for the bids is γ_i=ε.
The next phase is the assignment phase, where each object j selected as best object by the nonempty subset P (j) of persons in I, determines the highest bidder

$$i_j = \max_{i \in P(j)} \{\gamma_i\} \qquad (7)$$

Object j raises its prices by the highest bidding increment $\max_{i \in P(j)} \{\gamma_i\}$, and gets assigned to the highest bidder i_j. The person that was assigned to j at the beginning of the iteration, if any, becomes unassigned.
The algorithm iterates until all persons have an object assigned. It is proven [11] that the algorithm converges to the optimal solution, as long as there is one. The maximum number of iterations is

$$\frac{\max_{(i,j)} |a_{ij}|}{\varepsilon} \qquad (8)$$

and the algorithm terminates in finite number of iterations if

$$\varepsilon < \frac{1}{n} \qquad (9)$$

The above algorithm is further explained by considering three persons and three objects. All bidders have zero benefit for the third object and the benefit for the rest of the objects is a constant C>0. So the benefits for the objects are $a_{ij}=0$ for i=1,2,3 and j=3 and the other benefits $a_{ij}=C>0$. The initial prices are considered zero. For this example, the first four iterations of the algorithm are presented in Table 1:

umber of eration	Prices	Pairs	Bidder/ Object	Increment
1	0, 0, 0	(1,1)(2,1)(3,1)	3/1	ε
2	ε, 0, 0	(1,2)(2,2)(3,2)	2/2	2ε
3	ε, 2ε, 0	(1,1)(2,1)(3,1)	1/1	2ε
4	3ε, 2ε, 0	(1,2)(2,2)(3,2)	3/2	2ε

Table 1. The results of the auction algorithm for the first four iterations.

The second column of the table called "Prices" shows the price of the three objects at the beginning of each iteration. The column called "Pairs" shows the pairs (persons, object) that are assigned at the end of the iteration. The fourth column called "Bidder/ Object" shows which person was the bidder at the end of the negotiation and for which object a bid is made. Since we have only three objects and three persons, there can be only one bidder in each iteration. The last column shows the bidding increment at the end of the iteration.
In the first iteration, all persons desire object 1, so there are bids for this object. It should be mentioned that, in case the bids are equal, some selections are random or based on rules. The initial selection could be the second object or two persons could bid for the first object and the other for the second. The selected strategy does not affect the convergence of the algorithm according to [11]. According to the benefits, all the persons in this iteration have the same increment bid, the value of the bid is ε and the winning bidder is the last person. In the second iteration the bid increment for all persons is 2ε because:

$$\gamma_i=(u_i-w_i)+\varepsilon=\varepsilon+\varepsilon=2\,\varepsilon \qquad (10)$$

In this example the need of ε is clearly illustrated, because otherwise no price would increase. All persons desire object 1 or 2, since the benefit is C>0 for both of them. In the last iteration the prices of 1 or 2 object will be greater than C, object 3 receives a bid and this is the end of the algorithm.

5.2 Multi Agent Reinforcement Learning

The algorithm adopted is based on Multi Agent Reinforcement Learning (RL). Reinforcement Learning is a family of iterative algorithms that allows the agent to learn a behavior through trial and error. The well known Q-Learning algorithm is selected with its main characteristic being that each agent runs his own Q-Learning for the part of the environment it perceives, having however as a target to optimize the overall Microgrid performance.
Q-learning is a Reinforcement Learning algorithm that does not need a model of its environment and can be used on-line. Q-learning algorithms operate by estimating the values of state-action pairs. The value Q(s,a) is defined as the expected discounted sum of future payoffs obtained by taking action α from state s and following an

optimal policy thereafter. Once these values have been learned, the optimal action from any state is the one with the highest Q-value. After being initialized, Q-values are estimated on the basis of experience, as follows:

- From the current state *s*, select an action *a*. This will bring an immediate payoff *r*, and will lead to a next state *s'*.
- Update Q(*s,a*) based upon this experience as follows: $Q(s,a) = (1-k)Q(s,a) + k(r + \gamma \max_{a'} Q(s',a')$ where *k* is the learning rate and 0 < γ < 1 is the discount factor.

This algorithm is guaranteed to converge to the correct Q-values with probability one, if the environment is non stochastic and depends on the current state and the action taken in it. This exploration strategy does not specify which action to select at each step. In practice, a method for action exploration, called the *Boltzmann distribution* strategy, is usually chosen, which will ensure sufficient exploration, while still favouring actions with higher value estimates.
The main drawback of Q-learning is that it cannot operate in a stochastic environment, like the typical Microgrid environment. This means that when an agent selects an action and the environment is stochastic, then the next state of the system is unknown. For example, if an agent selects to allow a load to operate, the consequence of this action is unknown, since the total demand is unpredictable.
Alternative learning algorithms have been investigated by the authors, including the Nash-Q learning which is a general sum MAS reinforcement algorithm for stochastic environment. However execution times were prohibitive, because the Nash-Q learning requires that the Q table includes as a parameter the actions of the other agents. For systems like Microgrids, the Q table becomes huge, requesting a large number of episodes for training. Other approaches propose to forecast the decisions of other agent, however this cannot be easily done in our case.

The authors propose an alternative approach based on Q-learning that takes into account the stochastic environment and the size of the problem. The core of the algorithm is based on the idea that every agent runs a separate Q learning algorithm for itself, perceiving just a state environment variable that expresses the overall state of the system. Thus, the stochastic and complex environment is considered by adding a new variable, called state transition. This variable expresses the most possible transitions of the system and describes the possible actions of the other agents of the system.
In order to better understand this, let's consider that we want to have the full description of the system explicitly.
This means that we should include the selected actions of all agents in the Q table.

$$Q(s,\alpha 1,\alpha 2,\alpha 3,\ldots \alpha n) \quad (1)$$

a1,a2..an is the selected action of agent 1, agent 2 .. agent N. In this way the Q table becomes huge while the environment is still stochastic, since we still cannot predict the result of switching a load on the system state. The approach proposed in the paper

replaces all actions with one single variable called transition that represents the final result to the environment of all actions of the agents.

$$Q(s,\alpha 1,\alpha 2,\alpha 3,\ldots \alpha n) \rightarrow Q(s,\alpha 1,\mathbf{tr}) \qquad (2)$$

As an example, we consider the following system with three states.

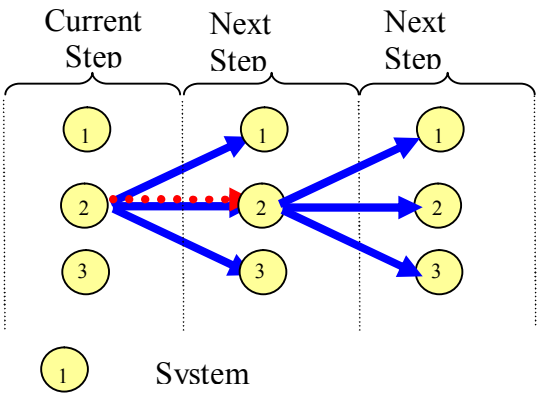


Fig. 2. The three states of the example system.

The states describe the power flow between the Microgrid and the upstream main grid. In state 1 the grid provides power to the Microgrid, state 2 represents the reverse power flow and state 3 represents zero power exchange. This is the system variable. Let's assume that the agent is a diesel generator and the possible actions are to produce power or not. In each state the selected action will lead to a different or the same state in the next time step depending on the action of all the agents. Three transition variables are introduced to describe the most possible transitions of the system. The three transitions are:

1. **System goes to State 1**
2. **System goes to State 2**
3. **System goes to State 3**

The agent learns the value of the three following cases:

1. "The agent is in State 2, selects action A and the **System goes to State 1**".
2. "The agent is in State 2, selects action A and the **System remains in State 2**".
3. The agent is in State2, selects action A and the **System goes to State 3**".

In this way, the agent learns what the value of its action is in all possible future states of the system and the system is not anymore stochastic. This is because we are interested in deciding if the system sends or receives energy and not what a single load should do.
It should be noted that the total number of possible transitions of the system are too many; however we consider only the ones which are most probable. For example, if a diesel unit has no fuel it is not expected that in the next state will have fuel.

During the learning phase the algorithm explores the various states and actions according to the fundamental rules of Q- learning.
The next question is how the agent uses this final knowledge that is in the Q table. Consider that the system is in the time step t and in state X. In this state all the agents together will select which transition is the best by using the following formula.

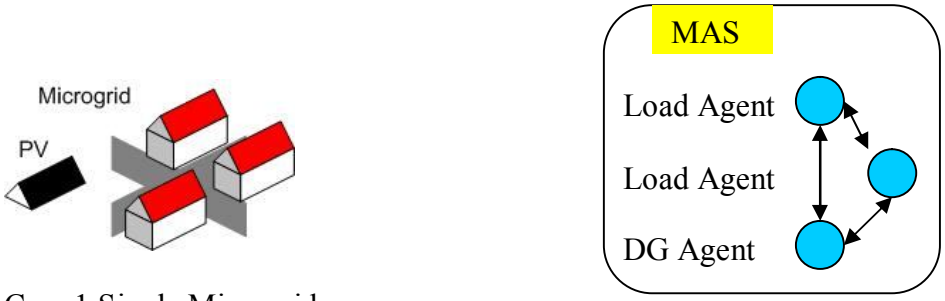
$$Selected \text{ transition} = \underset{tr=1,2,3}{\text{argmax}}\left(\sum \max_{\alpha}(Q(s,a,tr))\right) \quad (3)$$

This formula says that for each transition, each agent selects the action that provides the maximum Q value and they all add these values for each transition. The selected transition is the one with the higher value.

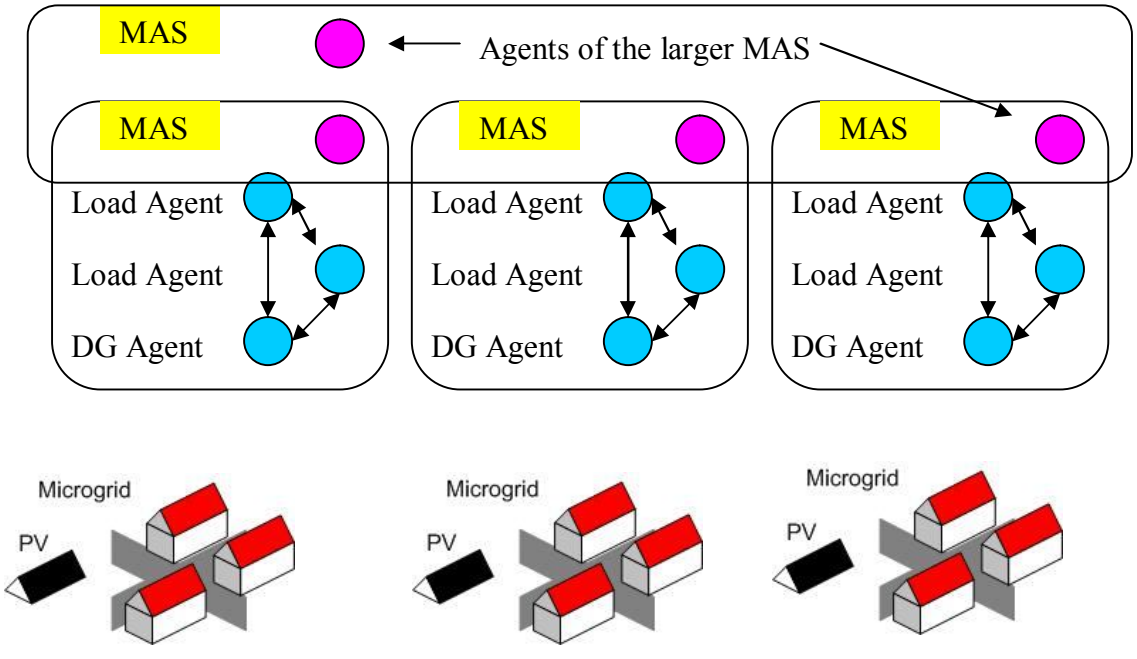
5.3 Advanced Architecture

Next question in the development of the MAS system is whether this technology is sufficient for large scale systems. The simple architecture is insufficient in controlling larger number of agents since the complexity increases significantly. Therefore, an extended architecture is introduced in Figure 8. The DG units and the controllable loads form small Microgrids and accordingly small MASs. These MASs form larger MASs and so on. The separation may be realised based on the electrical and topological characteristics like common MV transformer.

The groups of MAS are organized in three levels. The three levels are presented in Figure 9. All the agents associated directly with the control of the production units or controllable loads belong to the ***Field Level***. These agents directly communicate and control a production unit or a load and may be organized in MAS according to physical constraints of the system. Each of these MAS has also an agent that has the responsibility to communicate with other higher level MASs in order to cooperate with them. These MAS belong to the ***Management Level***. Finally, these MASs may form larger MASs in order to participate in the ***Enterprise Level***.



Case 1 Single Microgrid



Case 2 Multiple Microgrids

Figure 8 General scheme of MAS architecture

The model of the three layers for the distinction of the multiple MASs proposed above, has been based on principles both from the Multi Layer Learning approach described in the previous section as well as by the way a manufacturing enterprise is organised. Therefore, the field level consists of agents that are directly connected to the DG or the controllable loads and the Management level tries to coordinate the multiple MASs. Finally, in the Enterprise Level, strategic decisions are taken such as Market Participation. It should be mentioned that the higher the level the more abstract the agents are. Thus, in the Enterprise Level the ESCo could be considered as an agent.

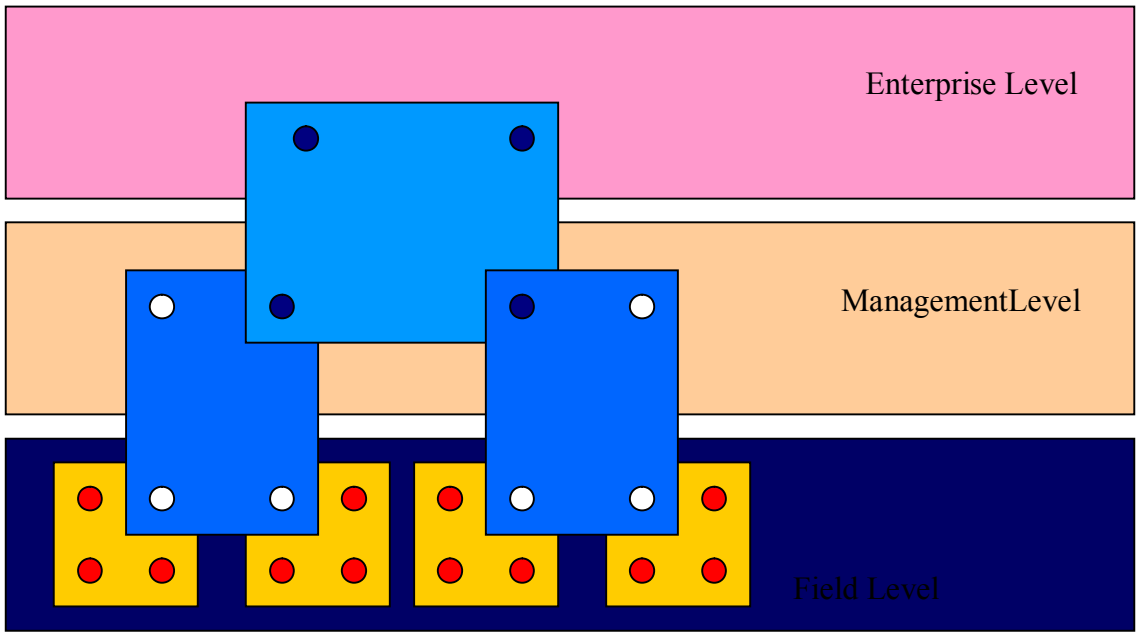


Figure 9 Management of the agents

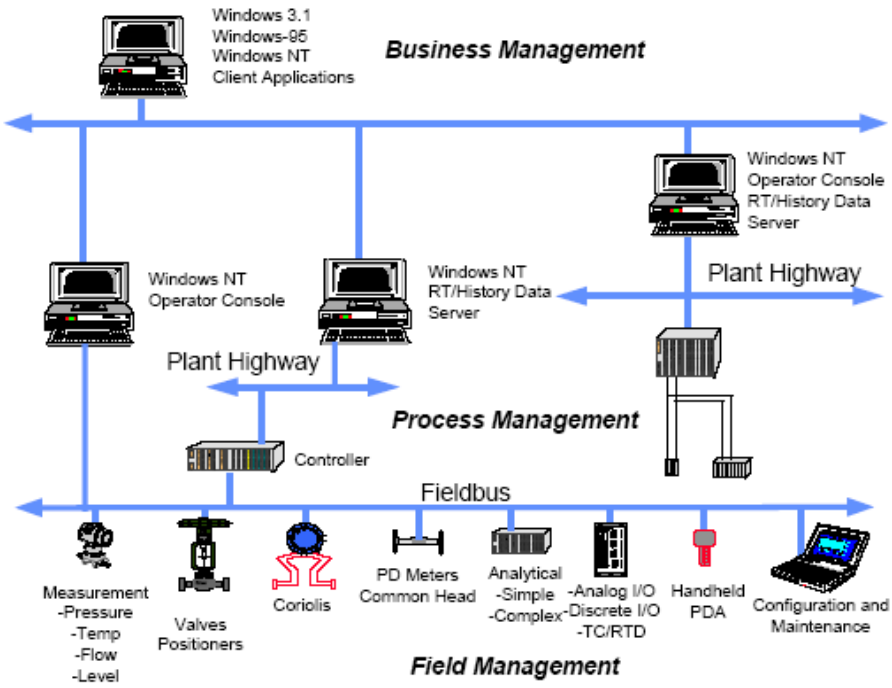


Figure 10 Management of a control process

6. Implementation of MAS

6.1 Introduction

The previous section provided detail on how the various services of the control system are organised and what kind of algorithms are used in order to accomplish the various tasks. However, a critical point is how these ideas are going to be implemented in a real system. This will be described in this section.
In order to proceed with the implementation, two main issues, related to each other, should be taken into account. The first one is how to model and formally describe all the processes and the system. The second concerns the development tool that will be selected for the implementation. It is obvious that these issues are strongly related to each other since the capabilities of the later should be taken into account during the design.
Therefore, since next section describes the modelling issues regarding the implementation, it should be mentioned first that the selected tool is the Java based toolbox Jade. It is not the scope of this document to justify whether Jade is the optimal available tool for this implementation nor to discuss anything about Java. However, it should be mentioned that the selection of a Java based tool may be unsuitable for high speed applications such as network reconfiguration. Besides, this type of applications the performance of Java is acceptable. In a later section more detailed comments may be given.

6.2 Formal design of the system

The formal design of the system and its various processes is critical for a complex system such as the Microgrid control system. The main complexity is due to the presence of simple but numerous processes that interoperate.
The design process is divided into two parts as shown in Figure 10:

1. Modelling the internal behaviour
2. Modelling the interactions with the other agents

The internal modeling suggests the use of a tool able to graphically describe the interaction of the various processes within the agent and more specifically the implementation of the Multi Layer Learning Model.
The interactions modeling is the other critical part of the MAS design, since it focuses mainly on how to define the communication dialogues among the agents and considering that this does not only concerns a common ontology but also the structure of the various dialogues.
In order to clarify this, an example will be presented. Consider the resource allocation algorithm presented in the previous section. The design of the algorithm itself as well as its interaction with the other processes (such as battery management or load control) is part of the internal modeling. On the other hand, the interaction among the agents according to this algorithm is part of the interaction modeling.More specifically, not only a common ontology should be described (e.g. common definition of the bids) but also a formal description on how the various dialogues are

settled (e.g. How do the agents know that a new cycle starts without having a bottleneck).

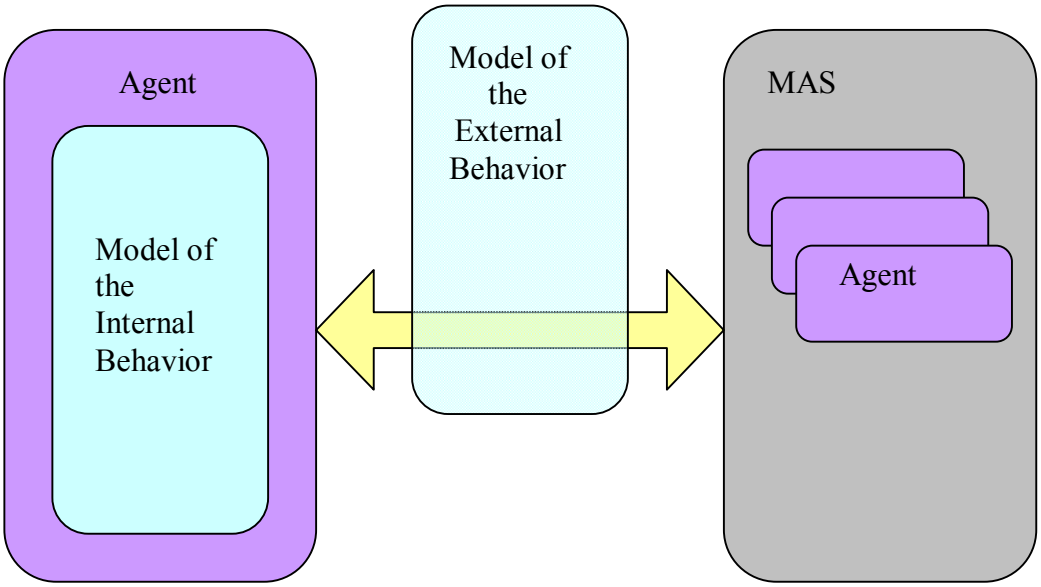


Figure 10 Modelling the agent behaviour.

Finally a significant part of the development of the MAS system is the definition of the ontology. Due to its significance, this will be described in a separate section

6.3 Design tools (EPC)

For the design of the processed the EPC (Event Driven process Chain) methodology is selected, since it provides an easy way to describe all the processes. The fundamental elements of the EPC are:

The "state or function" block that describes a process or a current state of the system. For example the function "Calculate next bid" or the state "Stand by" could be described by this block.

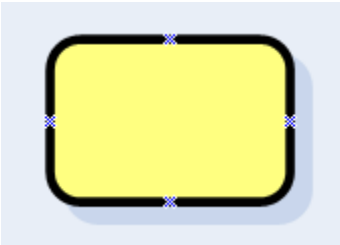


Figure 11 State or Function

The "Event" function block refers to an incoming event such as a measurement, a message from an other agent or an event from an other process.

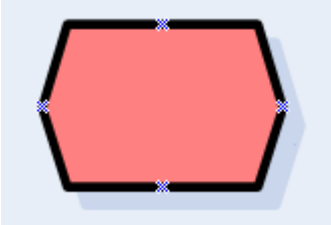


Figure 12 Event

The XOR operator defines a transition to one of the Multiple selected choices. Similar for inputs.

Figure 13 XOR

The OR operator defines a transition to at least one of the Multiple selected choices. Similar for inputs.

Figure 14 OR

The AND operator defines a transition to all of the Multiple selected choices. Similar for inputs.

Figure 15 AND

6.4 The external design

The external design focuses on the implementation of the communication and the coordination of the agents. In this part, we do not include the design and the implementation of the algorithms which are considered as parts of the internal design. However, the external design should support the implementation of any algorithm as well as allow the incorporation of multiple processes or future processes.
Since the difference between the internal and the external design is not clear enough, an example should be given for clarification. The problem the agents should try to solve is the network consistency identification, taking into account that each agent knows only the neighbouring agent. This means that agent A knows whether a direct

cable connects him with Agent B. The scope of the algorithm is whether the agents can identify that they belong to the same network or they are in independent islands. In real problems, this algorithm is similar to the problem of identifying in which MV transformer the agents belong.
The core of the algorithm is based on a fundamental algorithm of the graph theory called Depth First Search (DFS) for travelling in all possible nodes in a directed graph. Given the adjacency matrix representation of the graph, the simple recursive algorithm in pseudo language is:

```
DFS (vertex x)
{
Flag_vertex_as_visited(x)
For each neighbor vertex y of x
                If y is unvisited
                {
                DFS (y);


   }
}
```

In figure 16 a sample graph is presented. The nodes are enumerated by the order they are visited by DFS.

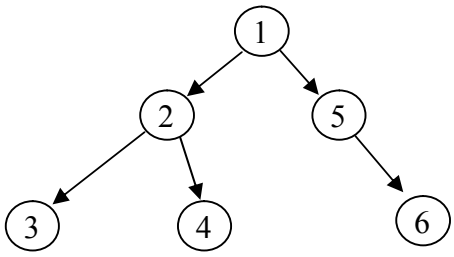


Figure 16 Sample graph.

The implementation in the case of a Multi-Agent System is based on tokens and the only necessary knowledge about the electrical system is the knowledge of the neighbouring agents. The steps of the algorithm are the following:

- One agent starts the algorithm by sending one token to each neighbouring agent.
- The agent that has received the token sends new tokens to each one of his neighbouring agents except to the one that passed the previous token to him.
- The algorithm stops under two conditions: if an agent has none other agents to pass the token or if the number of steps is greater than the total number of the agents. The last condition is essential in cases where the network is not radial and the algorithm would continue for ever.

After the end of the algorithm if one or more agents have not received the token, then the network has at least one island.
The implementation of this algorithm in a MAS environment could have several approaches. The one selected in this text focuses on the agent that starts the search, called MGCC (Microgrid Central Controller. This agent decides whether it should start the algorithm for a specific part of the grid. In order to start the algorithm the MGCC sends a "start algorithm" message to the first agent as presented in figure 17.

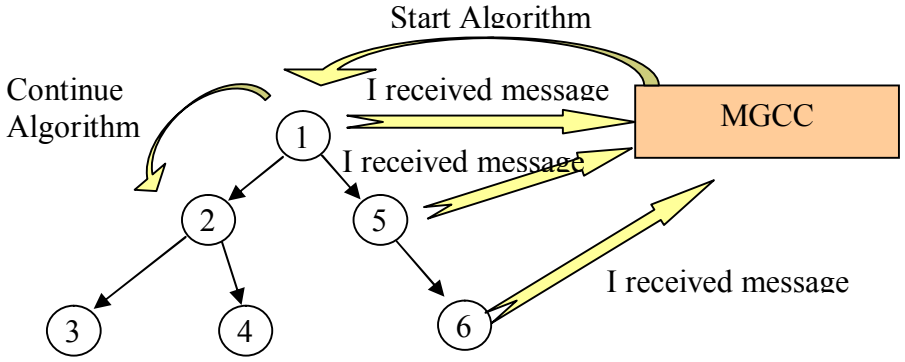
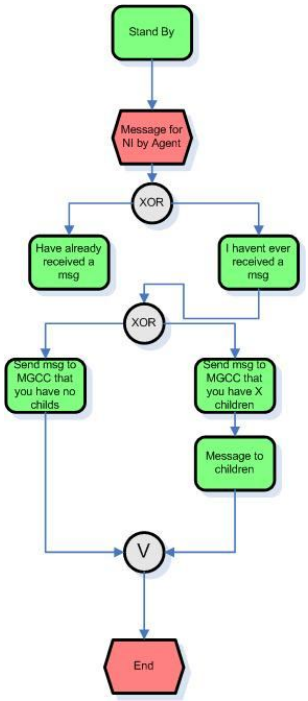


Figure 17 the realization of the DFS algorithm in MAS.

Once an agent receives this message he should inform MGCC how many children it has and afterward send a proper message to each one of the children in order to continue the algorithm. The MGCC just counts the messages and the algorithm stops when all agents have been visited.
In Figures 18 and 19 the EPC diagrams of the MGCC and the network agents are presented. As mentioned before the internal modeling focuses on the implementation of the algorithm inside the agent and more specifically on how the agent reacts to the various messages.



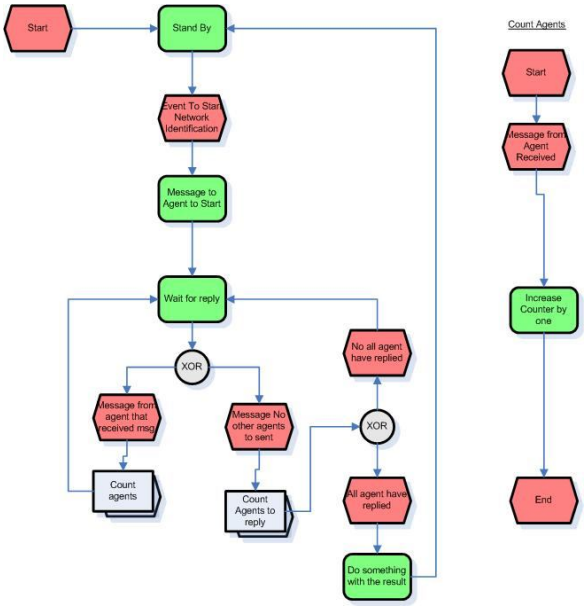


Figure 18 The EPC diagram of the Agent. Figure 19 The EPC diagram of the MGCC.

However, the issue in this section is the external design and what is its main purpose. There is one difficulty in this application not covered in detail yet in this document. Taking into account that multiple algorithms of this type may exist in the system, it is obvious that multiple messages are exchanged among the agents for different algorithms or even within the same algorithm. Furthermore, as discussed in the beginning of this document one of the most powerful capabilities of the intelligent agents is their communication ability.

Communication is one of the main elements that allow the intelligent agent to form a society. The communication problem focuses on the context of every message and the knowledge the agents exchange. Figure 20 presents the agent version of the story of the Tower of Babel where the workers could not finish the tower since they could not communicate.

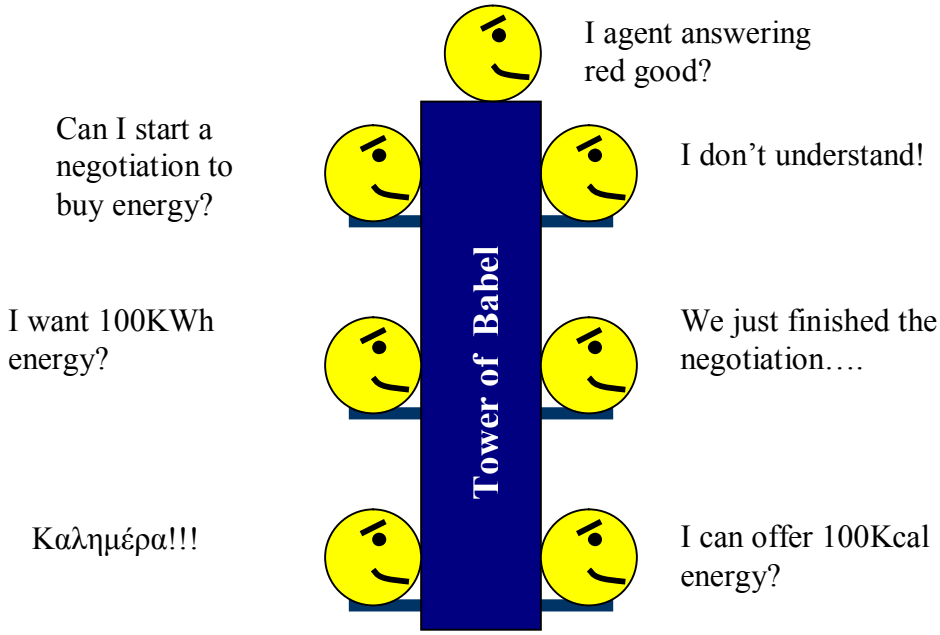


Figure 20. The problem of coordination

In Figure 20 the three main problems of the agent communication can be identified:

1. The first problem is the ontology or the vocabulary. All the agents speak English except one that says "good morning" in Greek. It is vital that the agents should have a common vocabulary. But this requires more than just using the same words. Same words should have the same meaning for all agents. In the example, one agent asks for energy in kWh and the other answers in Kcal. It is obvious that both agents should understand the word energy in the same.
2. One of the agents says "I agent answering red good?" which is a phrase without an understandable meaning. The agent messages should have a common structure that is provided by FIPA in ACL. This will be analyzed next.
3. The final problem is revealed, if we focus on the discussion of two agents of the example. The first asks to start the negotiation and the second replies that it just finished. This is a critical problem in an environment with multiple dialogues and a protocol that defines how each dialogue should be structured and identified must be introduced.

6.5 Formal Agent Ontology

One main feature of FIPA compliant MAS platforms is using the ACL (Agent Communication Language) with high-level ontologies. It is a general conclusion that high level communication is one significant element to develop intelligence inside a society, no matter if this society is a human one or a Multi Agent System. For the system presented here, an ontology was developed according to the needs of the system. This ontology supports two main tasks.

The first task is to provide an adequate description of the system, so that it includes all the necessary information for its control. Thus, all software modules of the control system perceive the system in the same way and this is important in order to develop a high level communication system, where the agents exchange knowledge. In order to understand the importance of having a common perception of the environment, consider the concept "energy" which has different meaning in quantum physics, food, or electric systems. Furthermore, the object oriented nature of the ontology and the data abstraction, support the development of a distributed control system, since each agent handles only the necessary (or allowable) part of information and knowledge.

6.6 Common Information Model

The Common Information Model (CIM) is an abstract model that represents all the major objects in an electric utility enterprise typically involved in utility operations. By providing a standard way of representing power system resources as object classes and attributes, along with their relationships, the CIM facilitates the integration of Energy Management System (EMS) applications developed independently by different vendors, between entire EMS systems developed independently, or between an EMS system and other systems concerned with different aspects of power system operations, such as generation or distribution management. This is accomplished by defining a common language (i.e., semantics and syntax) based on the CIM to enable these applications or systems to access public data and exchange information independent of how such information is represented internally.
The object classes represented in the CIM are abstract in nature and may be used in a wide variety of applications. The use of the CIM goes far beyond its application in an EMS. This standard should be understood as a tool to enable integration in any domain where a common power system model is needed to facilitate interoperability and plug compatibility between applications and systems independent of any particular implementation.

7. Technical Implementation of MAS

This section focuses on how the MAS system is implemented. The basis of the implementation is the common used agent SDK called Jade.

7.1 Jade

JADE (Java Agent Development Framework) is a software development framework aimed at developing multi-agent systems and applications conforming to FIPA standards for intelligent agents. It includes two main products: a FIPA-compliant agent platform and a package to develop Java agents. JADE has been fully coded in Java and an agent programmer, in order to exploit the framework, should code his/her agents in Java, following the implementation guidelines described in this programmer's guide. This guide supposes the reader to be familiar with the FIPA standards1, at least with the *Agent Management* specifications (FIPA no. 23), the *Agent Communication Language,* and the *ACL Message Structure* (FIPA no. 61).
JADE is written in Java language and is made of various Java packages, giving application programmers both ready-made pieces of functionality and abstract interfaces for custom, application dependent tasks. Java was the programming language of choice because of its many attractive features, particularly geared towards object-oriented programming in distributed heterogeneous environments; some of these features are Object Serialization, Reflection API and Remote Method Invocation (RMI).
The standard model of an agent platform, as defined by FIPA, is represented in the following figure.

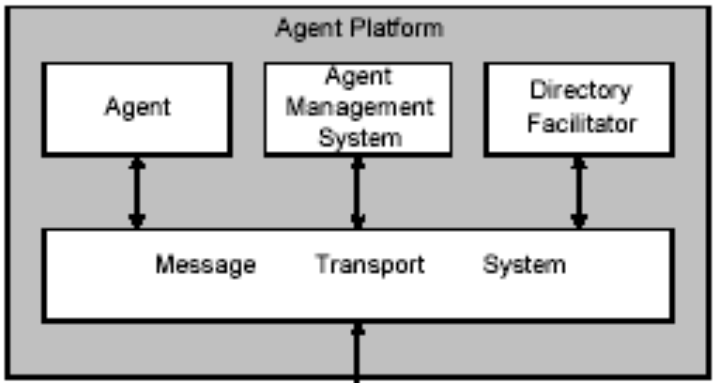


Figure 6. The AMS platform

The Agent Management System (AMS) is the agent who exerts supervisory control over access to and use of the Agent Platform. Only one AMS will exist in a single platform. The AMS provides white-page and life-cycle service, maintaining a directory of agent identifiers (AID) and agent state. Each agent must register with an AMS in order to get a valid AID.
The Directory Facilitator (DF) is the agent who provides the default yellow page service in the platform. The Message Transport System, also called Agent Communication Channel (ACC), is the software component controlling all the exchange of messages within the platform, including messages to/from remote platforms.

JADE fully complies with this reference architecture and when a JADE platform is launched, the AMS and DF are immediately created and the ACC module is set to allow message communication. The agent platform can be split on several hosts. Only one Java application, and therefore only one Java Virtual Machine (JVM), is executed on each host. Each JVM is a basic container of agents that provides a complete run time environment for agent execution and allows several agents to concurrently execute on the same host. The main-container, or front-end, is the agent container where the AMS and DF lives and where the RMI registry, that is used internally by JADE, is created. The other agent containers, instead, connect to the main container and provide a complete run-time environment for the execution of any set of JADE agents.

7.2 Implementation of MML

In the previous section, the internal design of the agent was based on the distinction of the various processes according to the Multi Layer Learning (MLL) technique and on the modelling according to the EPC diagram. Scope of this section is to describe how these two ideas were implemented within the agent.

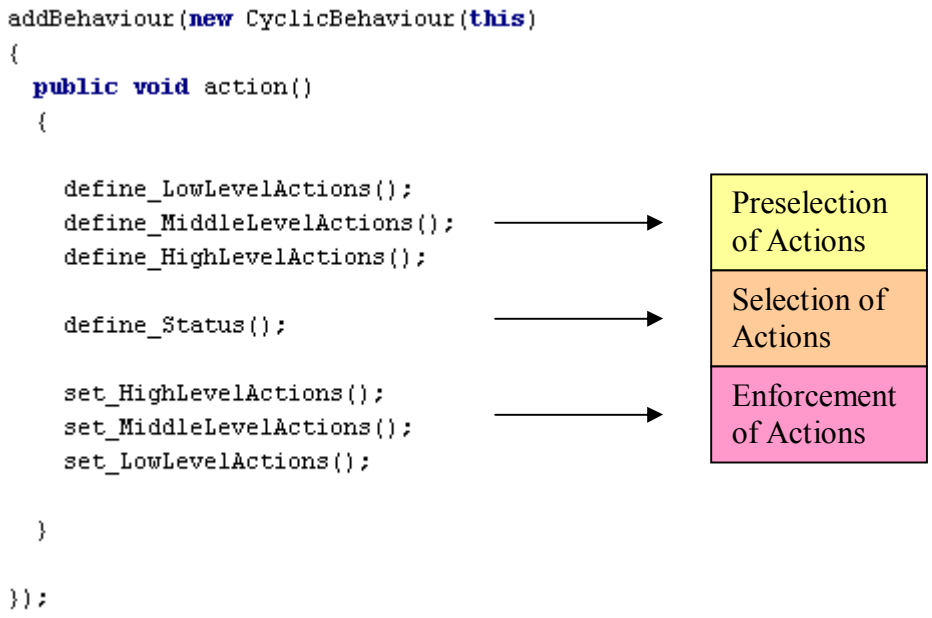


Figure 22. Implementation of the Multi Layer Learning in Jade.

Figure 22 presents the main behaviour of the agent regarding its actions. There is a second behaviour for receiving messages which is omitted for the sake of simplification. As shown in the figure, there are three main parts in the realisation of MLL which are essential for the realisation of the approach. The decision process is realised based on the scheme that the agent should first define its status in all levels and afterwards decide next actions. This is clearly illustrated in figure 23. During the preselection of actions the agent checks the status and the demands in the three levels (Local,MAS,Team). Taking into account the requests in these levels, the agent should decide the next actions providing priority to the lower level (Local).

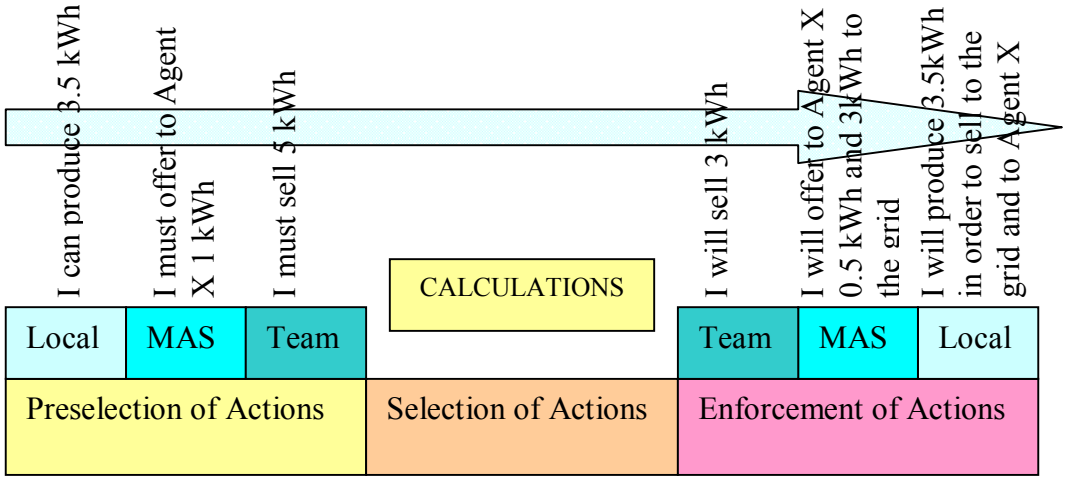


Figure 23 Decision making model

Next issue is how the agent incorporates the various services that he should provide to the system. The implementation became simple, using an object called myStatus as shown in figure 24.

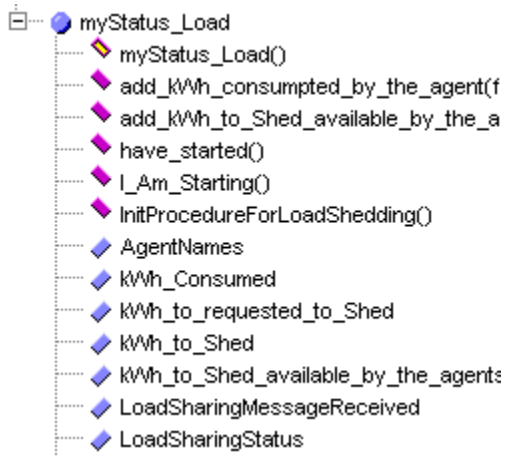


Figure 24. Object myStatus for Load controllers.

This object stores all the information regarding the current status of the agent. Furthermore, it stores the information on which sub process is currently according to the EPC diagram (Figure 25). The implementation of the behaviour is actually event driven, meaning that the agent changes states when an event occurs. This type of implementation provides another critical advantage, which is the ability to incorporate multiple parallel services. However, this characteristic is subject to the capabilities of the programming language as well the processor capabilities.

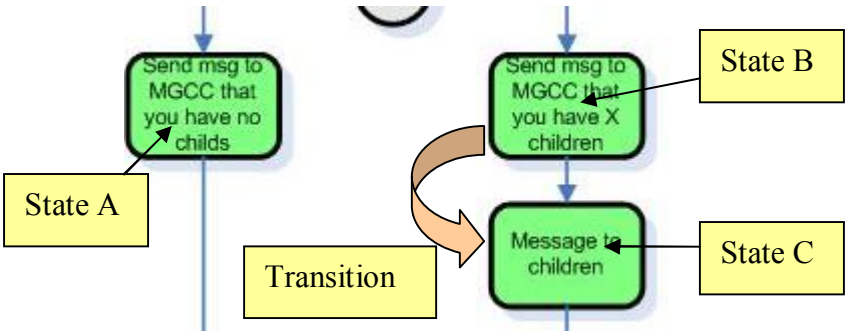


Figure 25 Coordination of States.

7.3 Implementation of Ontology

One of the most important parts of the agent communication, as described (above ?) , is the ontology which is the common vocabulary of the agents. Goal of the ontology is to provide a description of all the concepts that may appear in a conversation between the agents. CIM provides a good description of the electrical system, although it is not complete since description for small DG units is not present.
Furthermore CIM is a very complicated structure with several objects and it is far beyond the scope of this implementation to provide a fully compliant FIPA ontology. Therefore, only very elementary objects are imported as FIPA compliant ontology and the rest of the object are sent as Serializable objects within the message context. The creation of the Java file of the CIM, as presented in Figure 26 was created with tools that create the Java files from the Rational Rose UML Model

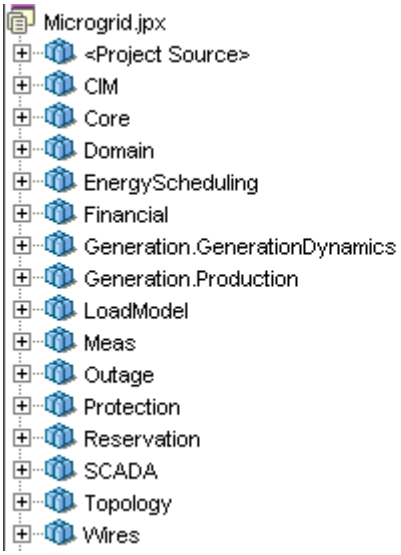


Figure 26 Usage of CIM in the Microgrid

7.4 Implementation of plug n' play capability

One of the very important services provided by the FIPA SDK is the operation of the DF agent. DF Agent provides Directory Facilitator (DF) services which are similar to the yellow pages operation of the phone catalog.
When an agent is created, he automatically registers the services it can provide to the DF service. Therefore, when an agent wants to send a message to the Load Controller, asks the DF to provide the list of agents for this service.

```
protected void TestVoid()
{
  int NumberOfAgents = 0;
  ActivePower AP = new ActivePower();
  NumberOfAgents = GetNoOfAgentWithService(v.SERVICE_LOAD_CONTROL);
  if (NumberOfAgents > 0) {
    for (int i = 0; i < NumberOfAgents; i++) {
      sendCIMmessage(AP, GetAgentWithService(v.SERVICE_LOAD_CONTROL, i),
                     ACLMessage.INFORM, conv.LoadShed);
    }
  }
  ;
}
```

Request for agents that provide a specific service

Send Message to agents that provide a specific service

Figure 27 Typical method to send a message to all load controllers.

This is one significant advantage of the MAS since it automatically registers itself and announces to the other agents the services it may provide. The communication as well as the algorithms are based on this functionality which means that all messages are sent via the DF and not directly from one agent to the other. In this way, when a new agent participates in the system, all the related agents detect its appearance and change their behavior accordingly.

	Service	Description
1	SERVICE_MGCC	Coordinating Agent
2	SERVICE_LOAD_CONSUMPTION	Controlled Load
3	SERVICE_POWER_PRODUCTION	Controlled Production Unit
4	SERVICE_POWER_SELLER	This agent participates in the market as power seller
5	SERVICE_POWER_BUYER	This agent participates in the market as power buyer
7	SERVICE_BLACKSTART_PRODUCTION	This production unit may be used as black start unit.

Table 2. Some of the services implemented in the MAS.

PART B Contribution of LABEIN

This document describes algorithms aimed to efficiently operate a Microgrid. The objectives of a Microgrid Management System are to economically operate the different resources in the Microgrid while maintaining a proper quality of supply for the users connected to it. Other drivers such as minimizing emissions and performance optimization could be also taken into account.
Operating the Microgrid in an economical way implies that the resources must be managed considering electricity prices and DER devices' operating costs in order to reduce the money paid for electricity or, in the best case, to obtain the maximum incomes by selling electricity from Microgrid's resources.
Guarantying an acceptable quality of supply is the first priority for a Microgrid Management system: It must ensure that demand and supply are matched in real time with acceptable frequency and voltage characteristics. In addition, when some fault occurs inside or outside Microgrid domain, the management system must ensure that electricity supply is not lost or at least, that the maximum number of loads is still supplied while outage time is minimized.
The management system algorithms must adapt to different scenarios regarding who owns and who operates the Microgrid and the elements within it. In an scenario where the operator of the Microgrid owns the devices installed in it, the management can be done by centrally operating the Microgrid while in an scenario where generation and loads are owned by different parties, the approach could be to operate in a decentralized way leaving the decision on how to participate in Microgrid operation to devices' owners.
The Microgrid system operation can be divided into different states where each state has its particularities, therefore, different functions must be executed to ensure proper system behaviour at each state. Following figure shows the state transition diagram for a Microgrid:

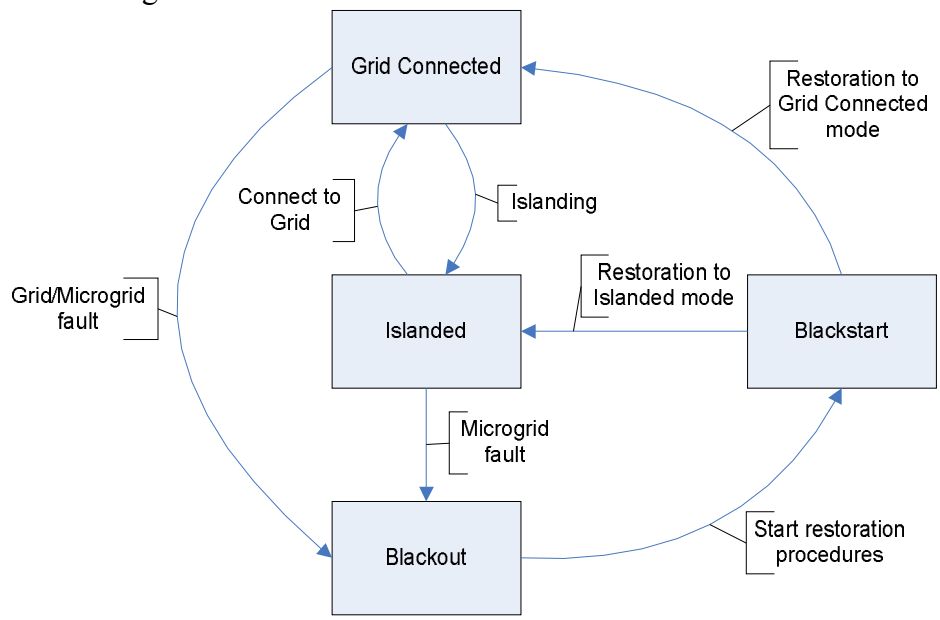


Figure 1: Microgrid state transition diagram

The normal situation for a Microgrid with main grid connection possibility will be the Grid Connected state, in this state the Microgrid Management System is in charge of maintaining voltage characteristics inside a certain admissible range, in addition, the

Microgrid will be managed so the owners of the different DER devices obtain the maximum profit with the possibility to buy or sell electricity from the main grid.
If a failure occurs in the main grid, the Microgrid will enter the Islanded mode. This mode tries to supply as many loads as possible, if the available generation is not enough to supply connected loads, the most uncritical ones should be shed while the critical loads would maintain a acceptable quality of supply. This working mode is more sensible to variations in frequency characteristics since there is no support from the main grid. The priority for the Microgrid Management System is to feed as much loads as possible while economically optimizing DER resources dispatch.
From the Islanded mode the Microgrid Management System will try to reconnect to the main grid as soon as possible returning to the Grid Connected mode.
From Grid Connected and Islanding modes the Microgrid can enter Blackout state if some severe fault occurs or system stability is lost. This state implies that some generating units have been disconnected from the Microgrid and there is no any electricity supply at all for some of the segments in it or even for the entire Microgrid. From this state the Microgrid enters the Blackstart state where system restoration takes place.
The operations to be executed in the Blackstart state will bring the Microgrid to stable operation. Depending of the Microgrid configuration and if the main grid is available for connecting to it, different operation sequences can be performed. For example, if the main grid is available the Management System could try to reconnect to it and then start connecting all the resources in the Microgrid, whereas if the main grid is not available the Management System could be in charge of reconnecting the generating units and loads progressively in order to achieve an Islanded stable mode.
Regarding hardware and communications architecture, we consider that there is a Microgrid Central Controller with some centralised functions and several Local Controllers holding local functions.

Figure 2 shows the control hierarchy for the Microgrid system, where MGCC is the Microgrid Central Controller, GC-s are local Generator Controllers, LC-s are local Load Controllers and SC local Storage Controllers:

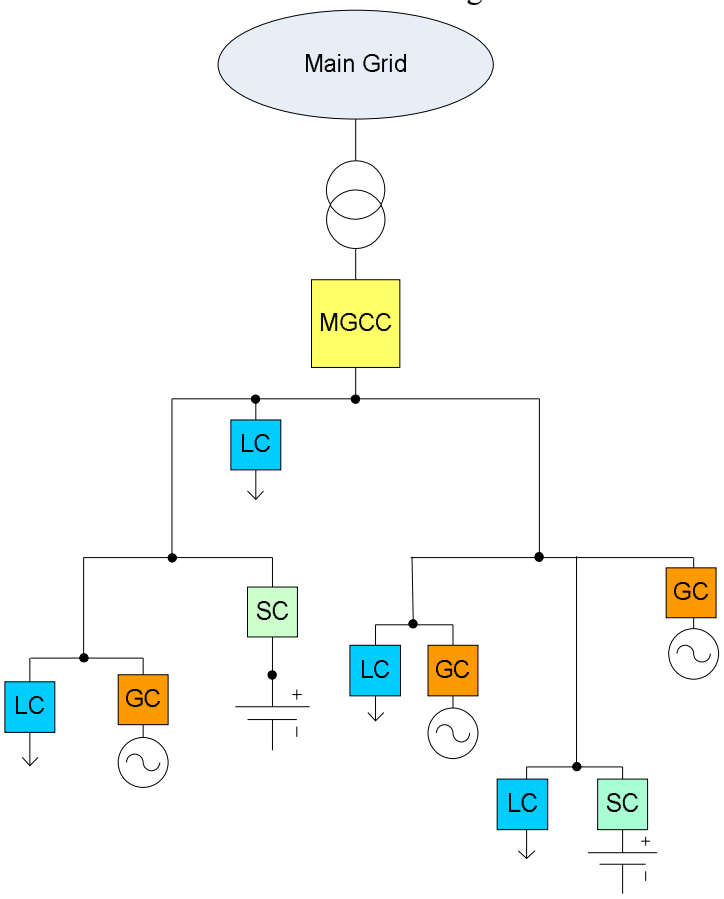


Figure 2: Microgrid control architecture

The above shown architecture allows distributing the intelligence needed to perform the management functions among the local and central controllers; this fact allows developing highly distributed and decentralized systems for implementing those functions.

Among all the different functions that can be developed for the Microgrid Management System, this document focuses on describing a ***Decentralised Secondary Control Algorithm*** aimed to both operate in Grid Connected and Islanded modes ensuring the proper quality of supply while economically dispatching the devices in real time Microgrid operation. In addition, and in order to effectively develop the mentioned secondary regulation function a ***Local Generation, Storage and Load Schedule Tracking System*** will be described, this function permits devices to run autonomously once a power schedule is sent to them.

The latter function is a general purpose function and will be used as base for complex and high level functions like the first one.

8. Decentralized Secondary Regulation Control Algorithm

8.1 Why is Secondary Regulation Needed?

In conventional electricity systems, the first approach for committing and dispatching generating units is to match in advance the power demand with power generation, as result a first generating unit dispatch is obtained. This unit dispatch can be performed following market rules where electricity is bought and sold by the different actors participating in the market or by a centralized generation resources economic optimization in order to match a forecasted demand.

The initial unit dispatch can be done with different time horizons but the point is that finally in a real time basis, deviations from the expected power demand occur. In order to maintain generation and consumption balanced in real time, generators have their own primary control mechanisms that react and adjust their output power according to real time demand variations. The new system equilibrium after the primary control actuation carries a different frequency in the system and different power productions from the scheduled ones, and thus, a deviation of the power shared between the different regulation zones. In order to share these real time power production deviations and to recover frequency to its reference value, secondary regulation control is applied to reassign power set points to generators in a certain regulation zone taking into account the reference frequency to be maintained, economical optimization and system stability [1] [2].

Finally and if the system operator detects that the secondary regulation capacity is below some target value, it can commit new units in order to ensure system security, this is called tertiary regulation control.

For a Microgrid scenario, a similar approach can be taken when operating in real time. It is expected that after market like sessions or Microgrid economic optimization, a first generating unit schedule set is obtained. However, in real time operation certain amount of demand changes from the expected one and RES production varies from the forecasted one. Here it comes the need for the secondary regulation control that must provide new set points to generators and controllable loads in order to share in an efficient and secure way the difference between the scheduled power and the real generating power.

Following chapters discuss two different scenarios, the first one when the Microgrid is in Grid Connected mode and the second when it works in Islanded mode.

The deviations from planned schedules depend on the Microgrid size in terms of installed load, controllable generation and installed RES generation without storage capacity. The greater the size of the Microgrid the fewer the deviations will be. On the contrary, the smaller the size of the Microgrid, the higher deviations will take place.

8.2 Secondary Regulation in Grid Connected Mode

When connected to the main grid, any changes in load or RES power production will be assumed by the main grid and frequency will be also the one imposed by the main grid. Deviations in Microgrid generated or consumed power will have little or no effect over system frequency and stability.

The duty of the secondary control in this situation is to maintain the power exchange with the grid according to the contracted schedule with the main grid operator. In addition, the Microgrid could also offer to the main grid operator the possibility to contribute to the minimization of frequency deviations; this ancillary service could

have sense in a scenario with Multi-Microgrids where operating the different Microgrids this way could have a significant effect on system frequency.
The criteria to decide how DER units are going to share the excess or defect of power must follow economic and technical criteria such that the Microgrid operation is done at minimum costs while technical constrains are met.

8.3 Minimization of the power exchange deviation

The fact of maintaining the power exchange within contractual schedule is a valuable service for the main grid operator since RES production and demand variability will be hidden for him. The Microgrid Management System will be in charge of monitoring the power exchange with the main grid, if any deviations from the schedule are detected, the management system must share the exceeded or defective amount of power among the controllable resources in the Microgrid.
In case the main grid operator does not restrict nor penalize the Microgrid operator whatever the amount of exchanged power is, maintaining scheduled exchange is not needed and DER units could be dispatched according to the initially optimized schedules. However, and as said before, it is expected that the main grid operator will limit and penalize deviations of imported or exported power, for this later scenario secondary regulation is necessary.
On one hand, in case an excess of power is being exported to the main grid or a defect of power imported from the grid the secondary control algorithm will have to decrease the generation of the Microgrid. This will be done by calculating the deviation from planned schedule and assigning new set points to generators or storage devices according to economic or security criteria.
On the other hand, in case the Microgrid is exporting less than scheduled power or importing an excess according to planned power, the Microgrid will be in charge of increasing the power generated by generating or storage devices according to economic and security policies.

8.4 Contribution of the Microgrid to Main Grid's frequency recovery

As said before, the Microgrid could also contribute to minimize grid frequency deviation by modifying the exported or imported power. The impact of such power exchanges depends on the Microgrid size but in general it will be very low comparing to the size of the Main Grid. Anyway, in a scenario with Multi-Microgrids, it could be useful for the main grid operator to make use of the secondary regulation power available from Microgrids.
Different approaches can be taken when assigning the participation factor of the Microgrid into main grid's secondary regulation: The participation of the Microgrid in grid frequency regulation could be contracted with the main grid operator, or could be set in a market environment where the Microgrid offers secondary regulation bids to the grid operator that assigns secondary regulation participation factors to available systems, other approach could be to leave the Microgrid decide the amount of secondary regulation power to export or import depending on an known price for that secondary regulation energy.

9. Secondary Regulation in Islanded Mode

When the Microgrid is in islanded mode, the operation is more sensible to changes in consumption and RES production since there is no support from the main grid. In this case, and in a Microgrid where generators and storage devices are connected through inverters, when a deviation of the demand or generation is produced, the resources acting as voltage sources will instantaneously assume the excess or defect in power following their P-Fr droop characteristics. Consequently and as frequency changes, generators acting as current sources and also equipped with P-Fr droops will help to reduce frequency deviations. After primary regulation stabilises the system, it will work with a different shared power compared to the initially planned power schedules and also with some frequency deviation from the reference one.
The secondary regulation algorithm will be in charge of detecting Microgrid frequency deviations, then calculating the amount of power needed to return to reference frequency and finally sharing that power among the resources in the system. The secondary regulation control will share the power among resources available for secondary regulation control according to economic, technical and security criteria.
An especial case is the one where a set of Microgrids connected to the same feeder disconnect from the main grid, in this kind of Multi-Microgrid system working without the main grid, each Microgrid could agree with the rest of the Microgrids the amount of power to export or import, in this case the secondary control will perform similar functions as the ones described in chapter 8.2, that is, maintain agreed exchanged power with the rest of the Microgrids and contribute to reduce the frequency error of the entire Multi-Microgrid islanded system.

9.1 System Architecture

This chapter presents the architecture of the system that performs the secondary regulation control. The presented approach is a distributed system where each part in the system has a high degree of autonomy. It is designed to deal with both grid connected and islanded operation scenarios as described in chapters 8.2 and 9. The system is designed taken into account following Microgrid characteristics:

- It is a distributed system, where resources are dispersed in the network.
- A Microgrid constitutes a dynamic environment where resources can be added or removed from the Microgrid.
- It is composed of heterogeneous systems with different generation, storage and load technologies, together with different communication protocols and control capabilities.

In order to match these characteristics a distributed intelligence system is presented with high degree of autonomy, flexibility and plug and play capabilities.

Two different elements form part of the secondary regulation control system. On one hand, the ***Secondary Regulation Matcher*** is in charge of collecting secondary regulation bids from the controlled devices and creating new power schedules for those devices in order to minimize the power and frequency deviations while economically operating the Microgrid. On the other hand each device participating in the secondary regulation control will have a ***Secondary Regulation Bidder*** in charge of creating secondary control power bids. This approach puts in the local intelligence the duty of considering to what extent is going the device to collaborate on the secondary regulation power sharing.

9.2 Characterization of devices

The following devices will be considered as potential participators into secondary regulation control:

- Generators: Distributed generators usually are fast enough reacting to power set points so they can participate into secondary regulation by reducing or increasing their power output. Non controllable generators such as RES or any other generator without communication or control capabilities will not be taken into account by the Secondary Regulation control.

- Storage systems: The same as generators, storage systems can participate by increasing or decreasing their power production and also increase or decrease their power consumption depending on their working conditions.

- Controllable Loads: Non critical loads could reduce their power consumption offering power reduction capacity to the secondary regulation control. There are loads that can reduce their power consumption a certain amount for a given time period such as air conditioning systems or electric heating systems and loads that can be shed to be switched on in a later time period such as washing machines or driers.

- Main Grid: Although the main grid can not be considered as a Microgrid device and it is not controllable in the sense that it cannot accept power set points, it plays a crucial role in the secondary control when working in connected mode. The secondary control must consider the power imported or exported from the main grid and maintain it within the contractual limits accorded with the main grid operator.

The devices enumerated before, can be classified into the following categories:

- Devices participating into the secondary regulation:
    - Main Grid: The secondary control must ensure that the power exchange schedule with the main grid is maintained when consumption or generation changes occur.
    - Devices with adjustable power schedule: These devices provide the power production and consumption decrease and increase capacity. The secondary regulation assigns new set points to these devices by updating their previous schedules while ensuring economically optimum operation.
- Devices outside secondary regulation control:
    - Non controllable devices: Renewable generators without storage capacity, not controllable loads and any other device without communications or control capabilities.
    - Devices with fixed schedule: Those devices having fixed power schedules are not considered by the secondary regulation control. The local generation schedule tracking system is the one in charge of ensuring that the device complies with its assigned schedule. These devices can be the ones that do not have enough reaction speed or the ones whose owners do not want to participate into secondary regulation control.

9.3 Secondary regulation matcher algorithm

The Secondary Regulation Matcher is in charge of collecting all secondary regulation bids from local controllers and deciding the new set points to be assigned to those devices, with the aim of reducing the frequency deviation and following the power exchange schedule while economically operating the Microgrid.
The key concepts involved in the algorithm are explained in the following chapters, together with the algorithm sequence description.

9.4 Microgrid Regulation Error (MRE)

The MRE represents the error given in terms of active power that must be corrected by the secondary regulation control system. The MRE has two different parts; the first one corresponds to the difference between the scheduled power exchange and the real measured power exchange at the Point of Common Coupling [3] while the second part is the frequency error, that is, the difference between the reference frequency and the measured frequency multiplied by a participation factor. The formula is given as:

$$MRE = \Delta P^{pcc} + k \cdot \Delta fr$$

where ΔP^{pcc} (Watts) is the difference between scheduled and measured power at the point of common coupling, k (Watts/Herzs) is the participation factor that multiplied by the frequency error gives the power correction to be applied for restoring frequency and Δfr (Herzs) is the difference between measured and reference frequency.
Depending on the Microgrid operating mode, islanded or grid connected, the participation factor is calculated two ways:

- Microgrid in Islanded operating mode: In this case the participation factor is given by:

$$k = \sum_{1}^{n} \frac{1}{R_i} + \sum_{1}^{l} \frac{1}{D_j} \quad 0 < i \leq n, 0 < j \leq l$$

  Where R_i (Herzs/Watts) is the frequency versus power droop for the generating unit i, the sumatory considers all generating units in the Microgrid. D_j (Herzs/Watts) is the frequency versus power variation of non-conforming loads such as electric motors, and the sumatory represents the sum of the inverses of all non-conforming loads in the Microgrid.
  This way of calculating the participation factor ensures that the resulting power correction restores frequency to its reference value.
- Microgrid in Grid Connected operating mode: when working connected to the grid the participation factor indicates the amount of power that the Microgrid will regulate in order to collaborate in restoring system frequency.

  Different approaches could be taken in order to define the value for the participation factor: This factor could be a fixed value determined by the main grid operator, it could be set in a market like environment where the Microgrid could act as a resource offering regulation capacity for a certain price, or it could be determined by the Microgrid itself comparing the costs of producing secondary regulation power to a previously known secondary regulation energy price. However, all these possibilities depend on how the main grid operator operates the resources connected to it, and its discussion is out of the scope of this document.

9.5 Secondary Regulation Bids

In order to perform the economic optimization when sharing the MRE among the different resources in the Microgrid, the matcher algorithm uses secondary regulation bids provided by the resources willing to participate in the secondary regulation control.
Each bid contains a set of discrete power steps, each one associated to an incremental price [1]; this way of representing economical behaviour of resources allows describing any kind of bid function ranging from linear to quadratic or discrete functions. These bids can be updated at any time by the resources in the Microgrid depending on their own situation and decisions.

The policies that each DER will use when creating secondary regulation bids is further discussed in chapter 9.7. An example of a bid is shown in next figure:

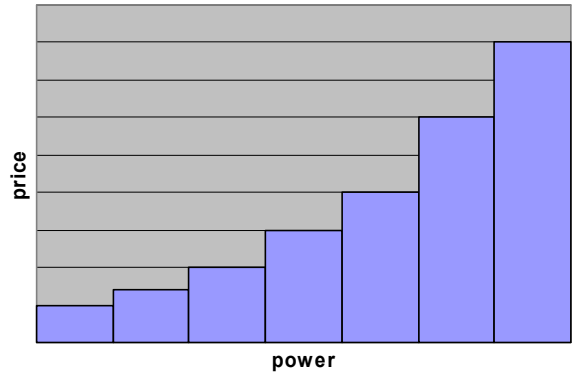


Figure 8-1: Example of a secondary regulation bid

Devices can mark their bids as indivisible, this means that the blocks can not be divided and must be assigned complete by the secondary regulation matcher.

9.6 Secondary Regulation Matcher algorithm flow chart

The flow chart of the secondary regulation algorithm is presented in Figure 8-2 and Figure 8-3; next lines describe step by step the procedure followed by the algorithm.
This algorithm is executed periodically, the time period should be adjusted depending on the reaction time of the devices and the frequency of the deviations in the Microgrid, in any case, this value should be adjusted after testing the Microgrid system where the control is going to be executed.
The first step is to obtain static data; this data does not change continuously:

- Power exchange schedule at the point of common coupling P_{sch}^{ppc} .
- Reference frequency to be maintained by the algorithm fr_{ref} .
- Participation factor to be used in case of the Microgrid is operating in grid connected mode RPF (Regulation Participation Factor).
- Set of frequency versus power droop characteristics for the generating units in the Microgrid $\left[R_1, R_{2,} \cdots, R_n\right]$.
- Set of frequency versus power characteristics for the non-conforming loads in the Microgrid $\left[D_1, D_{2,} \cdots, D_l\right]$.

The second step consists in obtaining the measured power value at the point of common coupling P_{meas}^{ppc} and the system frequency measurement fr_{meas}.
Then the algorithm calculates the MRE as explained in chapter 9.4, if the MRE is greater than a configured tolerance the algorithm continues its execution, otherwise, it waits until next regulation period.

When the MRE is significant, the algorithm obtains the set of measured powers $\left[P_1^{meas}, P_2^{meas}, \cdots, P_m^{meas}\right]$ for all devices participating into the secondary regulation control, together with the bids $\left[SRB_1, SRB_{2,} \cdots, SRB_m\right]$ and the power schedules $\left[P_1^{sch}, P_2^{sch} \cdots, P_m^{sch}\right]$ associated to each device.

Finally, with the obtained data, an algorithm that matches the bids and the power to be covered is executed. This algorithm generates new set points for controlled devices; these new set points are the basis for creating new power schedules that are sent to the corresponding device local controller.

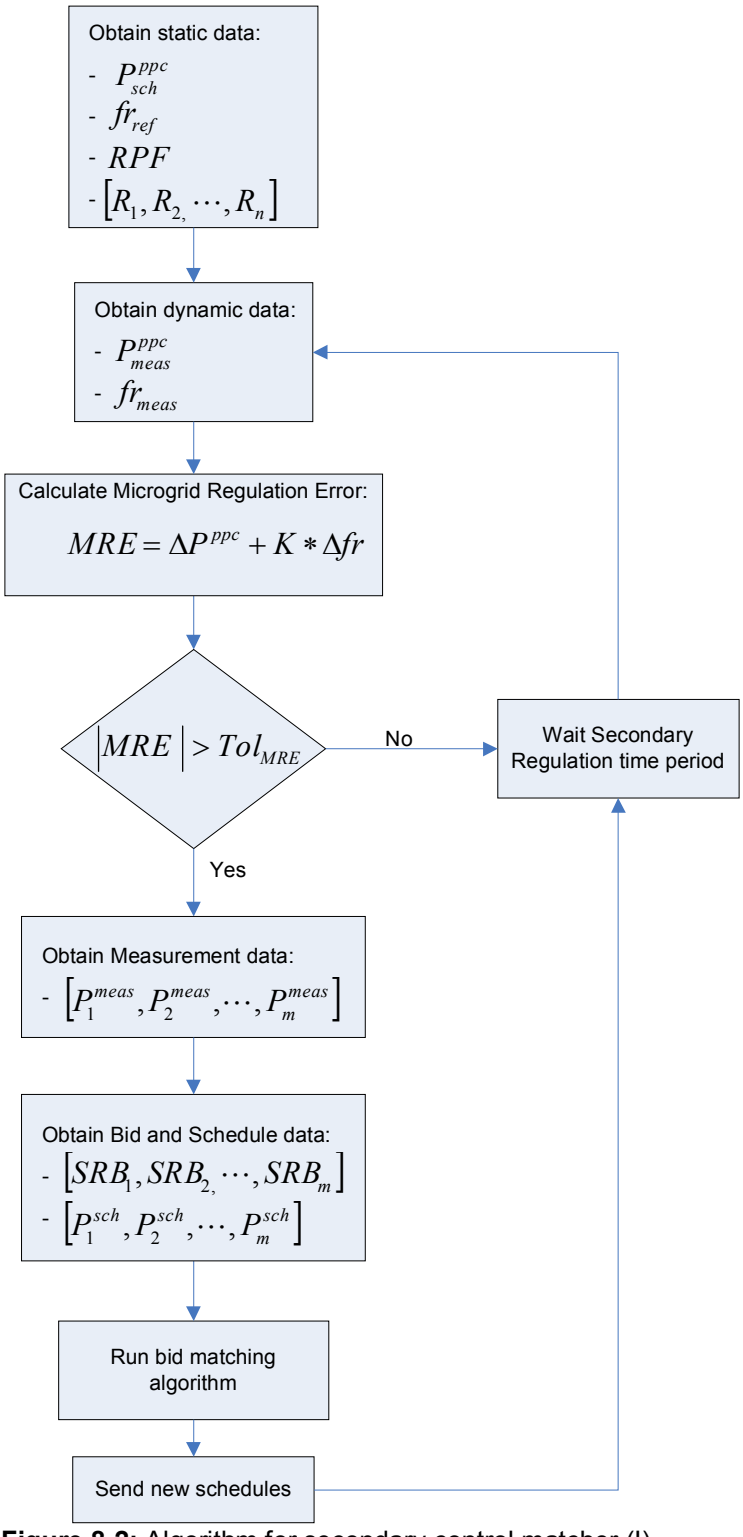


Figure 8-2: Algorithm for secondary control matcher (I)

The goal of the bid matcher algorithm is to share the MRE among the DER devices ensuring the economical operation of the Microgrid. The algorithm's flow chart can be seen in Figure 8-3.

The algorithm needs the set of bids that represent the incremental prices of the power being offered, the economic minimization consists on determining the point where all the incremental prices for each curve are equal while all the power is covered. For this purpose bids must have an increasing convex shape.

The algorithm starts by calculating the total measured power for all controllable devices $P_{tot}^{meas} = \sum_{1}^{m} P_i^{meas}$, the MRE is added to the total measured power that gives the total amount of power to be shared among the controllable resources $P_{tot}^{cover} = P_{tot}^{meas} + MRE$.

Next, the algorithm starts an iterative process that selects the cheaper power block based on devices' bids. If the total assigned power plus the power in the selected bid $P_{tot}^{assign} + P^{select}$ is lower than the total power to be covered minus a tolerance value $P_{tot}^{cover} - Tol^{cover}$, the algorithm assigns the selected bid and continues iterating by selecting the next cheaper bid. Otherwise, the algorithm checks if the total assigned power plus the power in the selected bid $P_{tot}^{assign} + P^{select}$ is greater than the total power to be covered plus a tolerance value $P_{tot}^{cover} - Tol^{cover}$. If that is the case, the algorithm checks for the indivisibility of the selected bid, otherwise the algorithm ends the iteration process.

When checking for indivisibility condition, if the selected bid is indivisible, the algorithm does not assign it and continues the iterating process by selecting the next cheaper bid, otherwise the bid is curtailed $P_{last}^{assign} = P_{last}^{bid} - \left(P_{tot}^{assign} - P_{tot}^{cover}\right)$ and the iteration process ends.

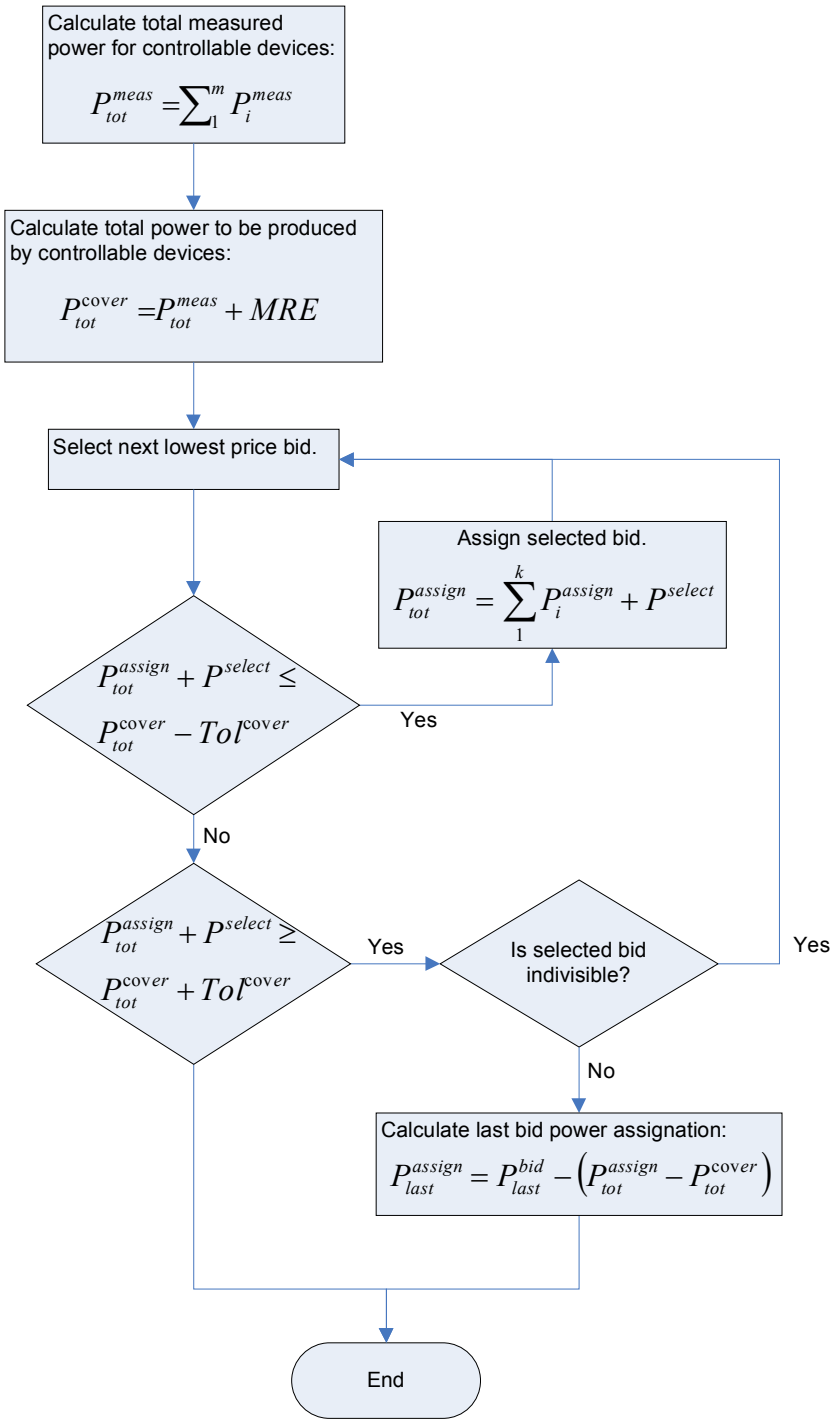


Figure 8-3: Algorithm for secondary control matcher (II)

9.7 Secondary regulation local bid creation policies

This chapter discusses the policies that different devices could adopt in order to generate the bids that they send to the Secondary Regulation Matcher. The flexibility of the decentralized system allows developing different algorithms to deal with different situations and needs from devices.

The Secondary Regulation Matcher will hold a pool of bids, for each secondary regulation algorithm execution the matcher will consider all the bids in that pool. Devices can update and send new bids as needed; these new bids will update already

existing bids in the pool held by the matcher. This way, those new and updated bids will be taken into account by the secondary regulation matcher algorithm. This approach for communicating bids allows reducing the amount of data to be communicated comparing to a polling approach where the matcher should ask for new bids each time it executes the algorithm.
The devices that will contribute to the Secondary Regulation control by sending bids will be controllable generators, storage systems, loads and the Main Grid (see chapter 9.2). Following sub-chapters discuss different policies that could be taken into account when designing and developing the algorithms for creating those bids.

9.8 Policies for creating generators' bids

Those generation systems that are fast enough to react to power set points changes in a range of seconds will be able to contribute to Secondary Regulation as adjustable power production units. The incremental price of their bids will be based on their operating costs; this is usually dependant on the amount of fuel consumed that increases following a quadratic curve with the power being produced.
Typical cost curve for a 50 kW micro-turbine generator is:

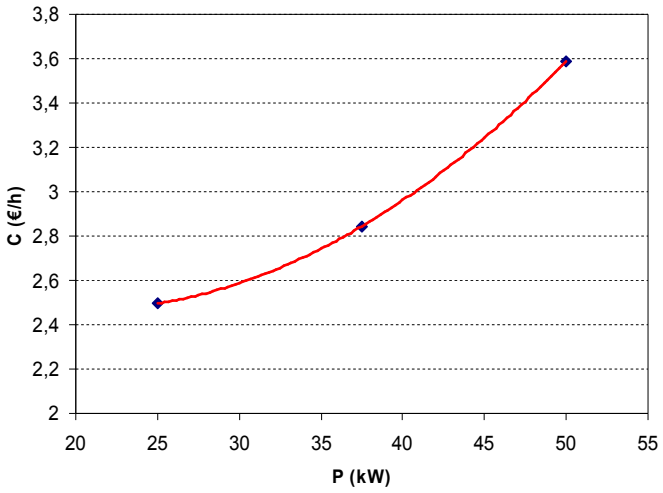


Figure 8-4: Typical cost curve for a micro-turbine

The incremental cost curve is calculated by differentiating the cost curve, this incremental cost curve will be divided into several steps and used as bid for secondary control regulation. See next figure for a typical bid based on an incremental cost curve:

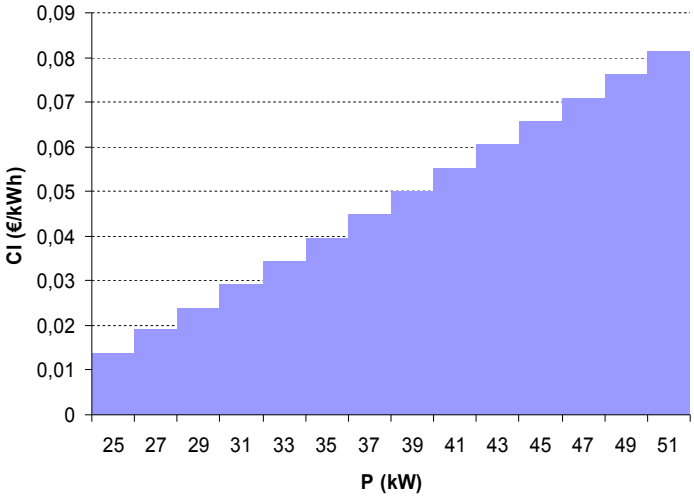


Figure 8-5: Typical secondary regulation bid for a micro-turbine

The owner of the generator can modify these bids on his own, for example, he can add a certain benefit margin to the cost curve or he can foresee secondary regulation prices and adjust its curve in order to be matched at the highest possible price, etc.

9.9 Policies for creating storage systems' bids

The prices to be offered by storage systems depend on the amount of energy stored, that is, when the storage system is empty, it will offer consumption power at very low price since it is willing to be recharged. On the contrary, when the storage system is full, it will offer very low prices for electricity. Finally, when the battery is in an intermediate state of charge it will offer low prices bids for consumption and also for production because it has no operating costs.
Following figure shows an example of a set of bids corresponding to different state of charge of a battery system:

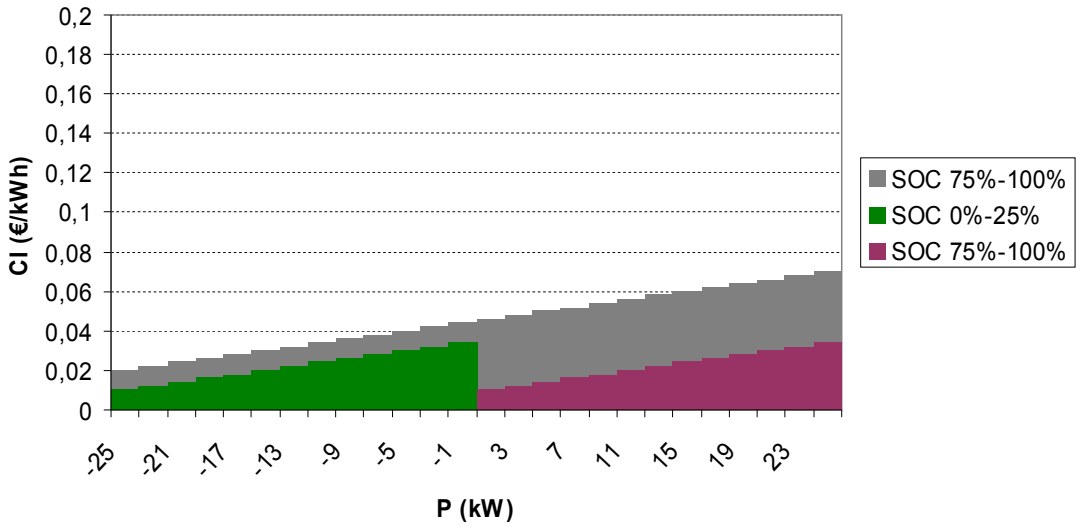


Figure 8-6: Example of a battery system bid

This way, the storage system will tend to increase its energy when its stored energy is low and will produce power when its stored energy is high. The device will change its bids in real time according to its specific situation.

9.10 Policies for creating controllable loads' bids

In general, loads will offer high price power reduction bids because a delay or reduction of consumed load has a direct incidence in the consumer. Anyway, in case of particular states like it would happen in Microgrid islanded working mode, the use of the power reduction bids from loads could be very beneficial for the stability of the Microgrid. For some critical situations, the secondary regulation control could not be fast enough and automated load shedding techniques should be used.
An example of a controllable load bid is found below:

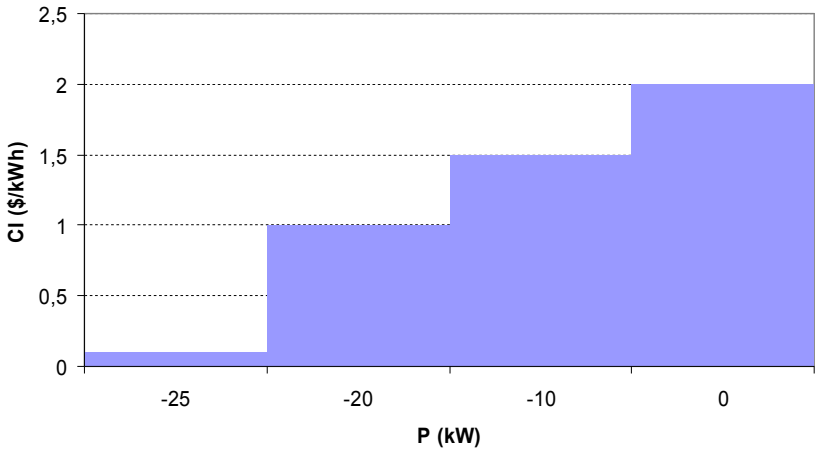


Figure 8-7: Example of a controllable load bid

9.11 Policies for creating Main Grid bids

When working in Grid connected operating mode, the power exchange with the Main Grid will be fixed after market contractual agreements with the system market operator and the distribution system operator. So the secondary regulation will try to maintain that fixed schedule, in case that in real time more or less power than the scheduled one is needed by the Microgrid, the Microgrid will have to pay some kind of penalty for that deviation. It could be that in some situations its worth to pay those deviation costs because it is still economically beneficial due to high costs of the generating units in the Microgrid. In these cases the Main Grid could participate into the Secondary Regulation system by offering bids with high prices that represents the costs of deviating from the scheduled power exchange with the Main grid.

A Main Grid bid should represent the cost of deviating from the contracted power schedule, an example of this kind of bid is shown below:

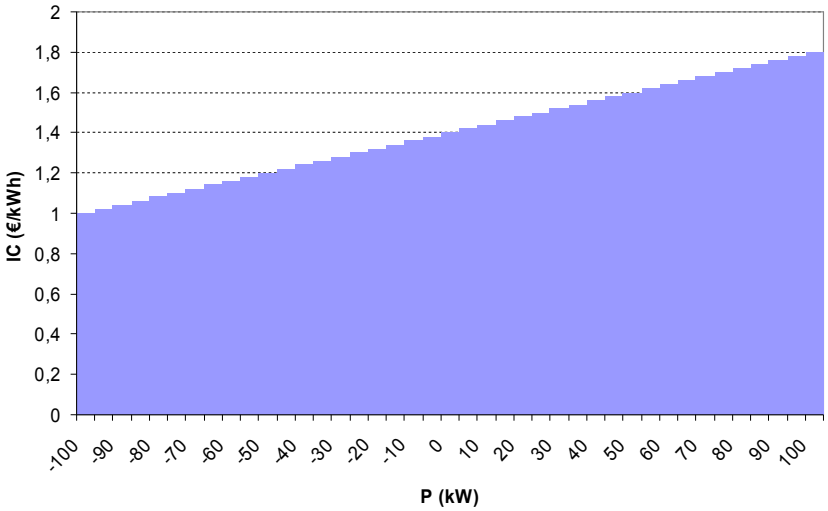


Figure 8-8: Example of a Main Grid bid

The main grid can not be dispatched directly; when all other devices are dispatched the exchanged power with the main grid will also vary accordingly, matching the desired power exchange.

9.12 System configuration

In order Secondary Regulation control to work properly, special care must be taken with the Microgrid configuration regarding the characteristics of the devices installed and their working modes.

System configuration in Main Grid connected mode

In case the Microgrid is operating in Grid Connected operating mode, the Main grid will act as an infinite voltage source and will regulate frequency and voltage on the Microgrid, the production devices in the Microgrid can be configured as current sources with or without droops.
In case of production units working as current sources without droops, these will be given output power set points by the secondary control system. In case some generating unit is working as a current source with droops, it will vary its power output with the frequency imposed by the main grid helping some way to regulate system frequency, the secondary regulation control will give power set points to these devices' local controllers that in turn will have to translate them to power offset set points.
The following figure shows an example of how to obtain the power P_2 required by the secondary regulation system by means of the power offset set point $P_{offset} = P_2 - P_1$ delivered to the device. In this case the initial and final frequencies are the ones set by the main grid.

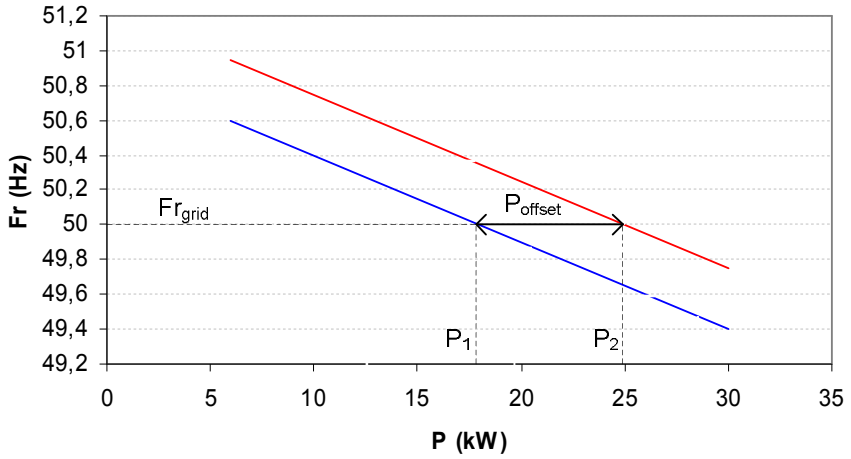


Figure 8-9: P_{offset} set point for a current source in grid connected mode

For this operating mode, there will be devices with fixed schedules; with the restriction that there must be enough devices configured as adjustable devices in order to cope with the possible deviations in generation and consumption.

System configuration in Islanded mode

When working in Islanded mode, there must be enough devices acting as voltage sources so they can handle instantaneous load changes. As in the previous case, devices working as current sources with droops will help to reduce the frequency error in the system. The set points for the current sources without droops will be power set points while for those working with droops will consist on frequency offset set points.
Figure 8-10 shows the power offset set point to be calculated for producing P_2 power at a given reference frequency Fr_2.

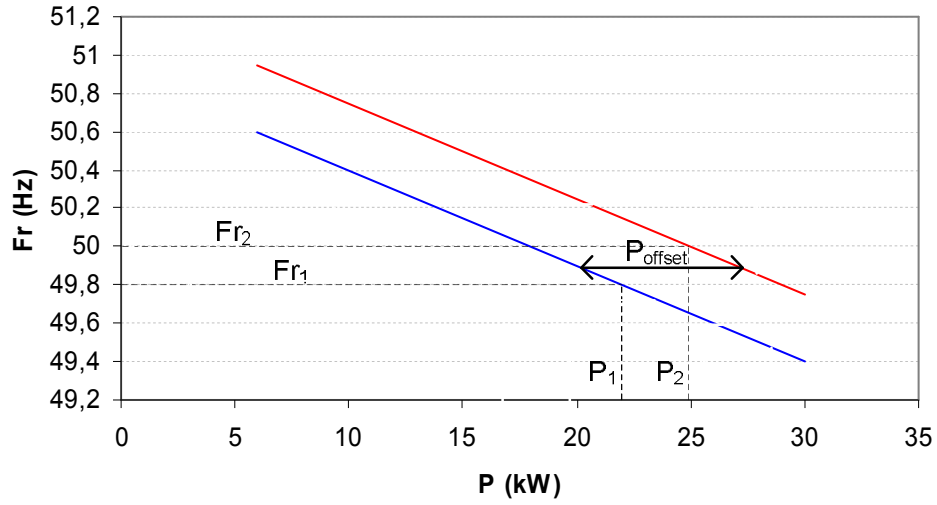


Figure 8-10: P_{offset} set point for a current source in islanded mode

The formula for calculating the power offset set point is: $P_{offset} = \frac{1}{R} \cdot (Fr_1 - Fr_2) + (P_1 - P_2)$ where Fr_1 is the initial frequency, P_1 the initial power, R is the P-Fr droop, Fr_2 the reference frequency and P_2 the power set point given by the secondary regulation control.
The devices working as voltage sources will always work with droops, in case they participate into secondary regulation control, they will be given power set points that they will translate into frequency offsets in order to match the required power output at the reference frequency.
Next figure shows an example of how to moving the droop in order to obtain the desired output power P_2 at the reference frequency Fr_2:

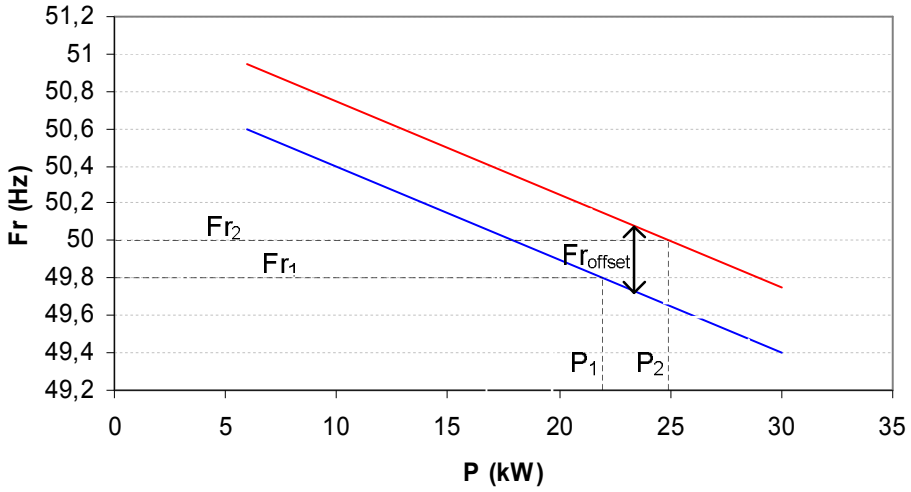


Figure 8-11: Fr_{offset} set point for a voltage source in islanded mode

For a voltage source the formula for calculating the frequency offset set point is: $Fr_{offset} = R \cdot (P_1 - P_2) + (Fr_1 - Fr_2)$ where Fr_1 is the initial frequency, P_1 the initial power, R is the P-Fr droop, Fr_2 the reference frequency and P_2 the power set point given by the secondary regulation control.
As same as in main grid connected case, the Microgrid must have enough available secondary regulation power in order to match the possible real time power deviations.

10. Agent Based Secondary Regulation Implementation Approach

This chapter describes in detail how to implement the algorithms described in chapter 8 by using intelligent software agent technologies. Software agents have been considered very well suited for this kind of applications characterized by the following aspects:

- Legacy systems integration.
- Need for plug and play capabilities.
- Distributed systems.

10.1 Introduction to Agent Systems

A multi-agent system is a distributed computational system where software pieces called agents, collaborate among them to reach certain goals.
An agent can be considered as an autonomous, proactive and collaborative computing system:

- Autonomous: By autonomous we mean that little or no human intervention is required for the agent to work.
- Proactive: proactive means that it is able to act according to certain objectives and not only to act in response to external events.
- Collaborative: collaborative agents should be able to communicate with other agents, asking for services offered by the other agents or responding to requests made by them.

According to their characteristics, Multi Agent Systems fulfil the requirements for implementing the secondary regulation control system described in previous chapters.
The methodology described in [7] has been taken as basis for describing the agent system. This methodology defines the agents and their communications following a step by step approach where the first step is an overview of the agent system and next steps detail the system in a progressive way.

10.2 JADE Overview

JADE is a software middleware that allows an easy development and deployment of a multi agent system [7] [8]. This tool provides all the software infrastructure and common functionality that allow developers to focus on the application logic and ontology definitions. Furthermore, it also provides the tools needed for agent system deployment, debugging and monitoring.
JADE has the following characteristics:

- Developed completely in JAVA, supports portability for a wide set of Operating Systems (Windows, Linux, Solaris…) and hardware systems (ranging from PC-s to PDA-s and resource constrained devices).
- Complies with FIPA (Foundation for Intelligent Physical Agents) standards [9]. FIPA is the most used set of standards for implementing Multi Agent basic architecture and communication aspects.
- It is a well known agent platform which work is continuously improving by adding new features and technologies to the agent platform.
- JADE is an open source tool distributed under the LGPL license (Lesser General Public License Version 2).

- Development of agent systems for JADE is well documented with several tutorials, methodologies etc.

JADE provides functionalities for the basic features an agent platform should have. These features are implemented by JADE in two ways: JADE services and JADE built-in agents. Additionally, JADE provides an API (Application programming Interface) with a set of classes that makes easier the development of a Multi Agent System.
JADE built-in agents:

- Agent Management Service (AMS): The AMS service is in charge of maintaining a directory of AID-s (Agent IDentifier) unique for each agent within the Agent Platform, and which contains transport addresses (amongst other things) for agents registered with the Agent Platform. Each agent must register with an AMS in order to get a valid AID.

  The AMS offers white pages services to other agents based on its agent's registry. The AMS is also responsible for managing the operation of an Agent Platform, such as the creation, deletion and migration of agents to and from the platform. It also maintains the life cycle associated with each agent.
- Directory Facilitator (DF): The DF service provides yellow pages services to other agents. Agents may register their services with the DF and query the DF to find out what services are offered by other agents or which agents offer certain services.

The most important JADE service is the Message Transport Service:

- Message Transport Service (MTS): The MTS is in charge of delivering messages between agents within the same Agent Platform and to agents that are running on other Agent Platforms. FIPA specifies the requirements for the MTS regarding the communications between agents in different agent platforms (IIOP, HTTP based communications), this way the inter-operability between agent platforms is maintained. JADE uses RMI for communication between agents in the same platform.

Following figure shows JADE architecture:

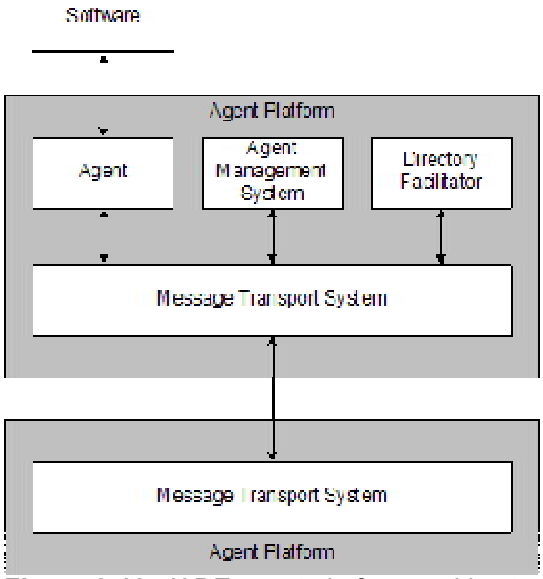


Figure 8-12: JADE agent platform architecture

10.3 Use Cases

Use case models present an overview of the functionality provided by a system in terms of actors, functions represented as use cases and any dependencies between those use cases.

The following UML (Unified Modelling Language) [9] use case diagram defines how the secondary regulation algorithm is divided into functions; next chapters will show the agents performing each function and how to implement them in detail:

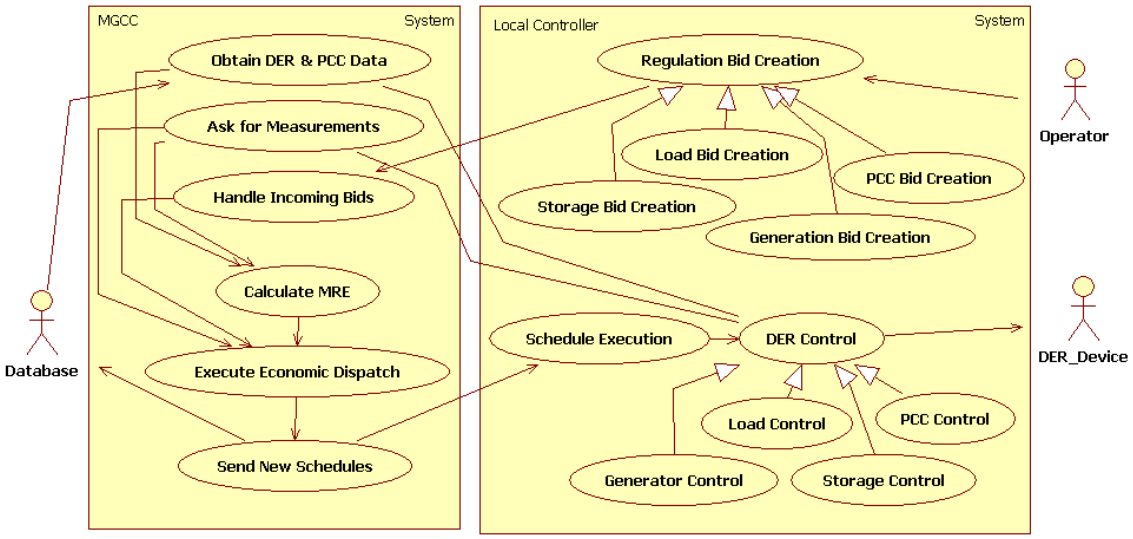


Figure 8-13: Use case diagram

Three different participating actors have been identified, the database will be in charge of storing in a persistent way data regarding PCC power exchange schedule and schedules corresponding to each DER device. The DER Device actor will receive the power set points from the secondary regulation control system and will provide measurement data. Finally, the Operator actor will give its input to Regulation Bid

Creation use case in order to set its preferences for participating into the secondary regulation control system.
As a decentralized distributed system, the control is divided into two different subsystems, the one in the Microgrid Central Controller and the part executed in the different Local controllers, each local controller system will be attached to a certain DER device.
Following table describes the use cases corresponding to the Microgrid central controller system:

Table 8-I: Use case table for the MGCC system

Use case	Description	Relationships
Obtain DER & PCC Data	Obtains the power schedule at the PCC, reference frequency, participation factor and the droop characteristics for controllable devices.	"DER control" use cases provides DER data. The Database provides schedule data.
Ask for Measurements	Frequency and power measurements are asked to the corresponding DER controllers.	Obtains measurements from "DER Control" use cases
Handle Incoming Bids	Handles incoming bids by updating them so they can be used in the control algorithm.	"DER Control" use cases update their secondary regulation bids as needed.
Calculate MRE	Calculates the Microgrid Regulation Error to be compensated by the control algorithm.	Receives input from "Obtain DER & PCC Data" and "Ask for Measurements" use cases. Provides output to "Execute Economic Dispatch" use case.
Execute Economic Dispatch	Executes the economic dispatch algorithm in order to obtain the set points covering the MRE and economically optimizing DER resources dispatch.	Receives input from "Ask for Measurements", "Handle incoming Bids" and "Calculate MRE" use cases. Provides output to "Send New Schedules" use case.
Send New Schedules	Sends secondary regulation schedules to devices and the database.	Receives input from "Execute Economic Dispatch" use case. Sends schedules to "Schedule Execution" use case and Database actor.

Next table describes the use cases corresponding to local controller systems:

Table 8-II: Use case table for Local Controller systems

Use case	Description	Relationships
Regulation Bid Creation	Creates according to the local controller's specific policy the bids to be used for secondary regulation power assignation.	Provides secondary regulation bids to "Handle Incoming Bids" use case.
Generation Bid Creation	Creates specific bids for the generating unit being controlled.	Extends "Regulation Bid Creation" use case.
Load Bid Creation	Creates specific bids for the controllable load being controlled.	Extends "Regulation Bid Creation" use case.
Storage Bid Creation	Creates specific bids for the storage system being controlled.	Extends "Regulation Bid Creation" use case.
PCC Bid Creation	Creates specific bids for the PPC	Extends "Regulation Bid Creation" use case.
Schedule Execution	Executes power schedules by sending control commands to "DER Control" use case.	Receives input from "Send New Schedules" use case. Provides power set points to "DER Control" use case.

Use case	Description	Relationships
DER Control	Acts as a gateway between the control system and the DER physical device.	Receives input from "Schedule Execution" use case. Gives output to DER Device actor. Provides information for "Obtain DER & PCC Data" and "Ask for Measurements" use cases.
Generator Control	Specific implementation for controlling Generating units.	Extends "DER Control" use case.
Load Control	Specific implementation for controlling Loads.	Extends "DER Control" use case.
Storage Control	Specific implementation for controlling Storage Systems	Extends "DER Control" use case.
PCC Control	Specific implementation for managing PCC power exchange.	Extends "DER Control" use case.

10.4 Agent Definition

This chapter defines the agents needed in order to implement the functions described in the use cases, see chapter 10.3.
The criteria for identifying the different agents are:

- Agents must be modular and reusable, functions used by several parties are implemented in different agents
- Functions that require the same data will be integrated into a single agent.
- Too many agents increase the overall system complexity.
- Big and complex agents are difficult to design and maintain.

Equilibrium must be maintained between having a system with lot of very simple agents and having a system with few complex agents.
Following these criteria, next table shows the agents identified for implementing secondary regulation functions:

Table 8-III: Agent identification table

Agent	Description
Database Gateway Agent	This agent acts as gateway between the database and the agent system. This agent has access to the tables in the database using SQL for making database queries. Any access to the database by any agent in the system is done through the Database Gateway Agent.
Regulation Matcher Agent	Executes all functions regarding MRE calculation and economic Dispatch calculations. Performs use cases: "Obtain DER & PCC Data", "Ask for Measurements", Handle Incoming Bids", "Calculate MRE", "Execute Economic Dispatch" and "Send New Schedules".
DER Gateway Agent	Generic agent aimed to act as interface between a particular DER device and the rest of the agent system.
Generator Gateway Agent	Specialization of the DER Gateway Agent. Is the gateway between a certain generator and the agent system, this agent hides the particularities that are specific of the generator that it is attached to it and offers a set of uniform generator services to the agents in the system.
Load Gateway Agent	Specialization of the DER Gateway Agent. Is the gateway between a certain controllable load and the agent system, it is able to obtain measurement and status information and also to send on and off control commands to the loads attached to it.

Agent	Description
Storage Gateway Agent	Specialization of the DER Gateway Agent. The same as the other DER gateway agents, this agent gives a set of uniform measurement and control services to the agents in the platform for managing storage devices.
PCC Gateway Agent	Specialization of the DER Gateway Agent. Manages PCC measurement and status (Main Grid connection and disconnection).
Schedule Tracker Agent	It is in charge of executing power schedules; it sends the set points in those schedules to DER Gateway Agent at the corresponding schedule time.
DER Regulation Bidder Agent	Creates bids considering user and device's status and sends to the agent performing the matching algorithm.
Generator Regulation Bidder Agent	Specialization of the DER Regulation Bidder Agent. This bidder agent will be in charge of creating the bids according to the generators cost curve.
Load Regulation Bidder Agent	Specialization of the DER Regulation Bidder Agent. The Load Bidder Agent will create the bids corresponding to the loads being controlled.
Storage Regulation Bidder Agent	Specialization of the DER Regulation Bidder Agent. This agent is in charge of creating the bids for participating into the secondary regulation control according to the state of charge of the storage system.

The following figure shows an overview of the agent architecture and deployment, note that there will be as much Local Controllers with their corresponding agents as devices in the Microgrid:

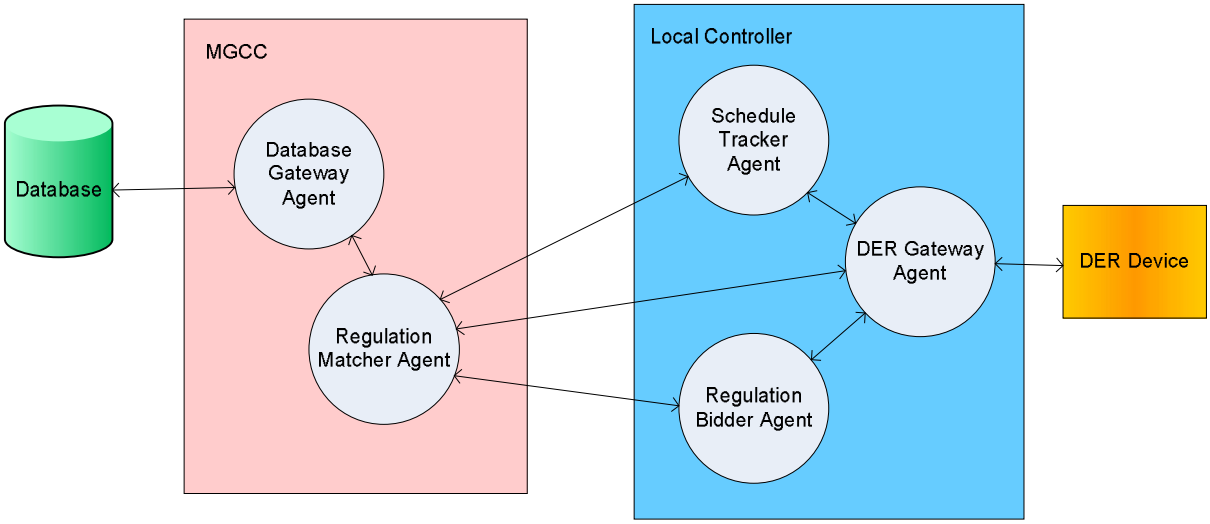


Figure 8-14: Agents for secondary regulation control

Following tables describe in detail the roles and responsibilities for each of the agents from Table 8-III.

Table 8-IV: MGCC agents' responsibility table

Agent	Responsibilities
Database Gateway Agent	• Manage connections with the database. • Registers itself in the DF providing database management services. • Translates received ACL (Agent Communication Language) request to SQL queries and executes them. Listens for measurement, status, schedules, and commitment add and browse requests. • Returns query results to the requesting agent in ACL

Agent	Responsibilities
Regulation Matcher Agent	• Searches in the DF for the agent offering database management service. • Searches in the DF for the agents offering generation, load or storage control services. • Searches in the DF for the agent offering PCC control service. • Select the agent providing frequency measurement. • Asks the Database Gateway Agent for the schedule of the PCC power exchange. • Listens for secondary regulation bids. • Asks PCC agent for power measurement. • Asks the agent providing frequency measurement for its measured frequency. • Asks for power measurements to the agents providing secondary regulation bids. • Calculates MRE and executes economic dispatch algorithm if required. • Asks the tracker agents corresponding to the devices providing secondary regulation bids to execute calculated new power schedules.

Table 8-V: LC agents' responsibility table

Agent	Responsibilities
Generator Gateway Agent	• Publish device characteristics into DF: Device identifier, Pmax, Pmin, Qmax, Qmin, R (droop). • Publish measurement and status capabilities into DF: P, Q, V, Fr, Started/Stopped, Droops ON/OFF, Operating Mode voltage source/current source. • Publish control capabilities into the DF: P set point, Q set point, V set point, Fr set point, Start/Stop, Droop ON/OFF, Operating Mode voltage/current source. • Receive measurement and status data requests and return the corresponding values: P, Q, V, Fr, Started/Stopped, Droops ON/OFF, Operating Mode voltage source/current source. • Receive device operation requests and execute them: P set point, Q set point, V set point, Fr set point, Start/Stop, Droop ON/OFF, Operating Mode voltage/current source. • Translate P and Q set points to P_offset, Q_offset, Fr_offset or V_offset depending on device's operating mode (voltage or current source) and droop status (activated or not).
Load Gateway Agent	• Publish device characteristics into DF: Device identifier, Prated, Qrated, D (power versus frequency variation). • Publish measurement and status capabilities into DF: P, Q, V, Fr, ON/OFF. • Publish control capabilities into the DF: ON/OFF. • Receive measurement and status requests and return the corresponding values: P, Q, V, Fr, ON/OFF. • Receive operation requests and execute them: ON/OFF.
Storage Gateway Agent	• Publish device characteristics into DF: Device identifier, Pmax, Pmin, Qmax, Qmin, R (droop). • Publish measurement capabilities into DF: P, Q, V, Fr. • Publish status capabilities into the DF: Started/Stopped, Droops ON/OFF, Operating Mode voltage source/current source. • Publish control capabilities into the DF: P set point, Q set point, V set point, Fr set point, Start/Stop, Droop ON/OFF, Operating Mode voltage/current source. • Receive measurement and status data requests and return the corresponding values: P, Q, V, Fr, Started/Stopped, Droops ON/OFF, Operating Mode voltage source/current source. • Receive device operation requests and execute them: P set point, Q set point, V set point, Fr set point, Start/Stop, Droop ON/OFF, Operating Mode voltage/current source. • Translate P and Q set points to P_offset, Q_offset, Fr_offset or V_offset depending on device's operating mode (voltage or current source) and droop status (activated or not).

Agent	Responsibilities
PCC Gateway Agent	• Publish measurement and status capabilities into DF: P, Q, V, Fr, Connected/Disconnected from the main grid. • Publish control capabilities into the DF: Connect/Disconnect from the main grid. • Receive measurement and status data requests and return the corresponding values: P, Q, V, Fr, Connected/Disconnected from the main grid • Receive operation requests and execute them: Connect/Disconnect from the main grid.
Schedule Tracker Agent	• Registers itself into DF offering schedule tracking services. • Searches in the DF for the agent offering control services for the controlled device. • Receives generation schedule execution requests. • Sends the corresponding set points according to the schedule to the device control agent.
Generator Regulation Bidder Agent	• Searches in the DF for the DER Gateway Agent controlling the device. • Searches in the DF for the Regulation Matcher Agent. • Creates secondary regulation bid according to user preferences and device's needs. • Each time a new secondary regulation bid is created sends it to the Regulation Matcher Agent.
Load Regulation Bidder Agent	• Searches in the DF for the DER Gateway Agent controlling the device. • Searches in the DF for the Regulation Matcher Agent. • Creates secondary regulation bid depending on the user preferences and device's needs. • Each time a new secondary regulation bid is created sends it to the Regulation Matcher Agent.
Storage Regulation Bidder Agent	• Searches in the DF for the DER Gateway Agent controlling the device. • Searches in the DF for the Regulation Matcher Agent. • Creates secondary regulation bid depending on the state of charge of the device. • Each time a new secondary regulation bid is created sends it to the Regulation Matcher Agent.
PCC Regulation Bidder Agent	• Searches in the DF for the DER Gateway Agent controlling the device. • Searches in the DF for the Regulation Matcher Agent. • Creates secondary regulation bid depending on the deviation penalty prices. • Each time a new secondary regulation bid is created sends it to the Regulation Matcher Agent.

10.5 Agent Interactions

The interactions between the agents in the system can be included within three different main processes:

- Bid creation and notification: Consists on the creation of new secondary regulation bids and notification to the Secondary Regulation Matcher of those bids. It will also handle the possibility to remove a bid from the pool of bids held by the Secondary Regulation Matcher Agent.

- Secondary Regulation Matching algorithm execution: Execution of the secondary regulation matcher algorithm after all required data is collected. The resulting new schedules are sent to schedule tracker agents and the database for accounting purposes.

- Autonomous schedule tracking: The schedules are executed in real time by sending the corresponding set points at scheduled times.

This chapter will show the agent interactions together with the messages exchanged between the agents defined in chapter 10.4 regarding the processes mentioned above. These processes include all the use cases described in chapter 10.3.

Bid creation and notification

Bid creation and notification deals with use case "Regulation Bid Creation" and is related with "Handle Incoming Bids" and "DER Control" use cases. The agents that take part in this process are all "DER Regulation Bidder Agents", "Regulation Matcher Agent" and JADE's "DF". Following figure shows an UML sequence diagram describing the agents and the message flows between them:

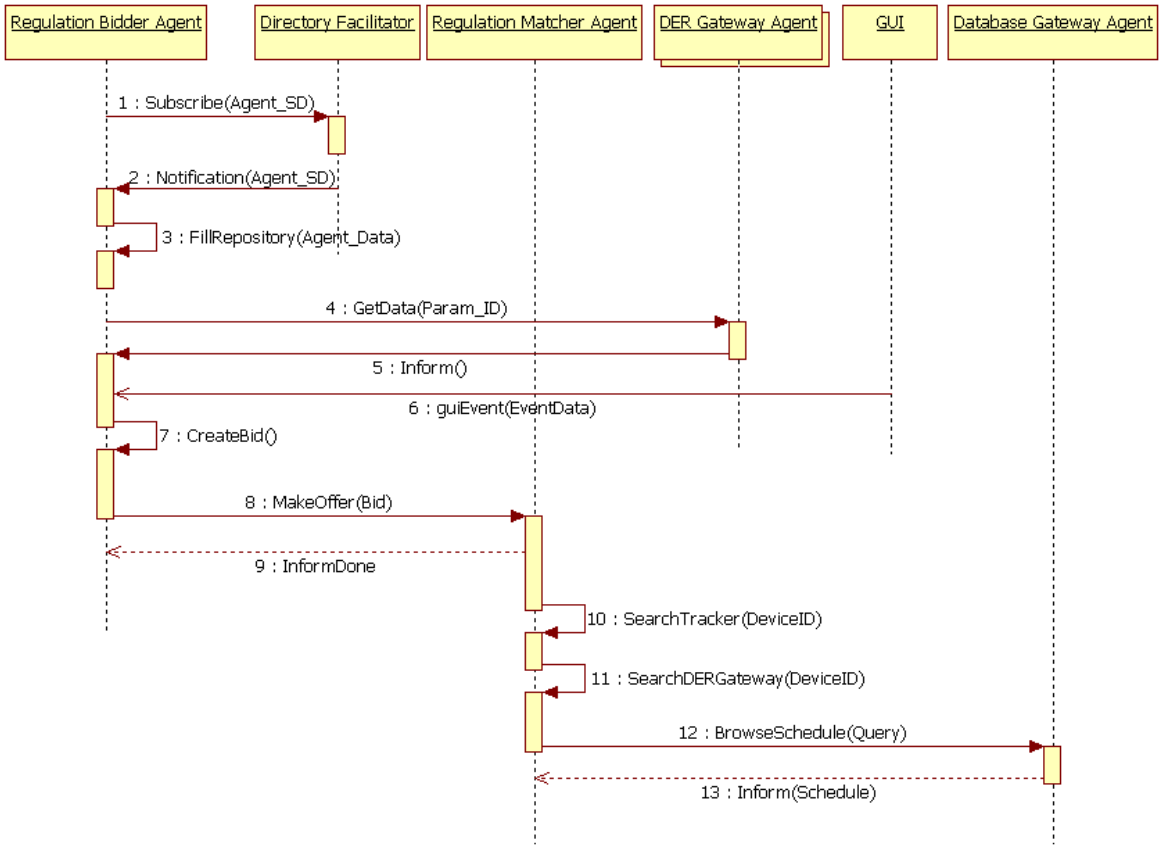


Figure 8-15: Bid creation and notification process sequence diagram

The steps shown in Figure 8-15 are described in detail in the following table:

Table 8-VI: Steps in bid creation and notification process

Step	Name	Data	Description	Producer to receiver
1	Subscribe	Template of the Service Description corresponding to the DER Gateway Agent controlling the device.	The agent subscribes into the DF in order to obtain the reference to the agent performing device gateway services.	Regulation Bidder Agent to Directory Facilitator.

Step	Name	Data	Description	Producer to receiver
2	Notification	Agent Service description with all data registered or modified by the agent that fired the notification. Contains data about DER characteristics and status.	The Regulation Bidder Agent is notified when any agent matching the previous subscription registers or changes its registration.	Directory Facilitator to Regulation Bidder Agent.
3	FillRepository	Agent reference for the DER Gateway Agent controlling the device.	Updates the internally stored reference to the DER Gateway Agent controlling the device.	Regulation Bidder Agent to Regulation Bidder Agent
4	GetData	Parameter identifier being monitored.	The agent monitors device's parameters. For example in case of a battery system the parameter to be monitored could be the state of charge of the battery.	Regulation Bidder Agent to DER Gateway Agent
5	Inform	Parameter value.	The agent receives monitored parameter and decides if a new bid must be created or not.	DER Gateway Agent to Regulation Bidder Agent
6	GuiEvent	Bid data, set by the DER operator.	DER owner sets bid data according to his willing to participate into secondary regulation system.	GUI to Regulation Bidder Agent
7	CreateBid	Configuartion data and data about controlled DER characteristics and status.	According to the DER characteristics, status and operator's input the Bidder Agent creates a new secondary regulation bid.	Regulation Bidder Agent to Regulation Bidder Agent.
8	MakeOffer	Bid containing a set of steps associated to DER device's incremental costs.	The Regulation Bidder Agent sends the created bid to the Regulation Matcher Agent.	Regulation Bidder Agent to Regulation Matcher Agent.
9	Inform	Success or failure indication.	Indication of reception and bid acceptance.	Regulation Matcher Agent to Regulation Bidder Agent.
10	SearchTracker	ID of the device corresponding to received bid.	Associates the bid to a Tracker Agent, this way the agent knows the receiver to which schedules must be sent after executing secondary regulation bid matching.	Regulation Matcher Agent to Regulation Matcher Agent.
10	SearchDERGatewa y	ID of the device corresponding to received bid.	Associates the bid to a DER Gateway Agent, this way the agent knows to which agent ask for measurements.	Regulation Matcher Agent to Regulation Matcher Agent.
13	BrowseSchedule	Query for retrieving the last schedule that corresponds to the device offering the bid.	The agent asks for the schedule that corresponds to the new arrived bid's device	Regulation Bidder Agent to Database Gateway Agent
14	Inform	Schedule obtained from the database that corresponds to the device offering the bid.	The agent stores the schedule that will be used when creating new schedules after the bid matching process.	Database Gateway Agent to Regulation Bidder Agent.

Secondary Regulation matching

The secondary regulation matching process covers use cases: "Obtain DER and PCC Data", "Ask for Measurements", "Handle Incoming Bids", "Calculate MRE", "Execute Economic Dispatch" and "Send New Schedules". Figure 8-16 shows the UML sequence diagram showing information flows between the different agents taking part in the process.

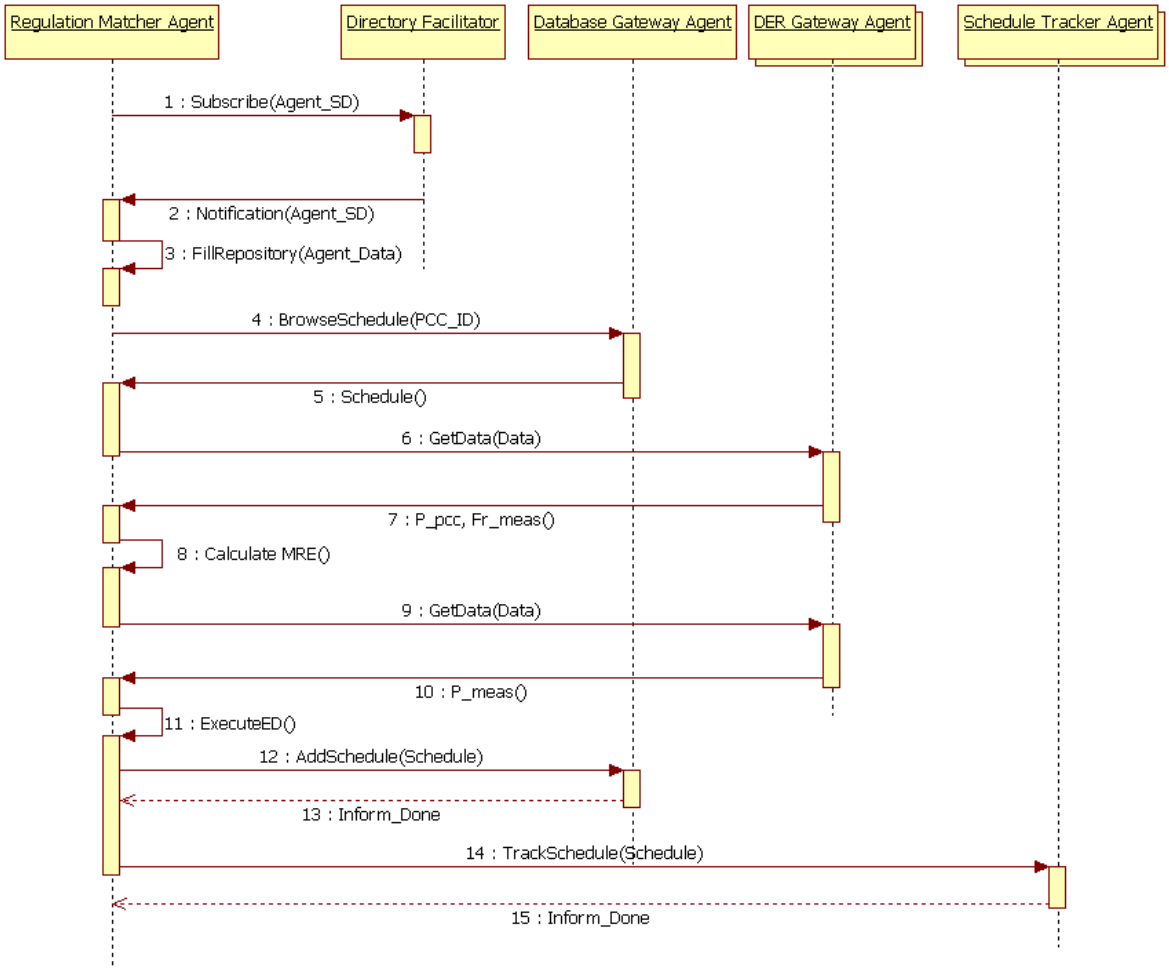


Figure 8-16: Secondary regulation matching process sequence diagram

The following table explains step by step the different information flows between the agents involved in the secondary regulation matching process.

Table 8-VII: Steps in secondary regulation matching process

Step	Name	Data	Description	Producer to receiver
1	Subscribe	Templates of the Service Descriptions corresponding to: • Agents offering database service. • Agents offering DER Gateway services. • Agents offering PCC gateway services.	The agent subscribes into the DF in order to obtain the reference to: • Database Gateway Agent. • DER Gateway Agents. • PCC Gateway Agent.	Regulation Matcher Agent to Directory Facilitator.

Step	Name	Data	Description	Producer to receiver
2	Notification	Service Descriptions of the following agents: • Agents offering database service. • Agents offering DER Gateway services. • Agents offering PCC gateway services.	The agent is being notified when new, agents are launched in the system and when agents are removed from it. This allows the Matcher agent to maintain an updated list of all the agents that it will collaborate with for performing its operations.	Directory Facilitator to Regulation Matcher Agent.
3	FillRepository	• Information about DER devices (R and D). • Reference to PCC Gateway Agent. • Reference to the agent providing frequency measurement. • Reference to the Database Gateway Agent.	The Regulation Matcher stores the information received on its own data repository for its later use.	Regulation Matcher Agent to Regulation Matcher Agent.
4	BrowseSchedule	Query asking for PCC schedule.	The Regulation Matcher Agent asks the database for the power exchange schedule at the PCC contracted with the main grid operator.	Regulation Matcher Agent to Database Gateway Agent.
5	Schedule	Power exchange schedule at the PCC.	The Database Gateway Agent returns the power schedule at the PCC.	Database Gateway Agent to Regulation Matcher Agent.
6	GetData	Parameters to be asked for: • Power measurement at the PCC. • System Frequency.	The Regulation Matcher Agent asks the PCC Gateway Agent and the agent selected as providing system frequency for the power measurement at the PCC and the system frequency.	Regulation Matcher Agent to PCC Gateway Agent and DER Gateway Agent.
7	Data	Measurements: • Power at the PCC. • System Frequency.	Regulation Matcher Agent obtains the measurements needed for MRE calculation.	PCC Gateway Agent and DER Gateway Agent. to Regulation Matcher Agent.
8	Calaculate MRE	Data for calculating MRE: • Measured power at the PCC. • Measured system frequency. • Droop and non conforming loads characteristics. • Participation factor (configuration). • Power schedule at the PCC.	Calculates the Microgrid Regulation Error, it is the power amount to be corrected by the secondary regulation control in order to restore frequency and maintain the power exchange schedule at the PCC.	Regulation Matcher Agent to Regulation Matcher Agent.
9	GetData	Measurements to be asked for: Power.	Asks the different DER Gateway Agents belonging to devices providing secondary regulation bids for their power measurements.	Regulation Matcher Agent to DER Gateway Agents
10	Data	Power measurements for all devices participating into the secondary regulation control.	DER Gateway Agents return their measured powers.	DER Gateway Agents to Regulation Matcher Agent
11	ExecuteED	MRE, bid data and measured powers.	Calculate the new power schedules in order to minimize MRE in an economically optimum way.	Regulation Matcher Agent to Regulation Matcher Agent.

Step	Name	Data	Description	Producer to receiver
12	AddSchedule	Schedules for the devices participating into secondary regulation control.	Ask the database to store the new set of schedules for the devices participating into secondary regulation control system.	Regulation Matcher Agent to Database Gateway Agent.
13	Inform	Indication that the request has been successfully processed or that a failure occurred.	The Database Gateway agent will store these new schedules in the database.	Database Gateway Agent to Regulation Matcher Agent.
14	TrackSchedule	Schedules for the devices participating into secondary regulation control.	Ask the Schedule Tracker Agents to track the corresponding schedules.	Regulation Matcher Agent to Schedule Tracker Agents.
15	Inform	Indication that the request has been successfully processed or that a failure occurred...	Indication of reception and schedule acceptance.	Schedule Tracker Agents to Regulation Matcher Agent

Autonomous schedule tracking

The autonomous schedule tracking mechanism implements use case "Schedule Execution". Its goal is to execute a given power schedule locally.

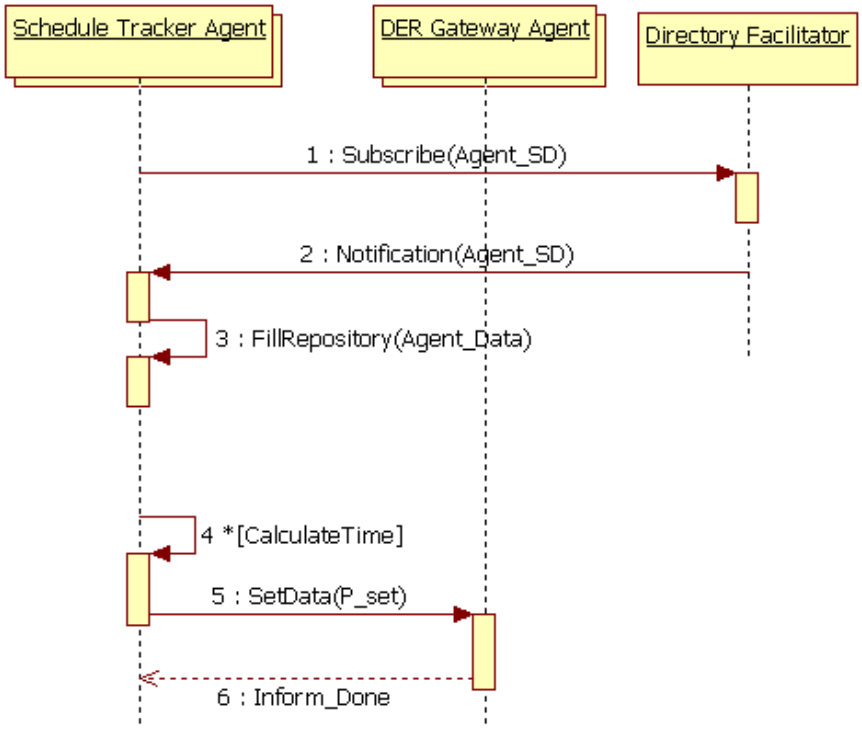


Figure 8-17: Autonomous schedule tracking process sequence diagram

The sequence diagram in above figure is explained in Table 8-VIII

Table 8-VIII: Steps in autonomous schedule tracking process

Step	Name	Data	Description	Producer to receiver

Step	Name	Data	Description	Producer to receiver
1	Subscribe	Template of the Service Description corresponding to the DER Gateway Agent controlling the device.	The Schedule Tracker Agent subscribes into the DF in order to obtain notifications when the DER Gateway Agent is launched or removed from the platform.	Schedule Tracker Agent to Directory Facilitator.
2	Notification	Service Description of the DER Gateway Agent controlling the Device.	The agent is notified when the DER Gateway Agent controlling the device is launched or killed. The reference to the gateway agent is used in order to set control commands to the device.	Directory Facilitator to Schedule Tracker Agent.
3	FillRepository	Agent reference of the DER Gateway Agent controlling the device.	Updates the internally stored reference to the DER Gateway Agent controlling the device.	Schedule Tracker Agent to Schedule Tracker Agent.
4	Calculate Time	Power schedule corresponding to the device.	The agent calculates the time until next power set point, this is done every time a new set point is sent.	Schedule Tracker Agent to Schedule Tracker Agent.
5	SetData	Power set point.	The Schedule Tracker Agent sends the power sent point corresponding to the current time in the schedule.	Schedule Tracker Agent to DER Gateway Agent.
6	Inform	Confirmation that the set point has been accepted. A failure message is returned if something went wrong.	The DER Gateway agent informs the Schedule Tracker Agent about the reception and process of the given set point.	DER Gateway Agent to Schedule Tracker Agent.

10.6 Ontologies

For defining the messages to be exchanged between the different agents in the platform, JADE provides three types of JAVA classes:

- Concept: entities with complex structure defining concepts in the application domain. Concepts are defined in terms of slots, e.g.: (Person :name John :age 33)
- Predicate: expressions that say something about the status of the world and can be either true or false, e.g.: (Works-for (Person :name John) (Company :name Labein))
- AgentAction: indicate actions that can be performed by some agents, e.g.: (Sell (Book :title “The Lord of The Rings”) (Person :name John)). Actions when executed may produce an effect and generate a result.

Each agent providing a certain service has its own *agent actions* used by other agents to access those services, *predicates* that model the results of executing those *agent actions* and *Concepts* defining the elements for describing *agent actions* and *predicates*.

Device control ontology

The device control ontology provides the messages for getting data from devices and setting set points into devices being controlled. DER Gateway agents will offer DER control services and they will be able to receive, process and return messages according to this ontology.
Figure 8-18 shows an UML class diagram with the agent actions predicates and concepts defined as classes and their corresponding slots as attributes.

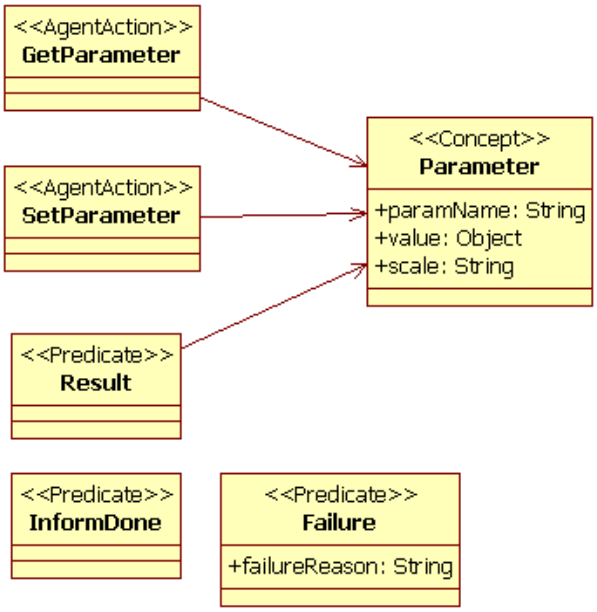


Figure 8-18: Device control ontology class diagram

The following table describes the elements in the ontology:
Table 8-IX: Device control ontology element table

Ontology Element	Description
Agent Actions	
GetParameter	This agent action is used to obtain the value of a certain parameter from the DER device.
SetParameter	Provides the capability to set a certain parameter to a given value. Used for setting set points into DER devices.
Predicates	
Result	The result corresponding to a certain GetParamater agent action, contains the value of the requested parameter.
InformDone	JADE built in predicate indicating that a given action has been successfully performed. Used as response to SetParameter agent action.
Failure	Failure is a JADE predicate indicating that a given action has not been successfully performed. Contains the failure reason. Used as response for GetParameter and SetParameter actions.
Concepts	
Parameter	Contains information about a given parameter: The name of the parameter identifies the parameter, the set of possible parameters are well defined and known to all agents in the system. The type of the value slot depends on the parameter and it can be Float or String. Each parameter value will be given according to a certain scale.

Schedule tracking ontology

The Schedule tracking ontology is used to access the service provided by the Schedule Tracker Agent. It contains an agent action for requesting the agent to track the given schedule. Next figure shows the UML class diagram with the elements in the ontology:

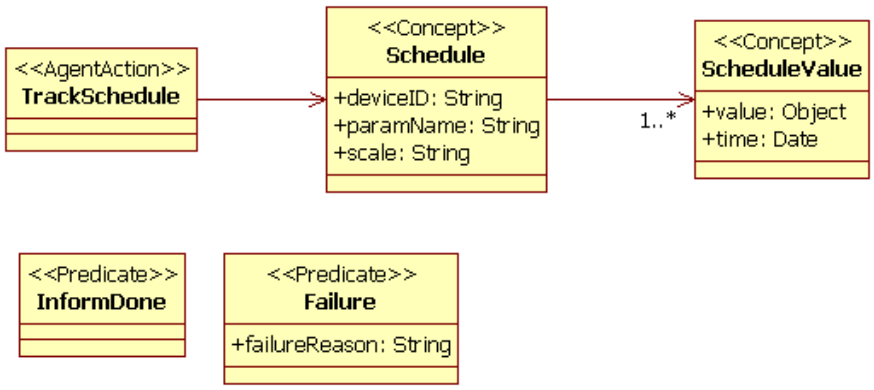


Figure 8-19: Schedule tracking ontology class diagram

Table 8-X defines the different elements in the ontology:
Table 8-X Schedule tracking ontology element table

Ontology Element	Description
Agent Actions	
TrackSchedule	This agent action is used to request the Schedule Tracker Ontology to start tracking the given schedule.
Predicates	
InformDone	JADE built in predicate indicating that a given action has been successfully performed. Used as response to TrackSchedule agent action.
Failure	Failure is a JADE predicate indicating that a given action has not been successfully performed. Contains the failure reason. Used as response for TrackSchedule action.
Concepts	
Schedule	The Schedule contains information about the device owning the schedule, the parameter for which the schedule is referred, the values that form part of the schedule and the scale for the scheduled parameter.
ScheduleValue	A schedule value contains information about a single value with is corresponding scheduled time.

Database access ontology

In order to give access to the information in the database, the Database Agent Gateway Agent provides the ontology elements shown in Figure 8-20. The messages exchanged with the Database Gateway Agent will be written according to this ontology.

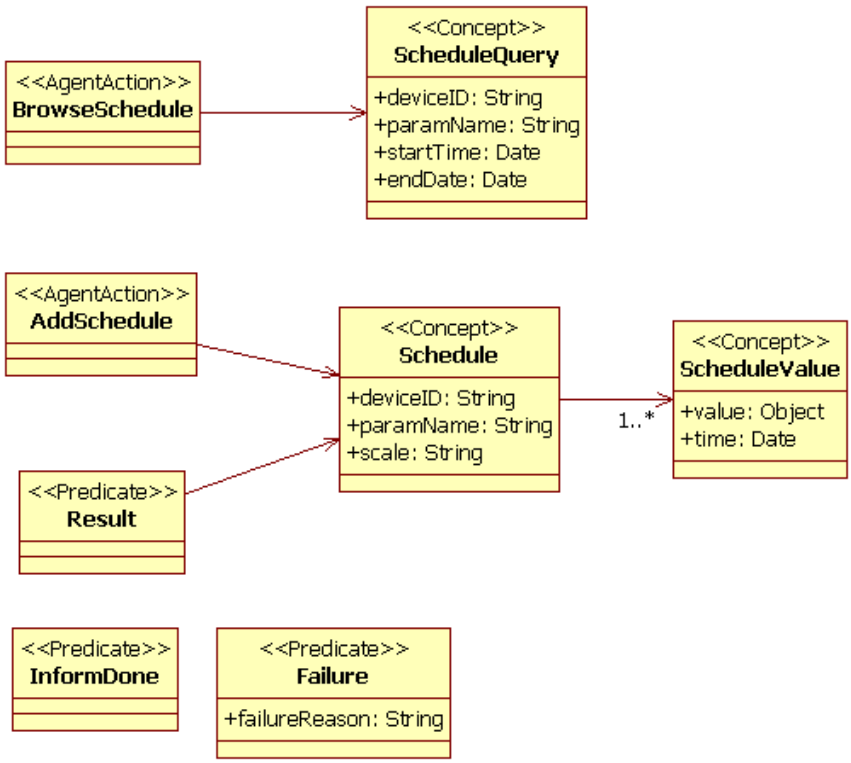


Figure 8-20: Database access ontology class diagram

Next table describes the different agent actions predicates and concepts used in the ontology:

Table 8-XI Database access ontology element table

Ontology Element	Description
Agent Actions	
BrowseSchedule	Agent action for requesting to browse a given schedule from the database.
AddSchedule	Agent action for requesting to add a given schedule into the database.
Predicates	
Result	As result of a BrowseSchedule agent action execution a Result message is returned pointing to the retrieved schedule.
InformDone	JADE built in predicate indicating that a given action has been successfully performed. Used as response to AddSchedule agent action.
Failure	Failure is a JADE predicate indicating that a given action has not been successfully performed. Contains the failure reason. Used as response for AddSchedule and BrowseSchedule actions.
Concepts	

Ontology Element	Description
ScheduleQuery	This concept holds information about the schedule to be browsed in a BrowseSchedule agent action. Contains the device ID for which the schedule is wanted, the parameter name corresponding the schedule and start and end times for the set of values in the schedule to be retrieved.
Schedule	The Schedule contains information about the device owning the schedule, the parameter for which the schedule is referred, the values that form part of the schedule and the scale for the scheduled parameter.
ScheduleValue	A schedule value contains information about a single value with is corresponding scheduled time.

Secondary regulation market ontology

The Regulation Matcher Agent will be listening for MakeBid messages containing secondary regulation bid information of the agents participating into the secondary regulation control system. Figure 8-21 shows the different elements in the secondary regulation market ontology:

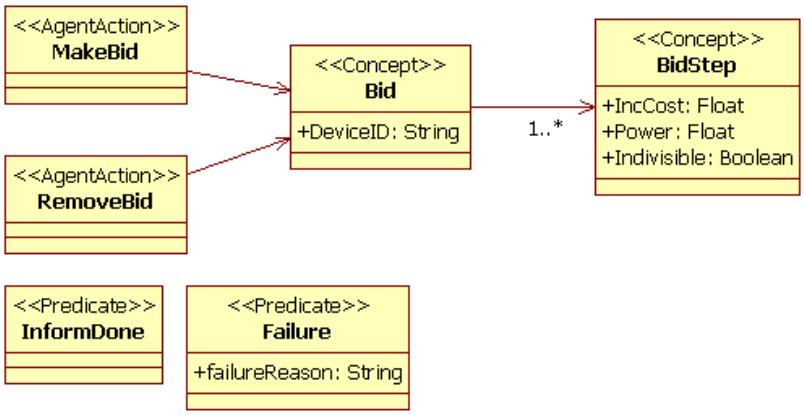


Figure 8-21: Secondary regulation market ontology class diagram

Next table describes the different agent actions predicates and concepts used in the ontology:

Table 8-XII: Secondary regulation market ontology element table

Ontology Element	Description
Agent Actions	
MakeBid	Used for making secondary regulation bids. Points to the Bid concept that holds information about the offered bid.
RemoveBid	Used to remove the bid associated to the specified device from the pool of bids held by the Regulation Matcher Agent.
Predicates	
InformDone	JADE built in predicate indicating that a given action has been successfully performed. Used as response to MakeBid and RemoveBid agent actions.
Failure	Failure is a JADE predicate indicating that a given action has not been successfully performed. Contains the failure reason. Used as response for MakeBid and RemoveBid actions.

Ontology Element	Description
Concepts	
Bid	Represents a bid for secondary regulation market system. Contains the identifier of the device providing the regulation capacity and a set of steps with power and price information.
BidStep	A bid step consists on the incremental cost corresponding to a given amount of power and the indivisibility indicator, if that indivisibility indicator is true, the step must be accepted totally, otherwise any power value can be accepted in the matching process.

10.7 Directory Facilitator Service Descriptions

This chapter defines the content of the agent services descriptions to be registered into JADE's DF. Each agent offering a certain service will register its self description in to the DF, this way, when any other agent wants to access a certain service, it can ask the DF to search fro the service descriptions that match the wanted service, the DF will return all found service descriptions, those service descriptions hold the reference to the agent providing the service. This way any agent can then send a message requesting the service execution to the agent providing it.
In JADE, service descriptions are given as a set of property=value pairs, some of these properties are defined by JADE while others are defined by the developer as needed by the application domain.

The following service description properties have been identified for implementing the Secondary Regulation Control system:

Table 8-XIII: Service description property table

Agent	Property	Description
All Agents (JADE built-in properties)	name	Name of the service. Identifies the service being described.
	type	The type of service.
	ontology	The JADE ontology used for constructing the messages that the agent offering the service needs in order to execute it.
	language	The content language of the ACL messages. FIPA-SL has been chosen as the content language for all the agent interactions.
	protocol	The protocol property specifies the interaction protocol used for accessing the agent service. Interaction protocols are standardised by FIPA and consists on a sequence of messages that form a conversation. More information about interaction protocols can be found at FIPA's specifications. For this implementation FIPA-REQUEST protocol will be used.
DER Gateway Agent	DeviceID	Unique identifier of the device being controlled by this agent.
	MeasParam	Comma separated values of the parameter identifiers corresponding to measurement and status values. (Active Power, Voltage, Started, …)
	ControlParam	Comma separated values of the parameter identifiers corresponding to controllable parameters. (Active Power Set Point, Stop, …)
	Characteristics	Comma separated value of device's characteristic such as maximum and minimum active power, etc. Each characteristic is given as *parameter_name=parameter_value:parameter_scale*
Schedule Tracker Agent	DeviceID	Unique identifier of the device being controlled by this agent.
	Schedule_Param	Comma separated values of the parameters that can be scheduled.
DER Bidder Agent	DeviceID	Unique identifier of the device being controlled by this agent.

Next figure shows an object diagram with examples about different service descriptions. Note that specific values in *DERGatewayAgent_SD* and *ScheduleTrackerAgent_SD* depend on the capabilities and characteristics of the device being controlled by the agent.

Figure 8-22: Service descriptions object diagram

10.8 Agent Implementation

The JADE definition for behaviour states that: "a behaviour represents a task that an agent can carry out, an agent can execute several behaviours concurrently and JADE provides the mechanism for automatically schedule the execution of those behaviours".

Agents will be developed based on JADE built-in behaviours that provide the way to easily create behaviours adapted to the different steps in agent conversations and algorithm execution. Those behaviours will implement messages reception and delivery together with message processing and task execution.

This chapter defines how to implement each agent in order to match the responsibilities defined in chapter 10.4 and agent interactions described in chapter 10.5. using messages in chapter 10.6 and agent descriptions defined in chapter 10.7.

The implementations are described by means of UML state transition diagrams showing the sequence of steps and behaviours that are used to implement agent tasks. Behaviours are managed by a behaviour queue that is in charge of scheduling them, executing all registered behaviours for a certain agent concurrently. This fact implies

that states corresponding to active behaviours will be executed concurrently. The arrows pointing to behaviours in the state diagrams represent those behaviours' instantiation, after instantiation the behaviours are automatically scheduled for execution by JADE's behaviour scheduling mechanism. As a general approach, when an agent is launched, first it performs an initialisation stage and then it starts listening for incoming messages, at the same time, the initialisation stage instantiates all required behaviours for performing agent specific tasks.
JADE provides some templates to develop behaviours for agent communications and task execution, the ones used for the implementation of the agent system are the following:

- Achieve Rational Effect Initiator Behaviour: This behaviour is used to start a FIPA-REQUEST interaction protocol; it sends a message to the specified receiver agent and contains methods for processing the response message received.
- Achieve Rational Effect Responder Behaviour: This behaviour is the complementary part of the Achieve Rational Effect Initiator Behaviour, it is instantiated when a message of the specified type is received by the agent. Holds the methods to process the request, execute requested task and send back a response.
- Finite State Machine Behaviour: Used to implement a composite behaviour that schedules its children behaviours according to a finite state machine whose states correspond to the FSM children.
- Ticker Behaviour: The Ticker Behaviour is scheduled when a specified time period is reached.
- One Shot Behaviour: This behaviour is executed just once when it is added to the agent.

Database Gateway Agent implementation

The following figure shows the behaviour diagram for implementing the Database Gateway Agent. This diagram shows the different states and behaviours in which the agent execution is divided.

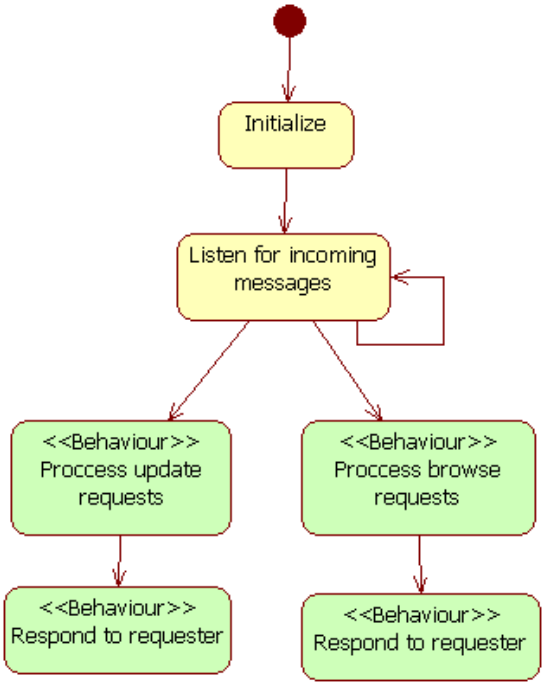


Figure 8-23: Database Gateway Agent behaviour diagram

Next table describes each behaviour and step for agent task execution:
Table 8-XIV: Database Gateway Agent behaviour table

State	Description
Initialize	The initialization sets the agent ready to perform its tasks. It consists on: • Configure agent and database specific properties. • Register the ontology and language to be used for communications (Database access ontology and FIPA-SL). • Register its service description into the DF. • Setup database engine. • Register behaviour in charge of processing update request. • Register behaviour in charge of processing browse request.
Listen for incoming messages	JADE's message queue management mechanism is in charge of listening for incoming requests and instantiating the proper behaviour depending on the arrived message type (BrowseSchedule or AddSchedule messages).
Process update requests	This behaviour extends JADE's Achieve Rational Effect Responder Behaviour. When a Browse Schedule message arrives, this behaviour is instantiated. It uses the database engine in order to perform the query that updates the data given in the request message into the database. When the database operation concludes it responds to the requester with an Inform message.
Process browse requests	This behaviour extends JADE's Achieve Rational Effect Responder Behaviour. When an Update Schedule message arrives, this behaviour is instantiated. It uses the database engine in order to perform the query that browses the data according to the query in the request message. When the data is obtained from the database, it creates and inform message with the result data and sends it to the requester agent.

Regulation Matcher Agent implementation

Figure 8-24 describes the different stages to be implemented for the Regulation Matcher Agent implementation:

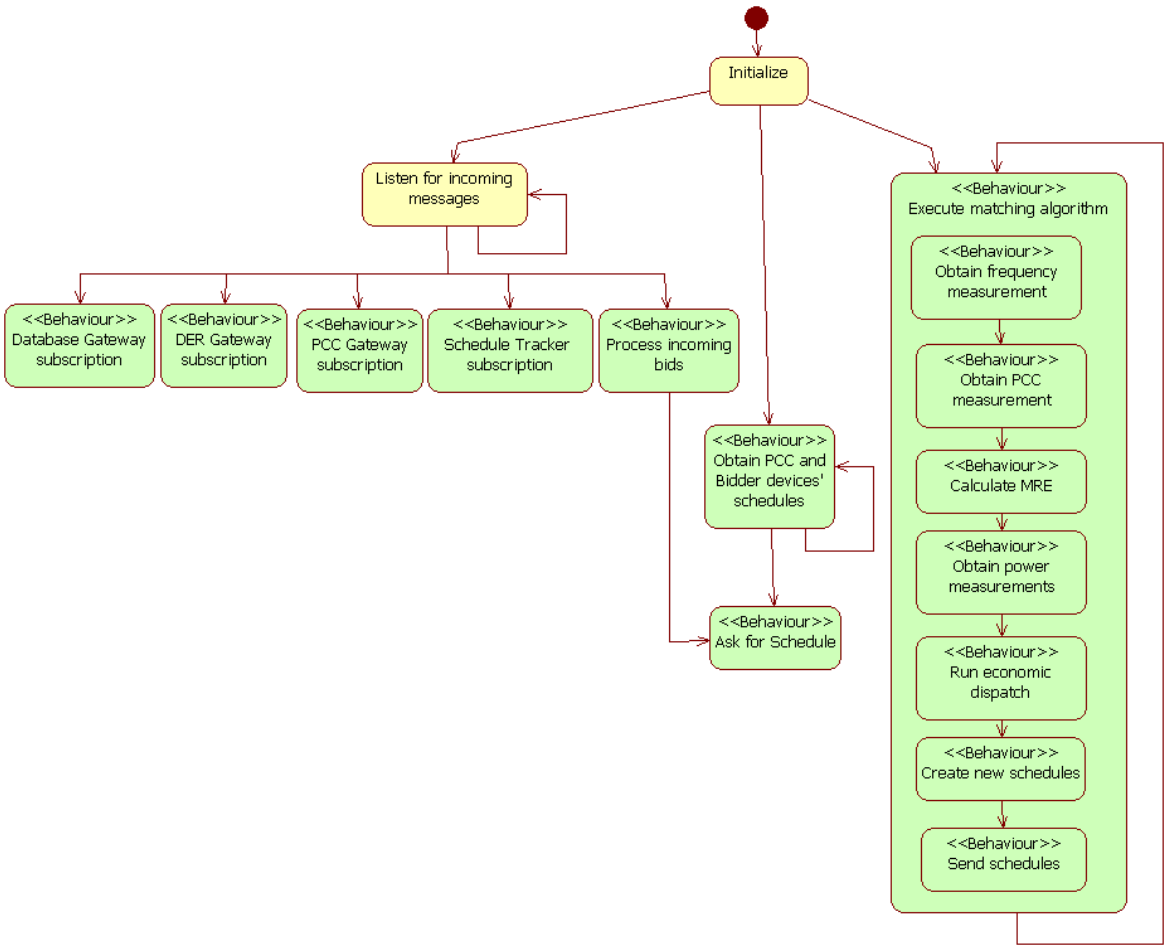


Figure 8-24: Regulation Matcher Agent behaviour diagram

The following table describes each state as shown in the above figure:
Table 8-XV: Regulation Matcher Agent behaviour table

State	Description
Initialize	The initialization sets the agent ready to perform its tasks. It consists on: • Configure agent's specific properties. • Register the ontology and language to be used for communications (Secondary regulation matcher ontology, device control ontology, schedule tracking ontology, database access ontology and FIPA-SL). • Register its service description into the DF. • Register following behaviours: o Database gateway service subscription behaviour. o DER gateway service subscription behaviour. o PCC gateway service subscription behaviour. o The behaviour processing incoming bids. o The behaviour used of asking the Database Gateway Agent for PCC and devices' schedules. o The behaviour in charge of executing the secondary regulation matching algorithm.
Listen for requests	JADE's message queue management mechanism is in charge of listening for incoming requests and instantiating the proper behaviour depending on the arrived message type (Subscription notifications, MakeBid or RemoveBid messages).
Database Gateway subscription	Implements JADE's Subscription Initiator behaviour. Every time an Agent offering Database Gateway service registers into the DF or deregisters from the DF a notification message is received and this behaviour is instantiated. The behaviour stores the reference to the agent providing database gateway service so it can be used by the agent to access the database.
DER Gateway subscription	Implements JADE's Subscription Initiator behaviour. Every time an Agent offering DER Gateway service registers into the DF or deregisters from the DF a notification message is received and this behaviour is instantiated. The behaviour stores the list of references to the agents providing DER gateway services so it can be used by the agent to access DER devices' parameters.
Schedule tracking subscription	Implements JADE's Subscription Initiator behaviour. Every time an Agent offering Schedule tracking service registers into the DF or deregisters from the DF a notification message is received and this behaviour is instantiated. The behaviour stores the list of references to the agents providing schedule tracking services so it can be used by the agent when sending new schedules after bid matching process execution.
PCC Gateway subscription	Implements JADE's Subscription Initiator behaviour. Every time an Agent offering PCC Gateway service registers into the DF or deregisters from the DF a notification message is received and this behaviour is instantiated. The behaviour stores the reference to the agent providing PCC gateway service so it can be used by the agent get PCC information.
Process incoming bids	This behaviour extends JADE's Achieve Rational Effect Responder Behaviour. When a Make Bid or Remove Bid message arrives, this behaviour is instantiated. It adds the incoming secondary regulation bid to the list of bids so it can be used by the agent when executing the matching algorithm. It also associates a Schedule Tracker Agent and DER Gateway Agent to the bid in order to be able to obtain measurements and send new schedules. Instantiates "Ask for schedule" behaviour, obtaining this way schedule in the database corresponding to the device's bid. When receiving an Remove Bid request removes the bid from the bid pool.
Obtain PCC and Bidder devices' schedules	Implements JADE's Ticker Behaviour that periodically executes "Ask for Schedule" behaviour in order to obtain PCC and Bidder devices' schedules.
Ask for schedule	Achieve Rational Effect Initiator Behaviour in charge of asking the Database Gateway Agent for the latest schedules corresponding to the power exchange at the PCC and the devices participating in secondary control bid.

State	Description
Execute matching algorithm	Implements JADE's Ticker Behaviour that periodically executes a Finite State Machine Behaviour in charge of performing the matching algorithm. The states that form part of the Finite State Machine Behaviour are: • Ask for frequency measurement. • Ask for power measurement at the PCC. • Calculate MRE. • Ask for power measurements to devices participating in the regulation control. • Run economic dispatch. • Create new schedules. • Send schedules.
Ask for frequency measurement	Implements an Achieve Rational Effect Initiator Behaviour in charge of asking the DER Gateway Agent selected for providing system frequency measurement.
Ask for power measurement at PCC	Implements an Achieve Rational Effect Initiator Behaviour in charge of asking the PCC Gateway Agent for the power measurement at the PCC.
Calculate MRE	JADE's One Shot Behaviour implementation that calculates the MRE.
Ask for power measurements	Implements an Achieve Rational Effect Initiator Behaviour in charge of asking for all the power measurements corresponding to those agents offering secondary regulation bids. The receivers' references are obtained from the data associated to secondary regulation bids.
Run economic dispatch	One Shot Behaviour implementation that runs the economic dispatch algorithm.
Create new schedules	One Shot Behaviour implementation in charge of creating new schedules for dispatched devices.
Send schedules	Achieve Rational Effect Initiator Behaviour in charge of asking the Schedule Tracking Agents to track the created new power schedules.

DER Gateway Agent implementation

The DER Gateway Agent comprises four types of agents: Load, Generator, Storage and PCC Gateway agents. All these agents are local agents, each one controlling one DER device. Even if they have specific functions and protocols, since they must deal with the specific features of the DER device attached to them, they can be described in a generic way.
When developing a DER Gateway Agent for a particular device, there are two ways to face the specific functions and communications used by the device:

- Integrate DER device's specific functions and communication protocols into the particular agent.
- Develop a separate software acting as protocol translator and offering a common set of services for the agent system.

The latter approach allows integrating the particular issues of the DER device outside the agent software, thus, the agents implementation will be the same independently of device's technology, capabilities and communication protocol. Anyway the implementation described here does not define in detail device specific issues, so it is valid for both implementation approaches.

The following figure shows the behaviour diagram for a generic DER Gateway Agent:

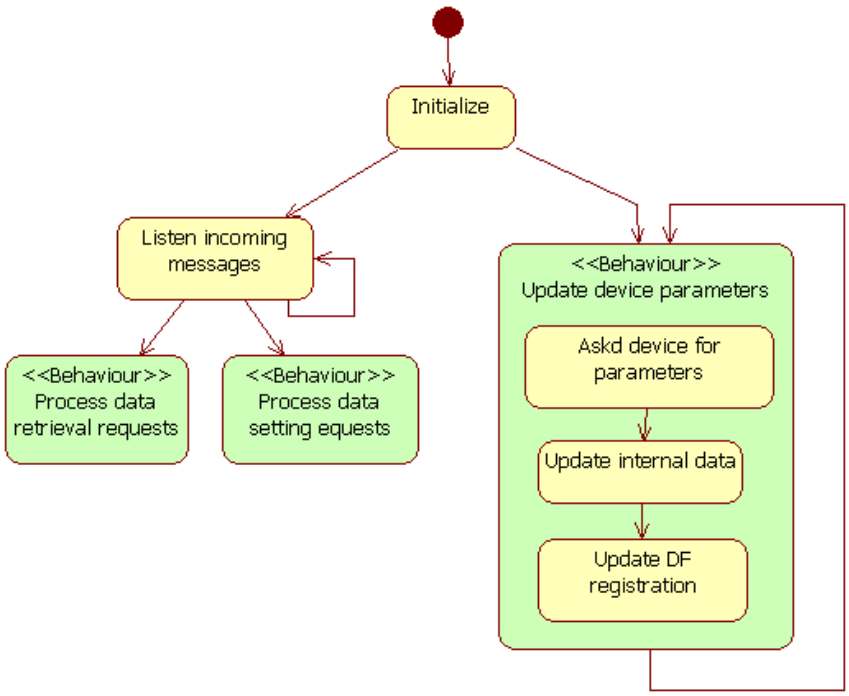


Figure 8-25: DER Gateway Agent behaviour diagram

The states shown in above figure are described in more detail in Table 8-XVI:

Table 8-XVI: DER Gateway Agent behaviour table

State	Description

Initialize	The initialization sets the agent ready to perform its tasks. It consists on:
	• Configure agent and device specific properties.
	• Register the ontology and language to be used for communications (Device control ontology and FIPA-SL).
	• Register its service description into the DF.
	• Setup communications with the device.
	• Register behaviours:
	  ○ Behaviour listening for GetData incoming messages.
	  ○ Behaviour listening for SetData incoming messages.
	  ○ Behaviour in charge of updating device parameter data.

Listen for requests	JADE's message queue management mechanism is in charge of listening for incoming requests and instantiating the proper behaviour depending on the arrived message type GetData or SetData messages).
Process data retrieval requests	This behaviour extends JADE's Achieve Rational Effect Responder Behaviour. When a GetData message arrives, this behaviour is instantiated. It returns the requested data value getting it from the agent's internal data repository.
Process data setting requests	This behaviour extends JADE's Achieve Rational Effect Responder Behaviour. When a SetData message arrives, this behaviour is instantiated. Communicates with the device and sends the corresponding command to it. Returns an inform message if everything went well, otherwise returns a failure message.
Update device parameters	Implements a JADE Ticker behaviour that periodically asks the device for the available data and stores it in an internal in-memory repository.

State	Description
Ask device for parameters	Communicates with the device in order to obtain last available data.
Update internal data	Updates the data held by the agent.
Update DF registration	Changes the agent's registration in the DF if needed. For example if the any of the device's characteristic changes.

Schedule Tracker Agent implementation

The schedule Tracker agent will receive power schedules and will send the corresponding SeData messages to the DER Gateway Agent controlling the device according to the received schedule.
The following figure shows the behaviour diagram for the Schedule Tracker Agent:

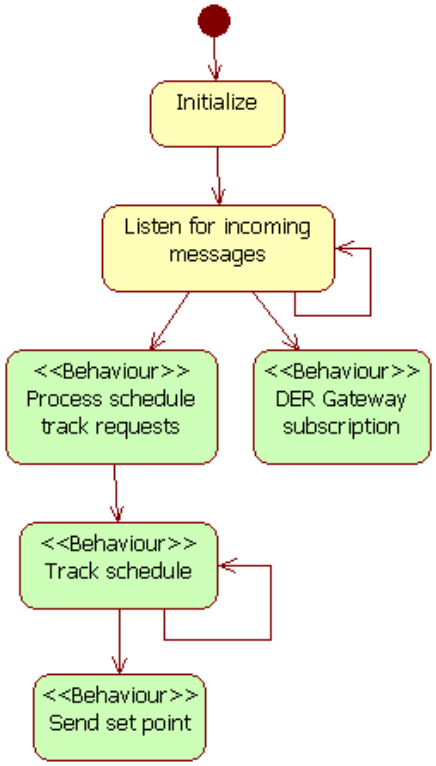


Figure 8-26: Schedule Tracker Agent behaviour diagram

Following table describes the behaviours corresponding to the Schedule Tracker Agent:
Table 8-XVII: Schedule Tracker Agent behaviour table

State	Description

Initialize	The initialization sets the agent ready to perform its tasks. It consists on:
	• Configure agent specific properties.
	• Register the ontology and language to be used for communications (Device control ontology, Schedule tracking ontology and FIPA-SL).
	• Register its service description into the DF.
	• Register behaviours:
	  ○ Behaviour listening for TrackSchedule incoming messages.
	  ○ Behaviour subscribing to agents offering DER Gateway service for the controlled device.

State	Description
Listen for incoming messages	JADE's message queue management mechanism is in charge of listening for incoming requests and instantiating the proper behaviour depending on the arrived message type (TrackSchedule or Notify messages)
DER Gateway subscription	Implements JADE's Subscription Initiator behaviour. When the Agent offering DER Gateway service for the configured device registers into the DF or deregisters from the DF a notification message is received and this behaviour is instantiated. This behaviour stores the reference to the agent providing DER Gateway service so it can be used by the agent to send set point commands to the device.
Process schedule tracker	This behaviour extends JADE's Achieve Rational Effect Responder Behaviour. When a TrackSchedule message arrives, this behaviour is instantiated. It adds a new behaviour in charge of sending the power set points to the device being controlled.
Track schedule	This behaviour extends JADE's Ticker Behaviour. When it wakes after the specified time period, adds a new behaviour in charge of sending the set point corresponding to that moment in the schedule to the Device Gateway Agent. Finally, adds itself to the behaviour queue to be waked the next time a new set point must be sent according to the schedule
Send set point	Achieve Rational Effect Initiator Behaviour in charge of sending a SetData message to the DER Gateway Agent controlling the device.

DER Regulation Bidder Agent implementation

The DER Regulation Bidder Agent is a generic agent in charge of creating bids for secondary regulation control. When implementing local controllers for the agent system, this agent will be specialized according to the needs and particularities of the controlled device.

The implementation description defined here comprises Generator Regulation Bidder Agent, Load Regulation Bidder Agent, Storage Regulation Bidder Agent and PCC Regulation Bidder Agent (see chapter 10.4).

Figure 8-27 shows the behaviour diagram for a generic DER Regulation Bidder Agent:

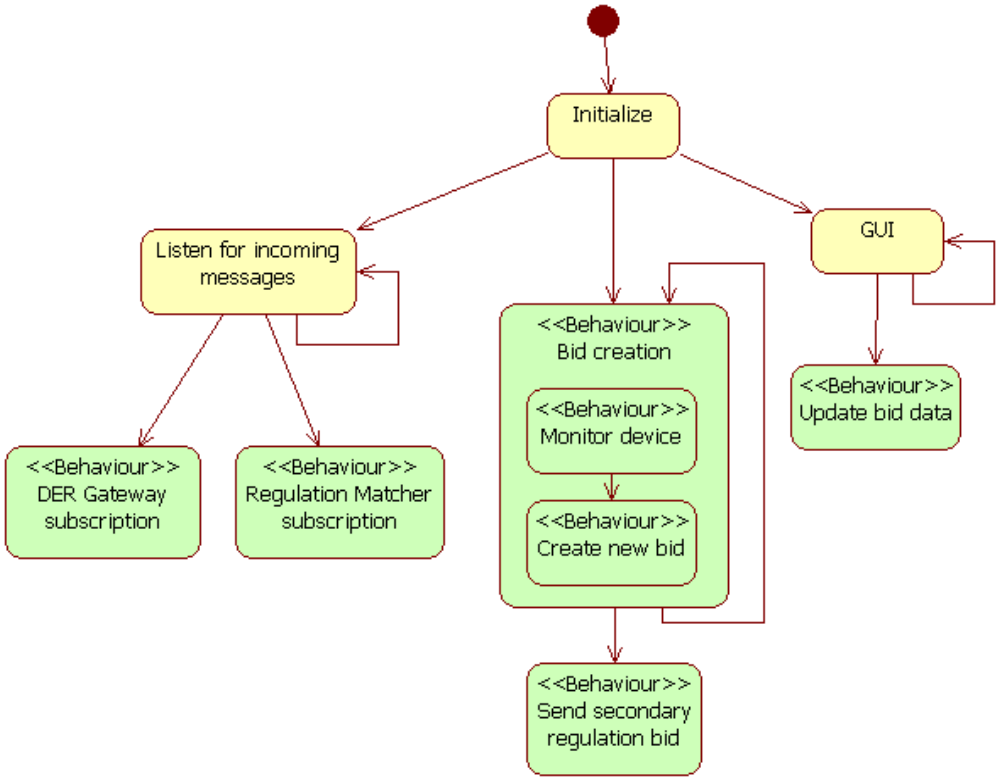


Figure 8-27: DER Regulation Bidder Agent behaviour diagram

The different steps and behaviours are explained in the following table:

Table 8-XVIII: DER Regulation Bidder Agent behaviour table

State	Description

Initialize	The initialization sets the agent ready to perform its tasks. It consists on:
	• Configure agent specific properties.
	• Register the ontology and language to be used for communications (Device control ontology, Secondary Regulation Market ontology and FIPA-SL).
	• Register its service description into the DF.
	• Register behaviours:
	  ○ Behaviour subscribing to the agent offering DER Gateway service for the controlled device.
	  ○ Behaviour subscribing to the agent offering Regulation Matching service.
	  ○ Behaviour for bid creation.
	  ○ Graphical user interface for interacting with device's owner.

Listen for incoming messages	JADE's message queue management mechanism is in charge of listening for incoming requests and instantiating the proper behaviour depending on the arrived message type.
DER Gateway subscription	Implements JADE's Subscription Initiator behaviour. When the Agent offering DER Gateway service for the configured device registers into the DF or deregisters from the DF a notification message is received and this behaviour is instantiated. It updates the agent's reference to the agent controlling the DER device, so the agent can communicate with it at any time.

State	Description
Regulation Matcher subscription	Implements JADE's Subscription Initiator behaviour. When the Agent offering Regulation Matcher registers into the DF or deregisters from the DF a notification message is received and this behaviour is instantiated. It updates the agent's reference to the agent performing secondary regulation matching functions, so the agent can communicate with it at any time.
Bid creation	Implements JADE Ticker Behaviour that periodically executes a Finite State Machine Behaviour with "Monitor device and user needs" and "Create new bid" sub behaviours. Depending on the approach taken for creating the bids the implementation of this step is different, some policies for creating bids are discussed in chapter 9.7. Anyway, as a general rule, the agent will monitor device's status and user inputs and create accordingly secondary regulation bids. Following are some examples of possible implementations for this step: • In case a generator is being controlled, the agent could provide the user with a GUI where he could enter the bid manually according to fuel prices and generator's cost curves. • For a battery system, the agent would be monitoring the state of charge of the battery modifying the offered bids according to it. • In case of a load is being controlled the approach could be also to have a GUI where the user of the load could enter its preferences in order to be curtailed or not. • When the PCC participates in secondary regulation control, the agent could monitor the prices for electricity and decide change its bids according to them
Monitor device and user needs	Implements an Achieve Rational Effect Initiator Behaviour in charge of asking the DER Gateway Agent for monitored parameter.
Create new bid	One Shot Behaviour in charge of creating new bids according to monitored parameters and user inputs.
Send secondary regulation bid	This behaviour implements a Achieve Rational Effect Initiator behaviour in charge of sending the newly created bids to the Secondary Regulation Matcher Agent.

11. Conclusions

This document presents a Microgrid management application for executing secondary regulation control functions. Secondary regulation control allows maintaining the contracted power exchange with the main grid together with the frequency close to the reference in a real time basis. The presented control algorithm has the following characteristics:

- The control algorithm is valid both for Microgrid islanded and grid connected operating modes.
- It performs active power correction by sharing the excess or defect of power economically among the different DER devices (generators, loads and storage systems).
- The control algorithm is based on a market approach allowing decentralized local decisions where each DER device decides autonomously how to participate in secondary regulation.

Furthermore, in addition to the algorithm definition, a detailed description on how to implement the control system using intelligent agent technologies has been presented. The agent implementation description follows a step by step approach covering from the analysis to the design and implementation of the software agent platform. This

agent implementation takes special care about the system plug and play capabilities, extensibility and easy deployment while accurately implementing the secondary regulation algorithm described in the first part of the document.
The presented secondary regulation control system will be tested in a laboratory Microgrid within Workpackage WPF, those tests will be used to analyze the control system's validity and performance, identify system's weakness and provide ideas for improving the algorithm and its implementation.

12. References

[1] Allen J. Wood and Bruce F. Wollemberg, "Power generation operation and control", Book, John Wiley & Sons, INC. 1996, USA, ISBN 0-471-58699-4.

[2] John J. Grainger, William D. Stevenson, "Power System Analysis", Book, McGraw-Hill, Inc. 1994, USA. ISBN 0-07-113338-0.

[3] R. Lasseter, A. Akhil, C. Marnay, *et al*. "White Paper on Integration of Distributed Energy Resources – The CERTS Microgrid Concept" California Energy Commission, PIER Energy-Related Research. 2003.

[4] Gibson, Gerald L., "Intelligent Software Agents – effective integration of Distributed Energy Resources into the California energy Marketplace", California Energy Commission, PIER Energy-Related Research. CEC-TBD. 2006

[5] K. De Brabandere, K. Vanthournout, J. Driesen, G. Deconinck, R. Belmans, "Control of Microgrids", Power Engineering Society General Meeting, IEEE, 2007.

[6] F. Bellifemine, G. Caire, D. Greenwood, "Developing Multi-Agent Systems with JADE", John Wiley and Sons Ltd, 2007

[7] M. Nikraz, G. Caire, P. A. Bahri, "A methodology for the Analysis and Design of Multi-Agent Systems using JADE", Murdoch University, Telecom Italia Lab.

[8] JADE web site, www.jade.tilab.com.

[9] FIPA web site, www.fipa.org.

[10] UML, Object management Group web site, http://www.uml.org/.