Managing Unavailabilities in a Dynamic Scenario Following an Agent-Based Approach

Angela Locoro and Viviana Mascardi
Computer Science Dept. (DISI)
University of Genova, Italy
{Angela.Locoro,Viviana.Mascardi}@unige.it

Franco Mortara and Renato Sanna
SELEX Elsag,
Genova, Italy
{Franco.Mortara,Renato.Sanna}@elsagdatamat.com

***Abstract*—The correct management of resources that can become unavailable over time and the efficient (in terms of both cost and time) re-allocation of the services they provided before becoming unavailable, is an open problem that arises in a wide range of application domains.**

Despite to some differences, many scenarios spanning very different domains, from logistics to industrial automation, from telecommunication to water systems, may be considered as instances of a more general situation, and the actions to take in order to solve the unavailability problem follow similar patterns.

In this paper we discuss the implementation of a multiagent system for solving unavailability problems in an artificial - although realistic - scenario which takes inspiration from to the electricity power network domain.

I. Introduction

Many real scenarios where assets are produced in some physical locations and must be delivered to other physical locations where they are consumed, can be described in a very abstract way as graphs of nodes of different types, connected by edges characterized by different transportation capacities. Assets are created in a set of "producer nodes", must traverse the graph towards "consumer nodes", and need to use edges and intermediate nodes during the graph traversal. According to the scenario, it may be useful to model entities specifically devoted to the physical transportation from one node to another one as well (think of a logistic network where freight trucks and other vehicles are in charge of moving assets in the network).

Producers and consumers may be further organized in hierarchies where each producer controls a set of "sub-producers" and each consumer dispatches the assets it receives to a set of "sub-consumers", connected by edges and intermediate nodes according to different topologies. Whatever the level in the hierarchy, nodes and edges may show some degree of autonomy and may be able to manage problems arising during their operation on their own – at least up to some extent, before involving entities higher in the hierarchy. The problems we are mainly interested in, are those due to resources (nodes, edges, other entities relevant for the scenario) that become partially or completely unavailable and whose functionalities must be redistributed among other resources in the network.

Such scenarios are very suitable to be analyzed following an agent-based metaphor [8]. In fact, the entities belonging to the scenarios above are

- situated, since they must be aware of the portion of the environment where they are located, in order to perceive changes in the neighboring entities and in their own production, consumption, and/or transportation capacity;
- autonomous, since they must be able to solve at least "small" problems locally, before spreading to all their neighbors or throughout the entire network the information that a problem occurred;
- flexible, namely
  - reactive, since they must realize that a problem occurred in a timely fashion;
  - proactive, since each of them has its long-term goal (producing, consuming, moving a certain amount of assets);
  - social, since real scenarios usually involve humans or companies which, of course, communicate using an high level communication language, and following complex interaction protocols.

Figure 1 shows a generic scenario for unavailability management: a multiagent system that implements a scenario of this kind, integrates the specific features of the application domain under consideration, and proposes a reconfiguration of the asset production, consumption and transportation load in order to face unavailabilities that dynamically show up, could be profitably used ad a system supporting decisions of a human operator.

Many real scenarios may be seen as instances of the general one, and the actions that a decision support system should take in order to help solving the unavailability problem follow similar patterns.

For example, in a logistic network the resource that becomes unavailable might be a stretch of road and the problem to be faced concerns to which other roads traffic should be deviated in order to ensure the quality of service required by the client; in a ship air conditioning plant, the resource that becomes unavailable might be an air channel and the problem to solve is how to re-configure junctions among available channels so to guarantee air conditioning to at least those rooms that must necessarily keep a pre-defined temperature; in an electric power network, the resource that becomes unavailable might be a power station or a cable transporting electricity and the problem to solve is which power stations should be required

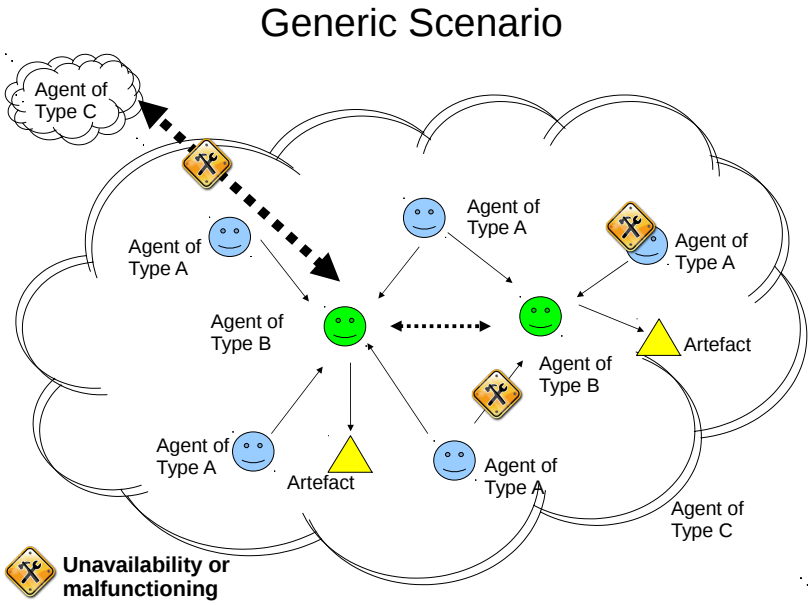

Fig. 1. The generic scenario for unavailability management

to start producing electricity, to provide for the reduction in power supply. Despite to some differences, all these scenarios may be considered as instances of a the more general situation that we modeled as nodes, edges, and, if it is the case, additional entities that traverse the graph.

In this paper we describe a feasibility study that the scientists of the CS Department of Genova University (DISI) carried out in collaboration with engineers from Elsag Datamat[1], the Finmeccanica center of excellence for the design and production of systems, services and solutions in automation, security, transport and information technology.

The purpose of this study was to assess the suitability of an agent-oriented approach to the management of unavailabilities in complex, distributed scenarios. In order to give an answer to that question coming from Elsag Datamat we made an "academic exercise" that takes inspiration from to the electricity power network domain: although artificial, it is almost realistic and grounds upon information publicly available that we obtained from the web sites of "Gestore dei Mercati Energetici S.p.A.", a public company whose mission is that of organizing and economically managing the Electricity Market, and from other public information sources.

The paper is organized in the following way: Section II introduces the model we designed after an accurate analysis of existing and publicly available information, Section III discusses the architecture of the UMMAS system we designed and implemented and shows some details about its execution, Section IV provides an overview of the related work and outlines some lines for our future research.

[1]From the 1st of June 2011, Elsag Datamat combined with SELEX Communications into SELEX Elsag. The work described in this paper was carried out with Elsag Datamat before that date.

II. Domain Analysis and Model

Starting from an accurate look into the Italian electricity network and market, an analysis of the field has been carried out, as the result of the following questions:

1) What are the main entities acting in the scenario at hand?
2) What are the relevant coordinative actions that such actors may play according to their roles and to the physical, economical and topological constraints of the real domain?
3) What kind of events may compromise the system consistency or, otherwise said, what are the main events that need to be simulated and managed in order to find a solution, if one of them occurs in the system?

An overview of the domain entities and the high level topological constraints that our study of the domain let emerge is depicted in Figure 2, where a Zone level topological perspective of the Italian electricity network is sketched. As the outmost subdivision of the Italian territory in the domain of electricity supply, a Zone contains the following elements:

- a set of Power Stations, each of which has one or more output links toward one or more Power Exchange Points[2]
- a Power Exchange Points set, each of which has input links from a Power Station subset and output links to

[2]We refer here to the entity with the following definition "a set of points in the electricity network such that, for dispatching purposes, it does not matter in which point electrical power is produced and in which ones it is consumed. Hence, one and only one Power Exchange Point is associated with each production/input (resp. consumption/output) point, whereas more input (resp. output) points are in general subsumed by the same Power Exchange Point.", translated into English by the authors from the Italian definition, available at http://www.mercatoelettrico.org/ in the documentation that the website returns as the first result of searching the key phrase: "punto di scambio rilevante" (Power Exchange Point in English).

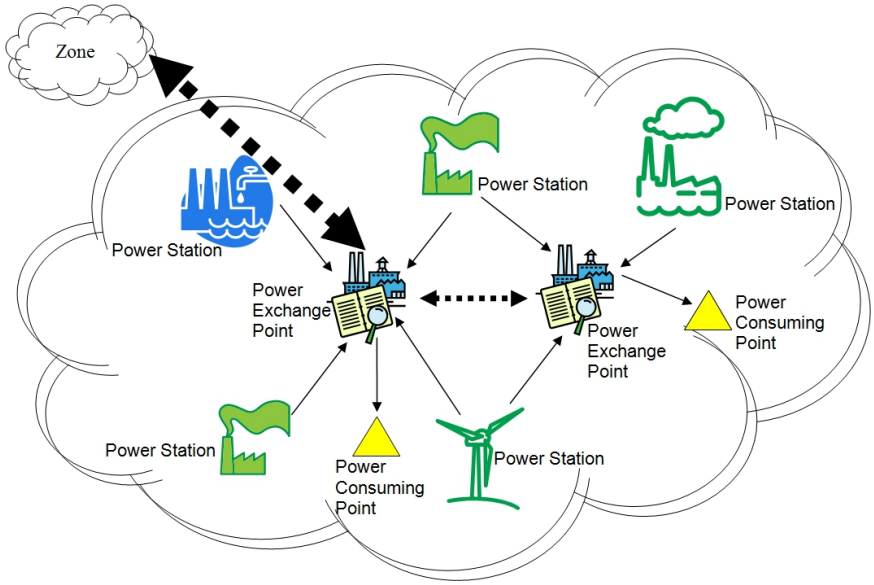

Fig. 2. The Italian electric power network as a specialization of the general scenario

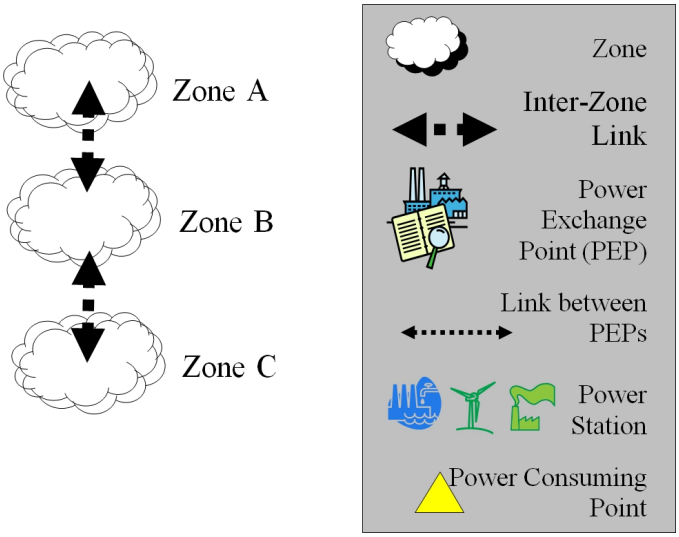

Fig. 3. Electric power network topology: the inter-zone domain and symbols used

Consume Points. Some of them may also have links with other Power Exchange Points of the same Zone, and one or more inter-Zone links, that is a physical link between Power Exchange Points of neighbor Zones.

An inter-Zone level overview of the Italian electricity network, together with an explanation of symbols for our domain, as described until now, is reported in Figure 3.

In the next paragraphs we are going to describe in details the main actors and the architectural choices that we adopted for our system as being inspired by the topological schema shown in the above figures, by the competency questions reported above, and by some simplifying assumptions made during our domain modeling analysis, which we will explain along with the system model itself.

A. Actors and their Properties

The main actors defined for modeling the Italian electricity network scenario are:

The Power Station (PS from now on), modeled according to the following properties:

- physical features:
  - type: thermal PS, hydroelectric PS or Renewable energy PS;
  - Minimum Electrical Power (minPow) and the Maximum Electrical Power (maxPow), that represent the PS energy supply boundaries, expressed in Megawatt per hour (MWh);
  - quantity of supplied energy at the PS full capacity (Baseload) that is computed as a function per hour based on the previous day MWh energy delivered by the PS itself, and that lies between its minPow and maxPow rates. We define it as $f_{bl} : h \rightarrow MW$;
  - power-on ramp and power-down ramp, determined by the MW and time necessary to the PS for reaching its Baseload from its start-up (resp. shutting down completely after being turned-off);
  - Secondary Reserve (SR), that is the quantity of ready-to-use electrical power always available for each PS to be put in the system after its power-on ramp time (resp. to remove from the system at its power-down ramp time);
  - Other Services (OS), that is the result computed as a function per hour of the following difference: maxPow - Baseload - SR.
- topological features: outward links to one or more Power Exchange Points.
- economical features:
  - Baseload price, calculated over the previous day energy supply and based on the price established by

the energy exchange market;
  - SR price, calculated as the above Baseload, and defined as the function $f_{sr} : h \rightarrow Euro/MW$;
  - OS price, calculated as the above Baseload, and defined as the function $f_{os} : h \rightarrow Euro/MW$.

The Power Exchange Point (PEP from now on), modeled according to the following properties:

- physical features: Demanding rate, calculated as a function per hour of the MW consumed. We define it as $f_{con} : h \rightarrow MW$
- topological features:
  - inward links from connected PS;
  - intra-Zone links that provide its connections with other PEPs, and are characterized by a function that states their availability per hour. For each link of this kind we define the function $f_{link_i} : h \rightarrow \{on, off\}$;
  - inter-Zone links that connect PEPs of different neighbor Zones. These links are further characterized by the so called transit limits, expressed by the function $f_{tl} : h \rightarrow MW$, representing the quantity of energy that can be carried at each hour in and out from two different Zones;

The Zone, as a geographic area fully characterized by the PSs, PEPs, intra- and inter-Zone links belonging to it.

B. Main Simulation Scenarios

An overview of the unavailabilities or malfunctioning analysed as part of the unexpected events that may occur into our system is reported in Figure 4. An exhaustive gloss for each of them is available in the next paragraphs.

The More Energy Demand event is an event that may occur in the system when a PEP has a higher energy consumption than expected. The situation requires that one or more PSs augment the quantity of energy delivery in order to fulfill the increasing request of electricity. The inquired PSs may be directly or indirectly connected to the PEP and may be located in another Zone with respect to those where the request comes from. In this last case the quantity of energy that out-of-zone PSs may supply should be able to respect the transit limit constraint.

The criterion to choose which PS needs to be activated in order to supply more energy is a search over all the PSs in the same Zone or in other Zones that can deliver their SR and, among them, all those PSs that offer it at the best available price. If the amount of SR available does not cover the entire energy request, than a search for AS energy is activated under the same best available price condition.

The intra-Zone broken link event is an event that may occur in the system when a damage causes the unavailability of a link between a PS and a PEP. This case has been reduced in our model to the "PS unavailability or malfunctioning" one that is depicted in the last paragraph of this Section. A sub-case that has been analysed even if not modeled into our system is the one in which the PS is linked to another PEP that may be used as a bridge in order to supply the energy to the original PEP, if a proper connection exists between the two PEPs.

The inter-Zone broken link event is an event that may occur in the system when a damage causes the total or partial unavailability of a link between two Zones. The two Zones involved should reconfigure their power production in order to avoid that more energy than those physically chargeable within the new transit limit is not supplied anymore.

The PS unavailability or malfunctioning event is an event that may occur in the system when a damage causes the total or partial unavailability of a PS that stops working or supply less energy than expected or needed. It is then necessary to know which portion of power was delivered to the PEPs involved and connected to the PS. They should retrieve the missing energy from other PSs, under the best available rate criterion. This event can be reduced to the first one (the More Energy Demand event) and be modeled and managed in the same way.

III. UMMAS: the Unavailability Management Multi-Agent System

An overview of the main features of UMMAS is presented in this Section.

A. Purpose

The UMMAS (Unavailability Management Multi-Agent System) purpose is to rapidly find a sub-optimal scenario (that is the cheapest solution in terms of best available price for electricity) able to maintain the whole electricity network configuration in a consistent state with respect to an event such as a maintenance operation, a malfunctioning or an unexpected change in energy consumption that may occur in the system itself.

UMMAS has been conceived as a decision support tool for planning maintenance operations. The requirement to come to a fast system reconfiguration has forced us to apply heuristics that do not guarantee to reach an optimal solution. Moreover, a MAS oriented approach is not by its own nature the best one to achieve a "global optimization" solution whereas its effectiveness and flexibility represent the best choice for dealing with local optimization problems.

B. Input Topology and Data

UMMAS works with an oversimplified model of the electric power topology that is embedded into the system. A future extension is foreseen in order to provide more complex topologies as input data or configuration files. All the data available by the system are:

- for each PS: all the fixed parameter features (that is the type, the minPow and maxPow, the power-on ramp and power-down ramp) and the functions per hour, that is its f_{bl}, f_{sr}, f_{os};
- for each PEP: its f_{con};
- for each intra-Zone link, its availability function f_{link_i};
- for each inter-Zone link, its f_{tl};

An input parameter is also given for each kind of unexpected event (we consider that a PEP consume rate, a PS baseload, an intra-Zone link availability, and an inter-Zone transit limit may all vary in the system) that may occur at each time slot (we consider a time unit as one hour).

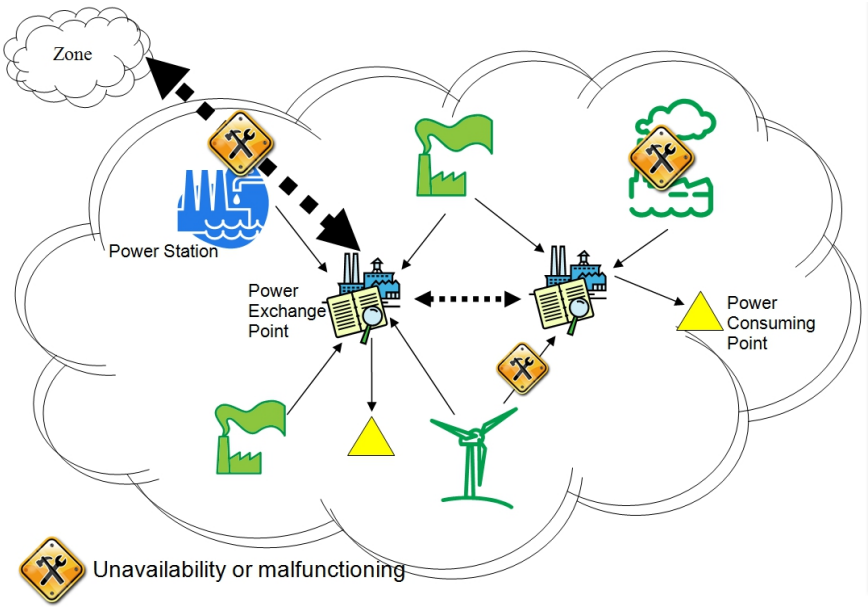

Fig. 4. Electric power scenario: an unavailability map

C. Agents in the system

The agents that have been modeled as part of UMMAS are:

- the PS Agent
- the PEP Agent
- the Zone Agent

The other elements of the domain (e.g. the physical intra- and inter-Zone links) have not decision capabilities and hence are not considered as agents in the system.

The PS Agent knows what are its fixed parameters, its supply functions and the PEPs to whom it is linked.

The PEP Agent knows its f_{con}, the PS that are linked to it, its f_{link} for each intra-Zone connection with other PEPs and inter-Zone connection with other PSs, the PEPs to whom it is connected to (either intra- and inter-Zone), the Zone Agent to whom it belongs to.

The Zone Agent knows all the PSs and the PEPs that belong to it and the neighbor Zone to whom it is directly connected.

D. The UMMAS-Lite prototype: Design and Implementation

The first UMMAS-Lite prototype release represents a demo version of the potential of UMMAS model and integrates some simplifications over the so far seen domain model in order to provide the fast design, implementation and testing of its main functionalities.

The UMMAS-Lite prototype is an event-driven simulator where an input event parameter is introduced in the system by a human operator and a stable and consistent system re-configuration after the event input is provided as output. Only after the system has reached a newly consistent state the human operator may introduce another disturbing event.

In the next paragraphs an overview of the simplifications adopted over the domain model is presented.

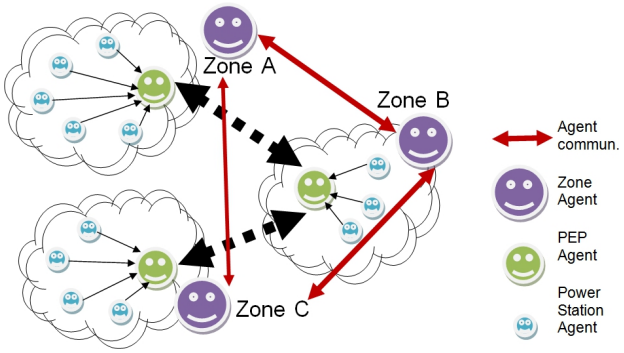

Fig. 5. The UMMAS-Lite Electric Power Network Topology

E. Simplifications over the Model

The following simplification with respect to the real domain model have been foreseen in UMMAS-Lite prototype:
Simplification over the electricity network topology: the electricity network topology given as input in UMMAS represents a simplification over the real one. Figure 5 shows the embedded domain model that is part of UMMAS-Lite.

The UMMAS-Lite Agents are:

- three Zone Agents (namely ZONE_A, ZONE_B and ZONE_C);
- three PEP Agents (namely PEP_A, PEP_B, PEP_C) where the letter stands for the Zone where they are located
- twelve PS Agents distributed over the three Zones as follows:
  - Agents from PS_1 to PS_5 belonging to ZONE_A, all linked to PEP_A;
  - Agents from PS_6 to PS_8 belonging to ZONE_B, all linked to PEP_B;

  - Agents from PS_9 to PS_12 for ZONE_C, all linked to PEP_C;
- a ClockAgent that represents the system timeline and has as main task that of synchronizing all agents at the beginning as well as at the end of each simulation.

An oversimplification on the number of PEPs, by considering only one PEP for each Zone in UMMAS-Lite prototype has as consequence that we can avoid considering the intra-Zone links that are not part of the system as it stands. This choice does not compromise the coherence of the model with respect to the real one.

Simplification over the input: both SR and AS can be purchased with a granularity of 1 MWh.

Simplification over the output: the PSs that are selected for the provision of energy are the cheapest in terms of best available SR price and, if this is not sufficient to cover the demand, the PS selected are also those able to provide the best available OS price. The location of PSs does not influence the choice so as the fact that a PS has a lower start-up ramp time than another, and hence could provide the necessary energy in a shorter time.

When a PS is asked to provide its best offer for SR and OS supplies, the calculation is done by considering the availability and prices at the time of the request, that it at time t_1 (the time when the event occurs), and sends all its offers at once.

When a decreasing transit capacity affects a transit limit towards a Zone at time t_1, the agents manage a transaction to reach a stable situation at time $t_1 + 1$ as if the decreasing capacity event occurs only for that time (and hence assuming that in the time instant $t_1 + 1$ the event does not occur anymore). In this way the agents do not worry about the fact that the situation would have been different if the event lasted until time t_1+1 (e.g. in case the specific event does not affect the quantity of energy that is traveling at time t_1 through the transit device, because the quantity of energy is less than the transit limit - the decreasing factor, the system does not need to reconfigure itself, even if at time $t_1 + 1$ the same condition would have implied a system reconfiguration, e.g. the quantity of energy traveling through the device at time $t_1 + 1$ results to be more than transit limit - the decreasing factor, and vice versa). This choice has been done for consistency reasons: we consider that an event happens at time t_1 and lasts until time $t_1 + 1$ excluded. And that at time $t_1 + 1$ the situation may be different and unpredictable before time $t_1 + 1$ occurs.

Simplification over the nature and frequency of events: each perturbation is introduced as a single event in each time slot for a single agent. This implies that at each time one single event may be managed by the system. An original configuration may be restored with the old data at time $t1 - 1$.

If an event is notified in the system at time t_1, the system will reach a stable configuration not later than at time $t_1 + max\_UpRamp + 1$, being $max\_UpRamp = max(startUpRamp(PS))$ such that PS is part of the network and has been selected for extra energy provision. In this way two events may occur at a time interval of at least $max\_UpRamp + 1$ time units.

All the simplifications introduced have the purpose of maintaining the system always consistent and of assessing that there is enough time to reconfigure it properly after an event is introduced and a solution strategy has been adopted.

F. UMMAS-Lite Implementation

In this Section we are going to give an account of the design and implementation details of UMMAS-Lite prototype. The prototype has been implemented using Jade 3.7 version.[3]

G. Agents Data

Each agent data is provided in Xml format and is read by each agent at the Jade Platform startup. An example of data available for the agent PS_1 is as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<PowerStation>
    <Id>PS_1</Id>
    <Type>thermic</Type>
    <PepId>PEP_A</PepId>
    <minPow>20</minPow>
    <maxPow>80</maxPow>
    <UpRamp>
 <UpRampMWh>10</UpRampMWh>
 <UpRampMWhMin>1</UpRampMWhMin>
    </UpRamp>
    <DownRamp>
        <DownRampMWh>10</DownRampMWh>
        <DownRampMWhMin>1</DownRampMWhMin>
    </DownRamp>
    <Baseload>
     <BaseloadTime>1</BaseloadTime>
     <BaseloadMWh>26</BaseloadMWh>
     <BaseloadTime>2</BaseloadTime>
     <BaseloadMWh>24</BaseloadMWh>
     ..

     <BaseloadTime>24</BaseloadTime>
     <BaseloadMWh>28</BaseloadMWh>
    </Baseload>
    <SR>
     <SRTime>1</SRTime>
     <SRQuantity>16</SRQuantity>
     <SRPrice>18.10</SRPrice>
     ...
     <SRTime>24</SRTime>
     <SRQuantity>16</SRQuantity>
     <SRPrice>20.71</SRPrice>
    </SR>
    <OS>
     <OSTime>1</OSTime>
     <OSQuantity>38</OSQuantity>
     <OSPrice>19.91</OSPrice>
     ...
     <OSTime>24</OSTime>
```

[3]http://jade.tilab.com.

```
      <OSQuantity>38</OSQuantity>
      <OSPrice>22.78</OSPrice>
    </OS>
</PowerStation>
```

H. UMMAS-Lite Classes

In what follows an account of the UMMAS-Lite Java package structure and a description of the main classes inside each package is provided.

- `org.ummaslite.utils`. This package contains all the classes that manage the UMMAS-Lite data structures, e.g. the Xml parsing and their read and write utilities. This package is exploited either for managing agents data and message contents (e.g. offers lists and so on).
- `org.ummaslite.data`. This package manages the data at agent level (PS, PEP and Zone). For each kind of agent a data object is instantiated.
- `org.ummaslite.message`. This package contains the `Message` class, that models all the message types that are exchanged among UMMAS Agents, as well as the `Helpers` class that simplify the `ACLMessage` object creation and contains the method to send and visualize its content.
- `org.ummaslite.behaviour`. This package contains the `AbstractAgent` class that extends the `jade.core.Agent` class and is used to model the agents `jade.core.behaviours.Behaviour` class according to the domain at hand. This classes are both implemented for each agent type.
- `org.ummaslite.agents`. This package contains the agents classes for each kind of agent foreseen in the specific domain (Clock Agent, Power Station Agent, Power Exchange Point Agent, Zone Agent).

I. Prototype Architecture

This Section presents a technical overview of the system from the point of view of how it works and of the agents interactions, states and transactions that structure the UMMAS-Lite prototype.

The system starts a scheduled simulation with the Clock Agent sending a message to every agent in the platform with the information of the simulation time unit. A second message is sent with the event that the user has introduced in the system to the address of the agent directly involved in the perturbation (e.g. if a user has introduced an increase in power consumption for a specified Zone then the receiver of the event message will be the PEP of that Zone, and so on). Based on the type of perturbation inserted the system starts different interaction patterns among the agents.

At the end of the simulation the Clock Agent keeps waiting until all the agents in the system send it a synchronization message to complete the interaction and to inform it that the system is now re-configured. Once received this synchronization message the Clock Agent will be able to start another simulation by giving a new start time unit and by sending a new event message to the agents involved. This last functionality has not yet being automatized in the prototype.

Each PS, PEP and Zone agent has been modeled and designed as a state machine with behaviour patterns, that is implemented inside each Agent's `Behaviour` class. Each state have as name the sender of a message and, inside each state, sub-states are designed, one for each kind of FIPA performative[4] and content of the message itself. In this way the sender and performative of the message are responsible for each Agent state, whereas transactions between Agent states are the messages sent by the Agent in reply to the performative of the message just arrived.

The communication protocol between agents in the system reflects the network topology of the domain. Hence, for example, PSs can send direct messages only to their PEP (that is the only entity with which they have a direct link to in the reality); the PEP may send direct messages only to its PSs and to its Zone; a Zone can forward such messages to its directly connected Zones, and so on.

J. Message-State Behaviours

In this Section we are going to illustrate how our state machine behaviour patterns works for each Agent. Each message received is responsible for a particular transaction and hence strictly drives the overall simulation case interactions among the agents.

The FIPA performatives that we use in the message exchanged among the UMMAS-Lite agents are the following:

- INFORM: is the performative used in the time unit message, the synchronization message, and the event message;
- CFP: is the call for proposals message sent to PSs when requesting an offer for energy;
- PROPOSE: is the offer message that each PS sends in response to the CFP;
- ACCEPT: is the acceptance notification of the offer sent to PSs when accepting their offer;
- CONFIRM: is used by the PS to confirm the energy delivery;

The *Clock Agent* state machine pattern, as depicted in Figure 7, is the Clock Agent behaviour modeled according to two possible states: the *InitState*, where it sends the time unit message to all other agents in the system and the event message to the agent involved in the unavailability (a PS Agent in case the event wants to simulate a decrease in its baseload; a Zone Agent in case a decrease in the limit transit capacity has occurred; a PEP Agent in case an increase in energy consumption needs to be managed); the *SynchState* in which it waits for a synchronization message from all the other agents and, after having received it, a new event may be managed (we recall that this last transaction is not implemented in the prototype).

The *Power Station Agent* state machine pattern is illustrated in Figure 8. The initial state for the two behaviour patterns

[4] http://www.fipa.org/.

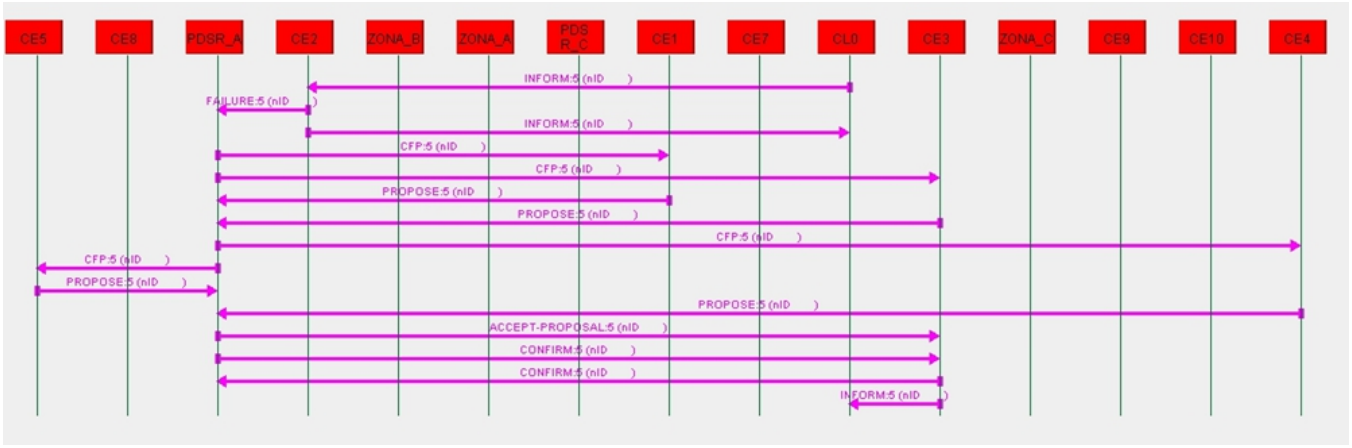

Fig. 6. A Simulation Example: the Power Station Unavailability

Fig. 7. The Clock Agent state machine pattern

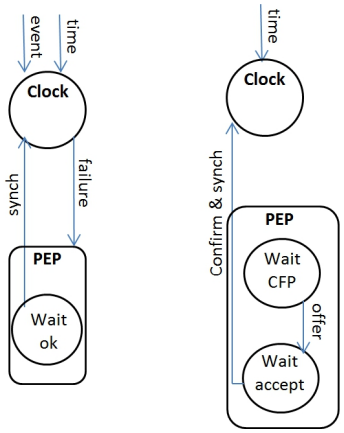

Fig. 8. The Power Station Agent state machine patterns

depicted is the *Clock State*, where it waits for the time unit message as well as the event message (in the behaviour pattern shown on the left side of the picture), in case this affects it directly. As the event message in this case can be only one between the unavailability or the malfunctioning of the PS itself, it sends a failure message to its PEP and sends the synchronism message to the Clock Agent that brings it into its initial state. In the behaviour pattern shown on the right side of the picture the PS Agent receives a CFP from its PEP, sends its offers and waits for acceptance. If the proposed offer is accepted it sends a confirmation message to its PEP, provides the extra energy and sends a synchronizing message to the Clock Agent.

The *Power Exchange Point* state machine pattern is shown in Figures 9 and 10. In the first behaviour pattern the PEP receives from the Clock the time unit message as well as an event message about an increase in the consumption of energy that involves it directly. The PEP sends a CFP message to all its PSs and to its Zone and waits for energy offer proposals. When the offers arrive the PEP ranks them according to the best price available and sends an accept message to each of its PSs of which it has accepted the offer, and to its Zone in case an offer from a PS of another Zone has been accepted. The PEP waits for a confirm message coming from all the PSs that are in charge of supply the extra energy just bought, and then it sends a synchronization message to the Clock. In the second behaviour pattern the PEP receives a CFP coming from its Zone (that is the case in which another Zone is requesting offers for power supply) and forwards the message to its PSs. It waits for offers to be forwarded to its Zone (for dispatching them to the original requesting Zone), and waits for acceptance messages (that will come from its Zone in case the original requesting Zone receives them from its PEP, that is in charge of collecting all the offers it needs and accepting some of them). If accepting messages arrive it forwards them to all its PSs that are involved and waits for confirmation messages from them. It finally forwards them to its Zone (that again is in charge of forwarding them to the accepting Zone that will forward them to its PEP in order to complete the transaction). It then sends a synchronization message to the Clock agent.

The *Zone Agent* state machine pattern can be visualized in Figures 11, 12, and 13 . The Zone may receive from the Clock an event relative to the transit limit damage or decreasing capacity. In this case the Zone sends a message to its PEP in order to inform it of the decrease in energy supply. The PEP will act accordingly (see PEP first behaviour pattern). In the second picture the Zone forwards a CFP from its PEP to

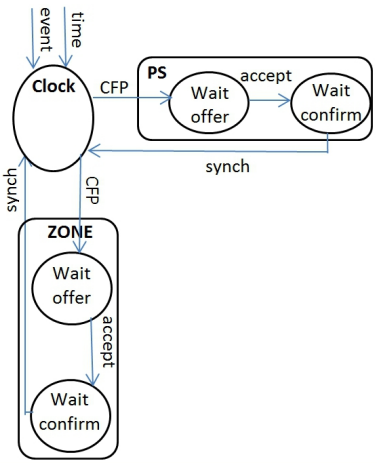

Fig. 9. The Power Exchange Point Agent state machine pattern 1

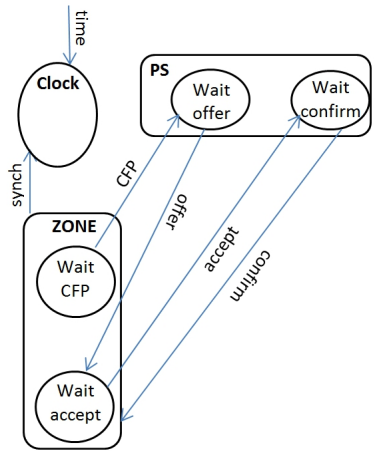

Fig. 10. The Power Exchange Point state machine pattern 2

another Zone, waits for the offer message and forwards it to its PEP. When the PEP sends an accept message it will forward it to the original Zone that has done the requests and waits from it the confirmation messages that it will forward to its PEP. The third picture represents the symmetric of the previous behaviour pattern. Here the Zone receives a CFP from another Zone, forwards it to its PEP, waits for offers and forwards them to the other Zone, waits for acceptance and forwards it to its PEP, and finally waits for the confirmation message and forwards it to the other Zone.

A third common behaviour pattern for all the agents is determined whenever the agent does not participate at all in the simulation case at hand. It then remains in the initial state (the Clock State) and sends to the Clock Agent a synchronization message.

K. A UMMAS-Lite Simulation Run

In Figure 6 a simulation example of UMMAS-Lite in action is reported in form of Jade Sniffer Agent Snapshot. The simulation case reported is that of an unavailable or malfunctioning Power Station that sends its unavailability

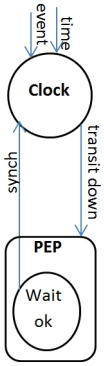

Fig. 11. The Zone Agent state machine pattern 1

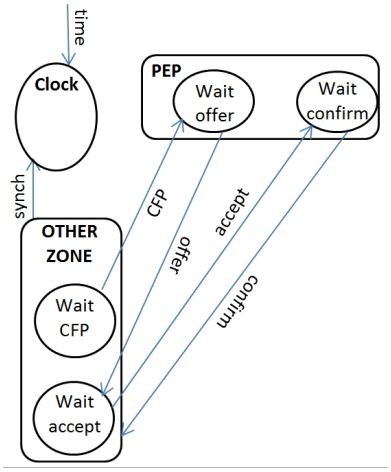

Fig. 12. The Zone Agent state machine pattern 2

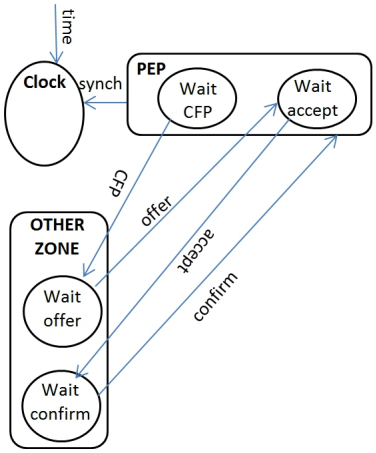

Fig. 13. The Zone Agent state machine pattern 3

message to its PEP and the transaction is conducted within a single Zone boundaries. The names of the agents in the upper boxes of the figure are in Italian. We provide hereafter their translation in English:

- CE stands for PS;
- PDSR stands for PEP;
- ZONA stands for Zone.

IV. Related and Future Work

The research collaboration activity between scientists of DISI and engineers of Elsag Datamat has been centered upon whether a MAS approach and technology is a well suited solution to the analysis and modeling of a high level system architecture where different entities with different roles, autonomous and distributed over a network, may cooperate in a complex scenario, as also discussed in [10]. In particular, the synergy efforts of this activity have been focused on modeling unavailability management simulation systems, where some unexpected malfunctioning or unavailability event may occur and compromise the system consistency, until a solution to the problem is found and the system is restored to a consistent state. Because of the relevance of the electricity domain and of the many challenging problems it raises, we opted for using the Italian electric power market as inspiring scenario.

Differently from our prototypical system which was aimed at a feasibility study in a quite general scenario, many fully-fledged agent-based tools developed ad hoc for simulating electricity markets exist. Among them we may cite SEPIA [7], EMCAS [2], [9], STEMS-RT [3], NEMSIM [5], [6], whose detailed analysis and comparison has been carried out in [11]. Although most of them include agent roles which are similar to those that we defined for UMMAS (just to make an example, in SEPIA there are Zones, Physical Generators, Generation Companies, Generator of Last Resort, Consumer Load, Consumer Companies, Transmission System, and Transmission Operator), the physical model for the load of individual consumers and the economic model the electricity market are definitely more sophisticated than in UMMAS.

Also, some of them are developed using general-purpose agent-based modeling and simulation (ABMS) tools like SWARM[5], Repast[6], MASON[7], and NetLogo[8], whereas UMMAS is implemented using JADE.

The choice of JADE as the tool for carrying out the research on UMMAS was due to the background of the academic partners that felt more comfortable with a tool they already knew, but also to an experience they had in a similar domain where the NetLogo ABMS tool proved not suitable for their purposes [1], and to the lack, at the time the research activity started, of settled methodologies for ABMS.

Since integrated methodologies able to seamlessly guide domain experts from the analysis of the system under consideration to its modeling and analysis of simulation results recently appeared [4], we are considering to re-design and re-implement the UMMAS system using an ABMS system instead of JADE.

As a concluding remark, we may state that the academic-industrial collaboration was fruitful and the engineers from Elsag Datamat were satisfied of the results obtained by the academic partners. In particular, both the ease of analyzing a complex scenario following an agent-oriented approach and the closeness of the architecture of the resulting MAS to the real scenario architecture were extremely appreciated.

DISI and SELEX Elsag are currently working at a generalization of both the model behind UMMAS and the implemented MAS, in order to make them applicable to other domains of industrial interest.

[5]http://www.swarm.org/.

[6]http://repast.sourceforge.net/repast_3/.

[7]http://www.cs.gmu.edu/~eclab/projects/mason/.

[8]http://ccl.northwestern.edu/netlogo/.

Acknowledgment

The work of the first and second authors is supported by the Italian research project Iniziativa Software CINI-FinMeccanica, and in particular their work is framed within the "Software Agents in Support of Complex Systems Interoperability" Laboratory.

References

[1] D. Briola and V. Mascardi. Design and implementation of a NetLogo interface for the stand-alone FYPA system. *In this volume*.

[2] G. Conzelmann, M. North, G. Boyd, R. Cirillo, V. Koritarov, C. Macal, P. Thimmapuram, and T. Veselka. Simulating strategic market behavior using an agent-based modeling approachresults of a power market analysis for the midwestern United States. In *Proc. of the 6th IAEE European Energy Conference on Modeling in Energy Economics and Policy*, 2004.

[3] R. Entriken. Using automated agents with probe: interface requirements specification. Technical Report 1012157, EPRI, Palo Alto, 2005.

[4] A. Garro and W. Russo. easyABMS: A domain-expert oriented methodology for agent-based modeling and simulation. *Simulation Modelling Practice and Theory*, 18(10):1453–1467, 2010.

[5] G. Grozev and D. Batten. NEMSIM: practical challenges for agent-based simulation of energy markets. In *Proc. of CSIRO Complex Systems Science Annual Workshop. CSIRO Manufacturing and Infrastructure Technology*, 2005.

[6] G. Grozev, D. Batten, M. Aanderson, G. Lewis, J. Mo, and J. Katzfey. NEMSIM: agent-based simulator for australia's national electricity market. 2005.

[7] S. Harp, S. Brignone, B. F. Wollenberg, and T. Samad. SEPIA: a simulator for electric power industry agents. *IEEE Contr. Syst. Mag.*, 20(4):53–69, 2000.

[8] N. R. Jennings, K. Sycara, and M. Wooldridge. A roadmap of agent research and development. *Autonomous Agents and Multi-Agent Systems*, 1(1):7–38, 1998.

[9] V. Koritarov. Real-world market representation with agents. *IEEE Power Energy Mag.*, 2(4):38–46, 2004.

[10] S. McArthur, E. M. Davidson, V. M. Catterson, A. L. Dimeas, N. D. Hatziargyriou, F. Ponci, and T. Funabashi. Multi-agent systems for power engineering applications – part i: Concepts, approaches, and technical challenges. *Power Systems, IEEE Transactions on*, 22(4):1743 –1752, nov. 2007.

[11] Z. Zhou, W. Chan, and J. Chow. Agent-based simulation of electricity markets: a survey of tools. *Artificial Intelligence Review*, 28:305–342, 2007. 10.1007/s10462-009-9105-x.