Self-Balancing Distributed Energy in Power Grids: an Architecture based on Autonomic Computing

Jennifer Pérez, Jessica Díaz, Carlos Vidal
Technical University of Madrid (UPM), CITSEM, E.U. Informática, Madrid, Spain
jenifer.perez@eui.upm.es, yesica.diaz@upm.es, cvidal@syst.eui.upm.es

Daniel Rodríguez, Diego Fernández
Indra Software Labs, Spain
C/ Julián Camarillo 16
28037 Madrid, Spain
dralvarez@indra.es, dfernandez@indra.es

Abstract

The management of distributed and intermittent energy generation is a critical challenge within the power domain. This challenge has emerged due to the increase of distributed and renewable energy resources in power networks. Smart Grids are a solution to integrate intermittent and dispersed renewable energy and to increase energy efficiency through the introduction of Information and Communication Technologies. However, Smart Grids require new and innovative models, and software architectures that enable Smart Grids to operate in an intelligent and self-managing way. To deal with the intelligent operation of power grids, this paper presents a reference architecture for autonomic power grids. Specifically, this paper focuses on the capability of self-balancing distributed and intermittent energy. It illustrates how this self-balancing capability is implemented and its usefulness for a scenario of a microgrid located in a real setting, in the south of the Spanish region of Ciudad Real.

1. Introduction

One of the goals of the horizon 2020 is to increase the percentage of renewable energies to 20% [38]. The incorporation of distributed and renewable energy generation makes managing power networks more complex, and it is necessary to deal with new challenges. One of these challenges is that the deployment of distributed energy resources with a distributed energy generation requires a shift from current centralized energy infrastructures towards more distributed ones [25]. In addition, renewable energy generation has an inherent variability of energy production depending on time frames (day/night) or weather, among other factors, which makes balancing supply and demand a challenge [3].

Smart Grids are presented as a solution to integrate intermittent and dispersed renewable energy and to increase energy efficiency through the introduction of Information and Communication Technologies (ICT) [38]. A Smart Grid is defined as «an upgraded electricity network in which two-way digital communication between supplier and consumer, intelligent metering and monitoring systems have been added» [8]. Moreover, Smart Grids promote the integration of traditional and renewable energy resources in distributed, open, and self-managed ways. In this regard, Lasseter [23] framed Smart Grids as a set of microgrids that are connected to the main grid, i.e., grids that are designed to provide electrical and/or thermal energy for local loads and communities. But currently, it is possible to also speak about Large-Scale Smart Grids that increase the electricity service to millions of customers by using intelligent metering [44][45]. Large-Scale Smart Grids are composed of intelligent microgrids that are constituted by a broad range of energy resources (see Figure 1), from large generating systems (e.g., nuclear power plants, or hydro power plants) to smaller generating systems (e.g., small solar farms, or wind generators), all of them operating as a single system providing both power and heat [11][24]. This decentralization of energy resources and generation provides scalability and robustness, facilitates access to the energy market for prosumers, and enables bi-directional flow power generation (generation - transport - distribution – consumption) [26].

Smart Grids require new models and software architectures that enable two-way exchange of power and information between suppliers and consumers (or prosumers) based on the introduction of intelligent communication, monitoring, control, and management systems in their microgrids [1]. Therefore, the deployment of ICT infrastructures in power grids requires the design of software architectures using innovative models [8] to provide intelligent properties.
In this paper, we present a reference architecture that is based on Autonomic Computing [9][22] and standards to support the intelligence and autonomy of the grid, and to guarantee interoperability, respectively. This reference architecture has been designed from our experience in the Smart Grid domain obtained participating in several large projects about power energy. In particular, this paper presents how Autonomic Computing facilitates the deployment of policies that allow the grid to self-balance the demand/supply of energy, by taking the intermittent power generation of renewable energies and the storage systems into account.

Figure 1. Overview of a Smart Grid. From [32]

This contribution is based on the experience of a real power microgrid setting in the south of the Spanish region of Ciudad Real. This scenario is used for illustrating how our autonomic reference architecture can be deployed in a real setting and how it allows for self-balancing the demand/supply of the energy network in order to provide supply with the maximum time.

The structure of the paper is as follows: Section 2 presents an overview of Autonomic Computing. Section 3 details the autonomic reference architecture as well as the technologies that support its implementation. Section 4 describes how the deployment of the architecture is feasible in a real setting. Finally, conclusions and further work are presented in Section 5.

2. Autonomic Computing

Autonomic Computing (AC) [9][22] emerged as a solution to deal with the increasing difficulty of managing today’s computing systems, and the limitation of human capabilities to deal with this complexity. IBM proposed the construction of AC Systems [22] in its manifesto [9] by defining them as software systems that mostly operate without human or external involvement according to a set of rules or policies.

The autonomy of systems is achieved through one or more facets of self-management, collectively named as self-* properties [1][5]. Among the broad range of self-* properties studied in the literature, IBM distilled four main properties that may wrap up the others, establishing that a computing system that possesses at least one of these four properties is autonomic. These properties are the following [2][22]:

- **Self-Configuration**: Automatic installation and configuration of system components in a manner responsive to the needs of the platform, the user, or the organization.
- **Self-Optimization**: Automatic optimization of the usage of resources according to a set of criteria regarding user needs, platform constraints or organization policies.
- **Self-Healing**: Automatic detection, diagnose and reparation of localized issues in system components, by using knowledge about the system configuration or log information.
- **Self-Protection**: Automatic configuration and tuning to achieve security, privacy, function and data protection goals.

These properties are provided by means of autonomic elements, which implement the high-level policies that define this self-* behavior. Therefore, autonomic elements are the fundamental building blocks for designing self-managed systems based on AC [22]. An autonomic element mainly consists of one or more managed resources, which represent any software or hardware resource that is provided with autonomic behavior; and a closed control loop, which periodically supervises the resources and takes the appropriate actions to achieve the set of established high-level policies (see Figure 2).

Figure 2. Internal Structure of an Autonomic Manager. From: [39]

In order to support its autonomic management, each resource provides a manageability interface. It is composed of: **sensors** (also called probes or gauges in the literature), which are services that provide information about the current state of the resource, and **effectors**, which are services that modify the state of
the resource. By using the services provided by sensors and effectors, the closed control loop: (i) monitors the state of the resource(s) and filters the accumulated data; (ii) analyses and correlates data to detect significant symptoms which may need corrective actions; (iii) plans the set of actions to change the current state of the managed resource(s) to a different state, according to a set of predefined goals; and (iv) executes the change in the plan through the effectors of the managed resource(s). This set of actions (Monitor, Analysis, Plan, and Execute) is operated over a knowledge base conforming what is generally referred to as the MAPE-K loop [9][41] (see Figure 2). The MAPE-K loop is a model that provides the advantage of isolating the main concerns that any autonomic process has to provide. This allows the hierarchical composition of autonomic elements (see Figure 3).

Figure 3. Reference Architecture for AC. From: [12]

The AC reference architecture [12] is essentially a three-layered architecture (see Figure 3) consisting of a hierarchy of: (i) resources, (ii) autonomic elements managing these resources, and (iii) orchestrators that manage other autonomic elements. This hierarchy, where higher level managers orchestrate lower level managers or resources, is the most common and natural arrangement, since it reflects how organizations generally work.

AC initiative has undergone a major expansion producing results in balancing vision, architecture, new techniques and applications in different domains, such as Service-Oriented Architecture (SOA) [6], ubiquitous computing [12], or cloud computing [21]. Several works emphasize the need of AC to deal with the new challenges of Smart Grids. The work of Javed & Arshad [17][18] shows that the application of AC to Smart Grids can provide important improvements and advantages to the power domain. IBM’s white paper “Smart Micro Grid: IT Challenges for Energy Distribution Grid Operators” [11] identifies the need of autonomous elements within microgrids. Finally, other works point out AC as a design solution for Smart Grids, specifically for some characteristics such as security [33]. Our reference architecture has been designed starting from these promising insights of using AC for software architecture design of Smart Grids.

3. A Reference Architecture based on AC for Designing Smart Grids

One of the main challenges of Smart Grids is the design of their architectural components and interfaces while guaranteeing the interoperability among components at different layers of the architecture through the specification of communication standards [16]. In this regard, we present the reference architecture that has been designed from our previous experience in the Spanish national project ENERGOS (Technologies for the Automated and Smart Management of the Future Distributed Power Networks [7]), and the two ITEA projects: NEMO & CODED (NEtworked MOntoring and COntrol, Diagnostic for Electrical Distribution) [27] and IMPONET (Intelligent Monitoring of Power NETworks) [14]. This architecture provides the software architecture infrastructure and the technologies for implementing AC, which has been designed from the experience in these projects. As a result, this architecture is independent of any case study and it could be considered a reference architecture for designing Smart Grids based on AC. In addition, from these projects, we have collected a set of premises that the design of the reference architecture should follow. These premises are set out below:

1. The microgrids that constitute the Smart Grid are connected to the main power network and are managed as another resource depending on the demand/response of energy (see Figure 4). This fact allows microgrids to operate in Connected mode or Island mode. The Connected mode is when the microgrid works connected to the main power network [4]. It is the mode in which the microgrid works by default. On the other hand, the Island mode is when the microgrid works when it is disconnected from the main power network [32].

Figure 4. Smart Grids from the Power Network Perspective
2. Contrary to the current power networks that are designed to operate according to a unidirectional flow (generation - transport - distribution – consumption), microgrids enable the operation in a bi-directional flow.

3. Software Architecture design has provided power resources software that enables them to perform their intelligent and autonomic management.

### 3.1 Infrastructure: Architectural elements

The software architecture infrastructure of the power grid is composed of a set of components [39] and connectors [35] that properly orchestrated establish its autonomous behavior. This autonomous behavior is mainly supported by real time, smart metering, and monitoring capabilities. On the one hand, real time capabilities allow software architecture to acquire, store, distribute, process and analyze data in real-time, and to flexibly connect and disconnect resources. On the other hand, smart metering and real time monitoring capabilities enable [7]: (i) increases in the automation and coordination of producers, providers and consumers to balance supply and demand, (ii) a response to the current market energy needs in real time, (iii) a better understanding of the electricity consumption, and (iv) a reaction in real-time to sudden changes of the aggregated generation profile, in order to balance supply from intermittent renewable resources. As a result, this autonomous behavior requires that the components and connectors that constitute the software architecture will be able to: (i) continuously monitor the network with real time or quasi real time data processing, (ii) analyze and respond to massive amounts of information that are received from the network and to store them, as well as the signals that will have to be transmitted to the devices in the field, (iii) reason and make decisions based on the run-time or persistent data, and (iv) perform the execution of the decisions made. In order to provide this autonomous behavior to the architectural elements (components/connectors) of the software architecture, they have been supplied with the capabilities of autonomic systems. As a result, the architectural elements that require autonomous behavior follow the AC model MAPE-K [9][41] (see section 2). This is achieved due to the fact that the AC model MAPE-K is encapsulated in a software component called autonomic manager. This autonomic manager is included in the architectural elements that require this intelligent and autonomous behavior - Figure 5. A illustrates how the autonomic manager is graphically represented in the architecture.

In software architecture, components are those computational elements that capture and execute the functionality of the software system [39] whereas connectors are coordinators between other architectural elements [35]. As a result, the components of power grid architecture are deployed in storage systems, producers, and consumers. In addition, components are deployed in those prosumers in which policies are internally managed, i.e. their decisions are made independently of the architectural elements that they are connected to. On the other hand, connectors are deployed in transmission resources, demand/supply managers, storage managers, or prosumers that requires interaction with other architectural elements to perform their policies. Connectors are characterized for implementing the autonomic manager that provides the intelligence and the smart capabilities to the grid in order to be an autonomic microgrid.

![Figure 5. Graphical representation of the Autonomic Manager and the GridController](Image)

In order to fit the composition structure that autonomic systems require (see Figure 3) and due to the fact that there is not a manual manager in microgrids with both intelligence and autonomy, it is required a complex component to orchestrate the different autonomic managers that are deployed in intelligent and autonomous microgrids by following a hierarchical composition. This hierarchical composition and the need of this complex component are key issues in guaranteeing the scalability of the software architecture and the first premise of the software architecture construction (see Section 3), respectively. This complex component is deployed in the resource of the microgrid that is connected to the main power network (see Figure 4). This component is the software connection point between the main power network and the microgrid that allows the power network to manage microgrids as another resource. This complex component is called GridController (see Figure 5.B).

The GridController is composed of two complex components connected by a connector. These components implement global policies about the microgrid behavior. In particular, they implement the policies and actions that need to be executed when the microgrid is operating in ConnectedMode or IslandMode, respectively (see Figure 5.B). Finally, the
connector implements the required policies for changing from *Connected mode* to *Island mode*, and vice versa. The policies of these architectural elements are implemented through autonomic managers to guarantee their autonomic execution in run-time (see Figure 5.B).

### 3.2 Technologies: SPEED Nodes

The autonomic managers allow the microgrid to behave as an autonomic system. Therefore, they must be implemented using technologies that make this distributed, open, and self-managed behavior feasible. In this section, we present the implementation of these autonomic managers, which are called SPEED nodes.

The technologies that SPEED nodes make use of are the following:

- **The OMG Data Distribution Service for Real-Time Systems (DDS)** [28]. DDS is a data-centric publish/subscribe communication middleware specially created for dealing with distributed systems. DDS’ main benefits are: low coupling between entities, high performance, dynamic scalability, deterministic data delivery, and high parameterized Quality of Service (QoS).

- **A component for Complex Event Processing (CEP)** [30][31][42] in combination with an adapted Event Processing Language (EPL). These technologies allow SPEEDs to identify and react to patterns in real time, despite the large amount of data to be processed.

- **Semantic technologies** that are applied to: (i) unambiguously describe the grid topology through an OWL [37][43] ontology, (ii) guarantee the interoperability and communication of devices based on different electrical standards (e.g. IEC 61850 [13], CIM [15], etc.), and (iii) offer the possibility of reasoning and inferring knowledge by means of Semantic Web Rule Language (SWRL) [10] and Pellet [36].

In addition to these technologies, a framework is required that allows hot deployment. To address this need, the OSGi (Open Services Gateway Initiative) framework was chosen [29].

The implementation of the SPEED node smoothly integrates all these technologies. The SPEED core consists of the OSGi container and the software deployed in it. This software is a set of software components. Each software component deployed in an OSGi container is called a “bundle”. Therefore, *bundelized* versions of the aforementioned technologies are needed (see Figure 6).

The MAPE-K loop is implemented in SPEED nodes by four bundles: Monitoring, Analysis, Planning and Execution (see Figure 6). These bundles make use of the SPEED technologies as follows (see Table 1):

- **The Monitoring bundle** is in charge of performing the tasks of data reception, making use of the DDS middleware, and grouping into symptoms through the CEP technology.

- **The Analysis bundle** is in charge of analyzing the symptoms detected by the Monitoring bundle. The output of this analysis is to determine which policy has to be executed. The policy is triggered by applying certain Event-Condition-Action (ECA) Rules.

- **The Planning bundle** is in charge of defining the execution of the actions that the policy triggered by what the Analysis bundle comprises. In order to create the list of actions to execute, the Planning bundle needs broad knowledge about the grid topology and the status of its components. This bundle is able to perform reasoning over the ontology by using Pellet for the definition of the grid topology or applying SWRL. The Bundle also makes use the Jena framework for loading the ontology in memory [19], and the Jess rule engine for inferring knowledge through SWRL rules triggering [20].

- **The Execution bundle** performs the execution of the actions required by the Planning Bundle.

Finally, it is important to emphasize that those components without intelligence and autonomy, and those devices with few capacities such as processor, RAM memory, among others, implement components with a reduced MAPE-K loop, i.e. a SPEED node with only the Monitoring and Execution bundles.
Table 1. Mapping between the MAPE-K model and SPEED nodes

<table>
<thead>
<tr>
<th>Implemented By</th>
<th>Uses</th>
<th>Input</th>
<th>Output</th>
<th>Interacts with</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monitoring</td>
<td>Monitoring Bundle</td>
<td>DDS and CEP</td>
<td>Data from sensors</td>
<td>Data grouped in symptoms</td>
</tr>
<tr>
<td>Analysis</td>
<td>Analysis Bundle</td>
<td>ECA Rules</td>
<td>Symptoms</td>
<td>Policies of the Smart Grid</td>
</tr>
<tr>
<td>Planning</td>
<td>Planning Bundle</td>
<td>Jess, Jena, Pellet</td>
<td>Policies of the Smart Grid</td>
<td>Actions to be performed</td>
</tr>
<tr>
<td>Execution</td>
<td>Execution Bundle</td>
<td>-</td>
<td>Actions to be performed</td>
<td>-</td>
</tr>
</tbody>
</table>

4. A scenario: Self-balancing distributed and renewable energy

Smart Grids with autonomous behavior provide advantages and important advances in the management of power networks. Autonomic Smart Grids are those whose microgrids are implemented as Autonomic Systems for being self-managed. One of these advantages is power grid capability of self-balancing the supply and demand of a network, which is especially critical in those contexts where renewable resources are present.

In this section, we present a real power grid setting in which there are renewable resources. This power grid is located in the towns of Santa Cruz de Mudela (SCM), Viso del Marqués (VM), and Almuradiel (ALR), in the south of the Spanish region of Ciudad Real (see Figure 8). In particular, we describe our experience in a scenario managed by the project ENERGOS [7] in this power grid setting. This scenario consists of the self-management of the SCM-ALR microgrid when the power supply of the main network falls down. This scenario description allows us to illustrate how the designed autonomic reference architecture with its SPEED nodes (see section 3) can be deployed in a real setting to provide power grids the intelligence of self-balancing the demand/supply of the network in order to guarantee the maximum supply time. In addition, this scenario allows us to demonstrate that microgrids are able to self-balance demand/supply through an architecture that implements the autonomic computing model [9][22].

4.1 Power Network: Topology Description

The Smart Grid of SCM and ALR (see Figure 8) is a semi-urban and rural power network, which is divided into two feeder sections, SCM703 and ALR703, connected by an exchange point (see Figure 7).

![Figure 7. Schema of the power network at the south Ciudad Real](image)

This power network has installed 119 km of Medium Voltage line (MV) with 14225kVA of power. In addition, it has established 4MVA as the maximum of energy demand (see Figure 8). Energy resources are distributed in the SCM and ALR feeder sections as follows:

![Figure 8. Geographical Location of the power network of the SCM and ALR](image)

2403
• Feeder section SCM703: (i) Substation SCM; (ii) 11 TTCC that represent the 637 customers of low voltage (LV). The customers are divided into the 11 TTCC in such a way that there are 10 TTCC with 58 customers per each and 1 TTCC with 57 customers; (iii) 32 TTCC that represent the 32 customers of MV; (iv) 1 Section center of MV (v) 1 Storage system; (vi) 1 Storage manager; (vii) 1 Prosumer: An industrial unit of microgeneration; and (viii) 41 demand managers.

• Feeder section ALR703: (i) Substation ALR; (ii) 29 TTCC that represent the 1679 customers of LV. The customers are divided into the 29 TTCC in such a way there are 26 TTCC with 58 customers each and 3 TTCC with 57 customers each; (iii) 21 TTCC that represent the 21 customers of MV; (iv) 2 Section centers of MV; (v) 1 Photovoltaic generation plant of MV; (vi) 1 Photovoltaic generation plant of LV; (vii) 1 Storage system; (viii) 1 Storage manager; and (ix) 52 Demand managers.

4.2 Architecture Reference Deployment

The software architecture of the SCM and ALR power network was designed following the reference architecture based on AC (see section 3). Each of the energy resources of the network (see section 4.1) has deployed at least one architectural element. Figure 9 illustrates the resulting architecture composed by a set of components and connectors interconnected among them. The name of the components and connectors represent the hardware energy devices where they are deployed in order to facilitate their comprehension.

The GridController (see Figure 5.B) is the software connection point between the main power network and the microgrid (see section 3.1). Therefore, it is deployed in the resource that the microgrid is connected to in the main power network (see Figure 9). The GridController is in charge of monitoring the events that occur in the main power network and that are relevant for the microgrid; such as power network failure, or losses of power or supply. In addition, it encapsulates the high-level policies that establish the microgrid’s actions depending on the events that occur in the power network (see section 4.3). As a result, the GridController contains a SPEED node that implements an autonomic manager. This allows the GridController to monitor the network, analyze data, define policies, and then execute the necessary actions through its components ConnectedMode and IslandMode, and its connector MutingMode (see Figure 9). Finally, it is important to mention that the GridController is the complex component that connects the feeder sections SCM703 and ALR703 through the interconnections of the substations SubstationSCM and SubstationALR of high/medium voltage (HV/MV) of each feeder section. This connection is performed through an exchange point that is a connector of the architecture, called Exchange Point (see Figure 9). The Exchange Point connector acts as a mere transmitter or an element able to block the energy supply in one of the feeder sections when it is required by the high level policies of the grid. As a result, the Exchange Point has a SPEED node to implement the policies related to isolating one feeder section of the network due to efficiency or balancing policies. In addition, there are other three connectors that allow the feeder sections to separate their different kinds of customers, LV and MV.

Figure 9. Software Architecture of the Smart Grid of SCM and ALR

2404
These connectors are deployed in the section centers. As a result, the connector CnctSection1 separates the MV customers of the feeder section ALR703, the connector CnctSection2 separates the MV customers of the feeder section ALR703, and the connector CnctSection3 separates the MV customers of the feeder section SCM703. This organization allows CnctSections to apply policies to the supply level for a specific kind of customer. Therefore, this kind of connector has a SPEED node to make it feasible to execute these policies during run-time.

The 2316 LV customers are designed as a 2316 components named as Customer1 to Customer2316 (see Figure 9). These components just monitor the network and execute the commands of other architectural elements. In order to guarantee the balance of the network, customers are coordinated by the demand managers.

These demand managers are the connectors of the architecture that have the local policies that allow balance the demand/supply of the components that they keep connected. These connectors are named as CnctMngmtDS, (CnctMngmtDS1...CnctMngmtDS63) and had a SPEED node to implement the local policies. They communicate with customer through the system transformation connectors. These system transformation connectors (CnctTrans1 ... CnctTrans640) are responsible of forwarding the data between CnctMngmtDS and Customers. In order to design a scalable architecture, each system transformation connector has 57 or 58 customers connected and each demand manager connector (CnctMngmtDS) that coordinates approximately 5 system transformation connectors (see Figure 9).

In addition, there are two software components for managing the two photovoltaic generation plants; PhotoPlant1 and PhotoPlant2 (see Figure 9). These components provide energy production data to the demand managers that manage them, CnctGestDm52 and CnctGestDm30. These demand managers CnctGestDm52 and CnctGestDm30 are in charge of: (i) deciding, based on the policies, if the energy produced by the photovoltaic generation plant must be used for supply or storage; (ii) deciding the level of production of the photovoltaic generation plant; and (iii) deriving the energy production to the storage system, Storage1.

The storage systems Storage1 and Storage2 are designed following the dashboard pattern [40] in order to guarantee the independence between the storage itself and the access and management of system. For this reason, each component has an associated connector that manages the storage and controls its access CnctStorage1 and CnctStorage2.

Finally, the industrial unit of microgeneration is managed by the component MicroGeneration that is in charge of managing the boilers generation in terms of: (i) deciding, base on the policies, how the energy should be used, (ii) deciding the level of production, and (iii) deriving the energy production to the storage systems, Storage2. To manage and implement these policies, the component MicroGeneration also has a SPEED Node (see Figure 9).

4.3 Scenario Description

The software architecture of the power network SCM and ALR allowed us, in the context of the ENERGOS project [7], to simulate the execution of a scenario in which the self-balancing of intermittent renewable energies within the Smart Grid was a critical issue. Specifically, the scenario reproduced the failure of the main power network and the need to guarantee supply to the customers of the network in as little time as possible. Tables 2 and 3 detail the execution of the scenario decomposed into steps, the policies that were necessary to execute the autonomic elements of architecture, and the role each architectural element played. Table 2 details the main scenario, whereas Table 3 describes one of the sub-scenario’s associated with one of the feeder sections. These scenarios show the execution steps that guarantee the balance of the demand/supply of energy. This has been made possible by the execution of policies implemented in the SPEED nodes deployed in the resources of the SCM/ALR feeder sections.

Table 2. Main Scenario

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
<th>Participants</th>
<th>Energy %</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial State</td>
<td>The Smart Grid is connected mode and is continuously monitoring the main power network.</td>
<td>Main Power Network</td>
<td>Smart Grid</td>
</tr>
<tr>
<td>1</td>
<td>A system transformation of the main power network is on fire and the energy supply falls down.</td>
<td>Main Power Network</td>
<td>Smart Grid</td>
</tr>
<tr>
<td>2</td>
<td>The sensor that connects the Smart Grid to the main power networks detects the supply fail and this event is intercepted by the MicrogridController.</td>
<td>MicrogridController</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>The MicrogridController changes its mode from connected mode to island mode by following the ECA that specifies this action when necessary.</td>
<td>MicrogridController</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>The MicrogridController notifies the change to island mode to the connectors SubstationALR and SubstationSCM through the Border connector with the purpose of forwarding the notification of the change to rest of elements of the Smart Grid.</td>
<td>SubstationALR, SubstationSCM, Border</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Launch of the scenarios “Island mode of the traín ALR0703” and “Island mode of the traín SCM703”</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Continuous monitoring of the main power network until the energy supply is re-established and this event is intercepted by the MicrogridController.</td>
<td>Main Power Network</td>
<td>Smart Grid</td>
</tr>
<tr>
<td>7</td>
<td>The MicrogridController changes its mode from island mode to connected mode by following the ECA that specifies this action when necessary.</td>
<td>MicrogridController</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>The MicrogridController notifies the change to connected mode to the connectors SubstationALR and SubstationSCM through the Border connector with the purpose of forwarding the notification of the change to rest of elements of the Smart Grid.</td>
<td>SubstationALR, SubstationSCM, Border</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Launch of the scenarios “Connected mode of the traín ALR0703” and “Connected mode of the traín SCM703”</td>
<td>–</td>
<td></td>
</tr>
</tbody>
</table>
5. Conclusion and Further work

From the experience in the projects ENERGOS [7], NEMO & CODED [27] and IMPONET [14], this paper presents an AC-based reference architecture for providing the intelligence and autonomy to microgrids that compose Smart Grids. This architecture is also based on standards guaranteeing the interoperability of the different devices within the power grid. The paper presents both: (i) the infrastructure deployment by defining components, connector and the GridController component; (ii) and the SPEEDs nodes to implement the MAPE-K control loop [41] that supports AC. This is then discussed contextually within a real scenario that illustrates in detail the deployment of this reference architecture. The scenario in which the intermittent power generation of renewable energies and distributed energies are integrated demonstrates that the reference architecture and the required SPEED nodes are able to support the self-balancing of demand/supply energy in the microgrid. Specifically, the architectural elements implemented by SPEED nodes must be orchestrated to maximize the time supplying energy by balancing the factors of energy produced, energy storage, and kind of contracts.

We have implemented this specific scenario with its specific properties and following the steps Table 2 and Table 3, however we are working on the creation of a set of patterns that could be reused by the community for implementing the policies of self-balancing the energy of the Smart Grid. These language-independent patterns establish both the algorithm and the data that must be monitored for their proper execution. Additionally, we plan to study how the variable parameters of these patterns, such as renewable resources, storage capability or economics, can affect their behavior. In addition, we are working on other self-* capabilities such as self-configuration, self-healing by evaluating the replication degree of SPEED nodes. Finally, new scenarios and tests should be executed to compare the obtained results with the results in other projects and real settings such as: the LINTER (LINTER – Union Fenosa Distribucion Smart Grids Laboratory) in which we are going to analyze and test the storage capacity, and the SPEED implementation as the software controller of power invertors, which control the power flow of cells.

6. Acknowledgements

This work has been partially supported by the Spanish Ministry of Industry (MITYC) through the R&D project IMPONET (ITEA 2 09030 - TSI-02400-2010-103), and the Centre for Industrial Technological Development (CDTI) through R&D projects NEMO&CODED (ITEA2 08022 IDI-20110864) and ENERGOS (CEN-20091048).

7. References


1 http://www.unionfenosadistribucion.com/en/home/smart+grids/1297137260045/find+out+about+our+laboratory.html


[26] de Mues M.O et al, Towards a Distributed Intelligent ICT Architecture for the Smart Grid, 9th IEEE Int. Conf. on Industrial Informatics (INDIN), 2011.

[27] NEMO & CODED: NEtworked MOnitoring and Diagnostics for Electrical Distribution Networks, ITEA 2m, http://innovationenergy.org/imponet/


[29] OSGiAlliance, Osgi service platform core specification. The OSGi Alliance Release 5, June 2012
