Scuola universitaria professionale della Svizzera italiana
Dipartimento tecnologie innovative

SUPSI **ISIN TECHNICAL REPORT**

Ad Hoc Nanogrid Testbed

with Support for On Demand Provisioning

Author(s)

Amos Brocco[1]

Technical Report Number

2014-1

Date

September 29, 2014

[1] amos.brocco@supsi.ch

Abstract

Smart grids aim at improving the existing power grid with active smart devices and decentralized energy resources. This concept also introduces smaller scale power generation and distribution infrastructures called microgrids and nanogrids. Accordingly, this paper details a testbed platform for an ad hoc self-organized nanogrid. The proposed nanogrid aims at creating an adaptive, scalable, and reliable power network that employs distributed resources to support energy provisioning with limited infrastructural planning. To support fully distributed power provisioning we employ smart routing devices which execute a decentralized routing protocol. Deployment scenarios include electrification and energy sharing in isolated rural areas and in post-disaster situations. We detail the provisioning protocol and we evaluate its adaptiveness, resiliency and scalability in a simulated system.

Introduction

Nowadays, power networks are rapidly shifting away from traditional designs, where energy flows in only one direction from power plants to consumers, toward intelligent and flexible architectures, known as *smart grids*, that support bidirectional power flows and are capable of fine-grain control of generation, transmission and consumption of electrical power. Smart grids take both communication and information technologies in the realm of power networks [1], easing the interaction between producers and consumers, and contributing to a more robust and efficient electric power delivery system. The *smartness* of the grid is frequently associated with smart metering devices and solutions for monitoring and controlling home appliances, reducing energy consumption and allowing for a more efficient usage of energy [2]. However, intelligent control is also required to integrate novel concepts in the existing infrastructure, such as Distributed Energy Resources (DER), which aim at bringing about significant improvements to the reliability of power networks [3]. Distributed generation typically relies on renewable sources such as solar panels or wind turbines, which are inherently volatile and characterized by intermittent production. As a consequence, the introduction of DER in the grid is susceptible to cause fluctuations in power flows as a result of variations in demand and offer [4]. To overcome this problem, distributed generation largely depends on the implementation of active control mechanisms [5], which help coping with unpredictable events and failures [6]. Active management of power networks is also related to the development of novel concepts that promote distributed generation, namely microgrids [7] and nanogrids [8]. Microgrids can operate either connected to the main grid, or independently from it (*islanded mode*), and focus on storage and consumption of energy near generation sites [3, 9, 10]. Nanogrids are standalone systems that rely exclusively on distributed sources and are thus more suitable for remote and off-the-grid operation. These new paradigms open up a wide new range of challenges, as well as possibilities and usage scenarios. On this subject, in this paper we focus on the problem of energy provisioning in rural areas and in post-disaster situations, where we envision the deployment of autonomous nanogrids that fulfill power requirements with little or no supervision. More specifically, we extend the concept of a nanogrid with ad hoc characteristics, presented in [11], by detailing and evaluating a suitable fully distributed energy routing algorithm and provisioning protocol. In the following we discuss the operation of such a system, by presenting its main hardware and software components. The rest of this paper is organized as follows: in Section 2 we describe the ad hoc nanogrid model and the considered application scenarios; in Section 3 we detail the design and basic operation of the smart power routing node. In Section 4 we present provisioning protocol and the routing algorithm, whereas in Section 5 we detail the considered evaluation scenarios and the corresponding results. Finally, Section 6 presents some related research work in the areas of autonomous control and power routing, and Section 7 provides conclusions and planned future works.

Ad hoc nanogrid

Ad hoc networks [2] are defined as local area wireless networks where hosts can communicate with each other without a fixed infrastructure. In computer engineering ad hoc networks are employed in situations where only temporary connections between peers are required and no planning is possible. Similarly, we intend an ad hoc nanogrid as a small mesh of interconnected devices and distributed energy resources that coordinate to provide *energy on demand* with little forethought and no underlying infrastructure. Other concerns of ad hoc nanogrids include topology control (i.e discover which nodes are spatially close or linked together), autonomous operation and configuration, and self-healing.
The purpose of an ad hoc nanogrid is to support energy provisioning with limited infrastructure planning in challenging operating conditions. In particular we consider two situations where infrastructure, namely the main grid, is not available or can be no longer used: electrification of rural areas and support to emergency response crews in disaster relief operations [11]. Both scenarios envision nanogrids that need to be rapidly deployed, might evolve and scale over time, and could be easily withheld or redeployed as needed. In the first scenario we aim at creating a nanogrid system that enables energy sharing in isolated or poorly developed areas, where power is typically provided by roof mounted photovoltaic modules. In this regard, proposed solutions must be scalable and simple to manage even by non-professionals. With the second scenario we target the creation of a robust emergency power network based on gasoline generators, solar cells and batteries. Such a system must be easily deployed and maintained, as well as quickly reconfigured to suit circumstances and requirements of rescue teams. In both the described situations the actual demand for power might exceed supply, requiring careful allocation of available resources depending on the needs and importance of each consumer. Consequently, an approach based on *on demand provisioning* is preferable, as it supports dynamic creation of transmission paths with different QoS levels. Furthermore, to ensure robustness and reliability, centralized control mechanisms should be avoided in favor of a fully distributed solutions. Finally, to ease the maintenance of the system, autonomous and self-organized behaviors need to be implemented to reduce the requirements for human intervention to a minimum. Conforming to these requirements, in the following we describe an autonomous on demand power routing algorithm based on fully distributed interaction between power switching components in the nanogrid.

Issues with distributed on-demand provisioning

Power routing in electrical networks might resemble data routing in computer networks, however significant differences hinder the implementation of the same solutions for both problems. In packet switched networks the routing protocol determines exactly where each piece of information should be transmitted to, and communication between computers is performed in a deterministic point-to-point manner. In contrast, power routing is bound to Kirchhoff's circuit laws, and actual power flows might differ from the routes intended by the logic of the algorithm. This becomes a problem in the event of a failure, because it will be more difficult to determine affected devices and apply the correct response. Moreover, with on-demand provisioning routing paths are allocated dynamically, and it is thus necessary to consider a number of side-effects resulting from each routing decision, such as cross connections. Solving these issues is made harder by the distributed nature of the system, because each decision must be taken without a complete knowledge of the grid.
An example of such issues is depicted in Figure 1, with small nanogrid consisting of four smart

[2] From the Latin meaning "*for this purpose*".

nodes A, B, C,D, and two generators G_1 and G_2 which can provide at most $15A$ $(amps)$. At time $t = 0$ a load L_1 requiring $2.5A$ is connected to port 1 on node C[3]. To fulfill the requirements of L_1 we suppose that a suitable distributed algorithm activates a provisioning path between port 2 on node C and port 4 on node A, passing through node B. At time $t = 1$, load L_2 (which requires $15A$) is connected to port 4 on node B. The provisioning mechanism employs generator G_2, because G_1 cannot provide more than $12.5A$. Accordingly, $15A$ are reserved on G_2 and the path from port 4 on B to port 3 on D is enabled. Unfortunately this decision joins two provisioning paths and $6A$ (instead of $15A$) and $11.5A$ (instead of $2.5A$) are drawn from G_2 and G_1 respectively [4]. In this situation L_1 and L_2 now rely on both generators, but only have an *agreement* with either one of them. If either L_1 or L_2 needs to be put offline, the management algorithm would have problem identifying all dependent loads.

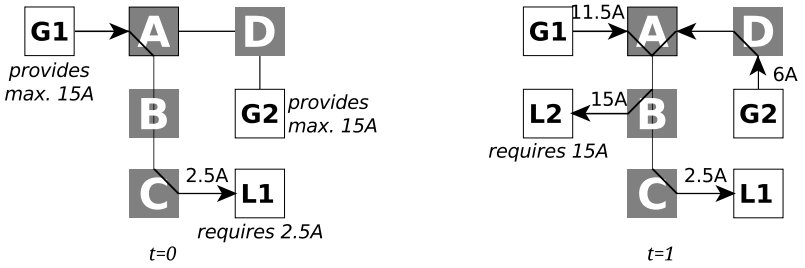

Figure 1: Example issue with distributed on-demand provisioning.

The distributed provisioning protocol detailed in Section 4 is designed to overcome these issues by continuously monitoring the network, registering observations, and reacting accordingly.

Smart power routing node

At the core of our ad hoc nanogrid concept lies a power switching device called *smart power routing node*, or *smart node*. A smart node (Figure 2) acts as an intelligent power router consisting of an embedded computer and several input/output ports where power sources, loads, or other smart nodes can be connected to. The actual design integrates four ports [5], and at most one external device can be connected to each port. Power routing is achieved by means of cross-point switches that enable internal connections between any pair of ports on each node. Smart nodes run a manage-

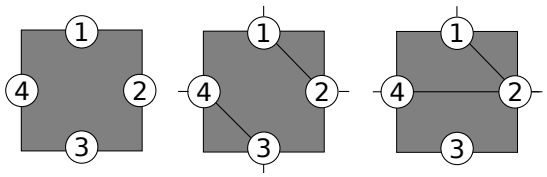

Figure 2: Smart node with 4 ports and example routing configurations.

ment software that is responsible for creating and maintaining power routing paths between loads

[3]For the sake of clarity, ports on each node are numbered clockwise starting from the one on the top edge.

[4]Values shown in these examples have been obtained using the Gnucap circuit simulator, and serve the purpose of illustrating situations where an incongruence between the power flow decided by the routing algorithm and the real one exists.

[5]This number has been arbitrarily chosen to limit the amount of electronic power switching components (relays) in the hardware prototype.

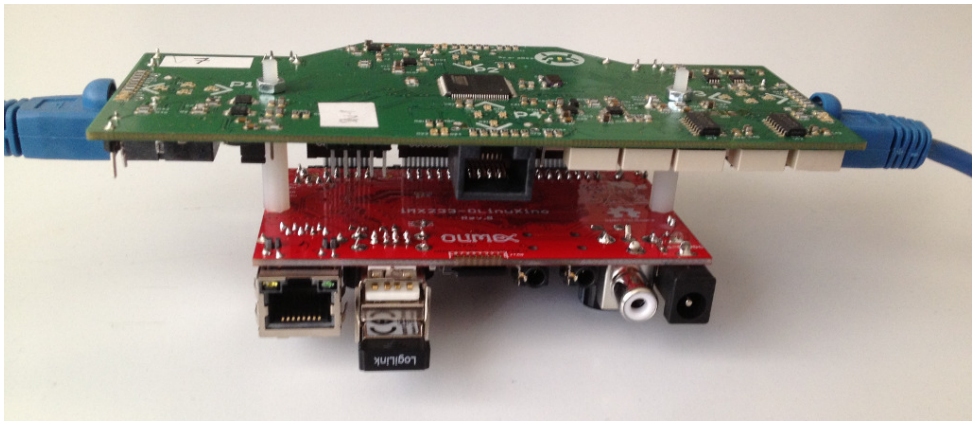

Figure 3: Smart node prototype: master (bottom) and slave (top) boards.

and generators [6]. More specifically, the control algorithm, using a best effort approach, ensures that all loads receive the required amount of power, that generators do not become overloaded, and that optimal transmission paths are chosen. To monitor electric current flows, smart nodes are equipped with voltage and current sensing circuitry for each port: this data is sent to the local management algorithm which then determines proper response actions.

Hardware prototype

At present the project employs a low-voltage (10V DC, 2A maximum) hardware prototype to validate the operation of the nanogrid in a realistic environment, thus complementing software simulations. Smart nodes are comprised of two main components, namely a master board and a slave board (Figure 3). The master board runs the high-level management software that takes care of energy provisioning in the nanogrid and interaction between smart nodes, whereas the slave board implements the electronics needed for power measurements and switching. Communication between the two boards is achieved using both I^2C and a serial UART link.
The master board is built around a Freescale iMX233 embedded ARM processor, sided with 64MB of RAM, running the control software (written in Python) on top of a GNU/Linux distribution. The slave board hosts 4 ports where external devices such as loads, generators or other nodes can be connected to. Switching hardware is controlled by an Atmel AVR32 microcontroller, which communicates with the master board and provides current and voltage measurements for each port. Beside a power connector, each port also has a serial interface allowing for point-to-point communication between nodes: data transmission is multiplexed on the single serial connection between the master and the slave boards. This serial link is used to advertise the node's local IPv6 address to neighbouring nodes in the network. Because control of the power network is fully distributed, communication and coordination between nodes plays an important role: to overcome the bandwidth limitations of the current UART link, the provisioning protocol is executed over an IPv6 wireless ad-hoc network. The node is currently powered by an external wall adapter, however this can be easily replaced by a battery in order not to depend on any external power source.

[6] For simplicity, in the following sections, loads and generators will be sometimes referred to as devices that can directly communicate with each other (by sending and receiving messages), although precisely speaking it is currently the connected smart node that performs this task.

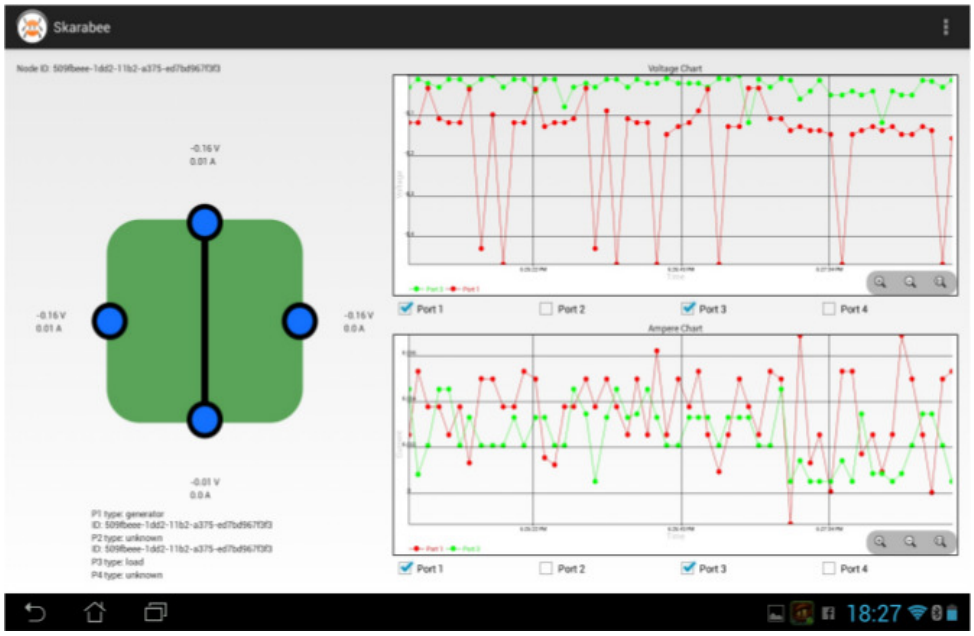

Figure 4: Management interface for Android.

Management interface

Smart nodes can operate mostly autonomously without user intervention. However, in the actual design each node can only detect the identity of neighbour nodes connected its ports (through the serial interface), but not the details of other devices (loads and generators). The capabilities of connected energy sources and the requirements of connected loads must be therefore defined by a human operator. Accordingly, a remote configuration tool for Android devices has been developed (Figure 4). An user scans a QR code printed on the smart node with his smartphone in order to obtain the details for a Bluetooth connection (MAC address). Once devices are paired, a graphical application can be used to monitor the status of the smart node (voltage, current and routing configuration) and to define the properties of connected devices. Future hardware revisions will consider the integration of *intelligent* devices (*smart appliances*) whose capabilities can be directly queried by nodes, reducing the need for human intervention.

On demand provisioning protocol

The control of a power grid relies on a careful balance between demand and offer, and should pay attention to the limits of each component. On the one hand, an ad hoc nanogrid should ensure that all connected loads receive the required amount of power. On the other hand, energy sources and transmission lines must operate within their limits and not be overloaded. In contrast to traditional power grids, control mechanisms in an ad hoc nanogrid must operate in a dynamic environment without a global overview, and consider sudden changes in the network topology originating from a rearrangement of transmission lines (either voluntary or accidental) or from churn (devices being added or removed). At a higher level too the system must be prepared to deal with dynamically changing constraints (for example, changes in user-defined load requirements or priorities). Hence, the routing algorithm needs to continuously monitor the system and must be able to react promptly and adjust its decisions. To achieve robust, reliable and optimal provisioning we propose an on

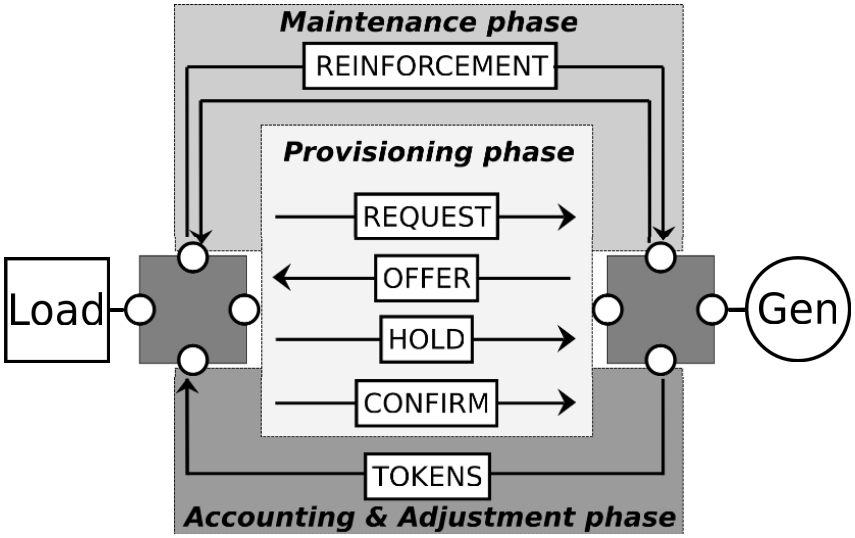

Figure 5: Provisioning protocol: schematic overview.

demand provisioning protocol and a fully distributed power routing algorithm. The protocol requires coordination between smart nodes, and is divided into several phases, which are illustrated in Figure 5. Each phase employs different messages that are exchanged by nodes on the communication network. In the following the operations of each node in each phase, as well as the information exchanged by nodes, will be detailed.

Provisioning phase

In the provisioning phase, each node verifies that connected loads receive the required amount of power (as defined by a human operator). If requirements are not fulfilled or only partially fulfilled, a discovery process is started in order to find energy routing paths toward power sources in the network. Allocation of new routing paths is performed using a *check-then-act* sequence: first, discovered paths are evaluated and the optimal solution is determined (for example, the one crossing the least number of smart nodes); subsequently, routing nodes are notified and required to store the details of the path. Finally, the routing decisions are confirmed and the routing path is activated. The node connected to the load initiates and coordinates the whole process, from discovery to confirmation, by means of different types of messages.

Request

Request messages are queries that contain information about power requirements, and are marked with a unique identifier for tracking purposes. The initiating node starts by inquiring generators connected to its ports (if any). If the requirements cannot be fulfilled locally, the request message is forwarded for a limited number of hops in the network following the topology determined by transmission lines, in order to have a one-to-one match between the path travelled by request messages and the resulting power routing path. Point-to-point communication is achieved either on the wireless network or using the low level serial connection between nodes. To help determining the best route, additional information about discovered nodes is stored in the message, such as the resistance of transmission lines. Moreover, nodes might define forbidden paths, for example through transmission lines that are about to be voluntarily disconnected or are approaching their maximum

capacity. To limit the amount of network traffic, requests are forwarded using an iterative deepening strategy: messages are initially propagated at a small depth (hop count) in the network; the limit is then progressively increased should the request fail to find suitable routing paths. If all attempts fail, a *best effort* request is issued in order to find all existing power sources whose combined capabilities satisfy the requirements of the load.

Offer

Upon receiving a request message, a node verifies if local generators have any leftover capacity. If available power satisfies the requirements defined in the query, the node replies to the initiating node with an offer message. On the contrary, if no generators are connected to the node or if the capacity is insufficient, the request message is forwarded further in the network. Offer messages detail the type of power source as well as its actual and maximum capacity.

Hold

Following a request, the initiating node waits for some limited amount of time for incoming offer messages. All received offers are evaluated using a cost function, which can vary depending on user requirements: example costs include total impedance of the routing path, amount of leftover capacity on the power source, etc. The cheapest offer is selected and hold messages are issued to register the details of the path on each traversed node. These details, which define an *agreement* (or *contract*) between the power source and the load, include the unique identifier of the provisioning request, the amount of power requested, as well as the identifiers of all nodes and transmission lines along the path, etc. In contrast to request and offer messages, hold messages (as well as confirm messages) must be transmitted using a reliable protocol to ensure that all recipients are successfully informed. Should a node on the path be unreachable, the provisioning phase is restarted.

Confirm

When all nodes on a path have successfully received the hold message, a confirm message from the initiating node can follow. Upon receiving a confirm message, the node retrieves the stored details of the path and activates the appropriate routing configuration between local ports. When all nodes on the path are properly configured electricity will flow from the generator toward the load. Subsequently, the initiating node switches to the maintenance phase for this path, and becomes responsible for monitoring and verifying that routing configurations for this path remain active. Because resources in the nano grid are shared, each node might end up being involved in several provisioning contracts, each one serving a different load-generator pair.

Maintenance phase

Each confirmed path has to be maintained in order to detect and solve all unexpected failures and ensure robust operation of the nanogrid. Hence, nodes must monitor confirmed paths for locally connected loads: if a problem is detected or the agreement with the power source is revoked, the path is cancelled by requesting each node on the path to deactivate the corresponding routing configuration. However, if the node responsible for a path experiences a failure, routing configurations on other nodes might incorrectly remain active. To automatically remove inactive or broken paths the maintenance phase employs a bio-inspired signaling protocol that mimics the behavior of ants. In particular, each path stored on a node is associated with a soft-state numerical value called *artificial*

pheromone[7]. The concentration of the pheromone is initialized at 1 when the path is confirmed, and is periodically decreased as to simulate evaporation. When the pheromone concentration associated with a path falls below a certain threshold, the path is discarded by the node and the corresponding local routing configuration is disabled (if not used by any other path). To keep a confirmed path alive, the node on the demand-side (i.e. connected to the load) periodically forwards a reinforcement message along the path. On each node encountered while travelling forward toward the generator, transmission lines are verified to ensure that power can correctly flow across the path. When the reinforce message reaches the last node (i.e. the one connected to the power source) it changes its direction, heading back to the starting node. At each step on the back route, the pheromone value associated with the path is incremented to its maximum value. If for some reason the reinforce message cannot complete its journey while travelling forward (for example because the path has been deleted by a node), a negative reinforcement is employed on the way back to speed up the clean-up process.

Accounting and adjustment phase

Because only a limited number of transmission lines exist in the nanogrid, multiple path might easily cross, making it difficult to determine dependencies between loads and generators and enforce provisioning agreements. Furthermore, power flows might not follow the intended routing, as they are bound to physical laws. These situations hinder an accurate control over all resources, and might prevent appropriate response in the event of failure. Even under normal conditions, if a power source is to be put offline for maintenance, it is necessary to know and notify all connected loads to avoid any disruption. Accordingly, our protocol incorporates a mechanism for determining how power is propagated from generators toward loads. To achieve this goal, each node distinguishes between positive ingoing power flows and negative outgoing flows: power sources contribute to positive flows, whereas loads sum up as negative flows. To monitor energy consumption in the network, each node tracks current flows by means of network messages called *tokens*, which store the identifier of a power source and the amount of provided power (the *weight* of the token). To account for the amount of ingoing current the concept of *ingoing tokens* is used, whereas for outgoing currents *outgoing tokens* are employed. Tokens are generated and propagated along transmission lines as follows.

Token generation step

Nodes connected to power sources measure the amount of current flowing through the corresponding port (in *amperes*) and generate a corresponding *ingoing token*. Additional tokens might be received from neighbour nodes, if a positive amount of power is provided.

Token propagation step

For each set of connected ports, the node computes the whole amount of *ingoing tokens* that each port contributes. Subsequently, for each port in a set with a negative current flow, *outgoing tokens* are generated. The weight of these tokens is a proportional fraction of the weight of the corresponding *ingoing tokens*, and measures the amount of current exiting the node through that port. Afterwards, *outgoing tokens* are either propagated to neighbouring nodes (which consider them as *ingoing tokens*) or used determine source dependencies for locally connected loads.

[7] Pheromones are chemicals released by ants to mark paths between the nest and food sources.

Adjustment step

When source dependencies have been determined, the node verifies that all power received by a load device is accounted for (i.e. that the corresponding agreements with the power sources exist). An adjustment step is required when the energy received does not match the value agreed upon during the provisioning phase. If an existing agreement needs to be updated, an adjustment request is piggybacked to reinforcement messages: if the target node does not accept the adjustment the path might be be cancelled if the generator becomes overloaded. Following the same schema, it is possible to make adjustments for potentially more energy, however if the increase cannot be accepted the requesting node should look for alternatives. Furthermore, if power is received from a generator but no confirmed path exists, the path is created. Confirmed paths toward a generator can also be deactivated if the node detects a negligible contribution from a source.

Improvement and alternative discovery phase

Beside maintaining existing paths active, nodes are also responsible for issuing proactive requests to find alternative provisioning routes. Improvement queries allow for continuous optimization of the nanogrid by taking into account changes in the topology and availability of new power sources that might reduce the overall provisioning cost (for example, the number of active transmission lines). The switch-over to an alternative path is seamless, and performed only when the initial requirements are fulfilled. When a better path is discovered through the aforementioned proactive queries, the node can perform a switch-over by simultaneously confirming the new path and cancelling the superseded one (which would be subsequently cancelled by the maintenance process). Proactive queries might also be executed when a node is notified that an existing path is about to be interrupted, and an alternative solution has to be found.

Load priorities and QoS levels

Because ad hoc nanogrids are aimed at resource constrained situations, the provisioning algorithm must support multiple load priorities and pre-emption. The implemented solution considers load priorities in terms of different QoS levels: low priority loads should expect a lower QoS, whereas high priority loads would receive the best QoS. Priority-based provisioning is thus concerned with both the choice of suitable energy sources and of transmission paths. In this regard, operators might also choose to restrict some generators to serve only certain load priority classes, for example to guarantee that high priority loads get the least volatile sources. During both the request and the maintenance phase, nodes ensure that high priority paths do not intersect with low priority ones, otherwise voltage fluctuations caused by intermittent sources on one side would affect the other. QoS levels are particularly useful when demand exceeds offer, as the system would be able to pre-empt low priority loads. To minimize disruption, loads are given the opportunity to discover alternative power sources in the nanogrid before being disconnected.
We assume that each provisioning query, as well as hold messages, carry additional information about the priority level of the contract. When a contract is confirmed, each node on the path resolves priority conflicts by disabling all crossing paths which have a lower priority. To limit penalization of low priority contracts, the amount of paths that might be pre-empted is accounted for as an increased cost in the provisioning phase. More specifically, if multiple provisioning paths exists, the system is more likely to choose the one which results in the lowest amount of priority conflicts.

Evaluation

The operation of the provisioning algorithm was tested both on the hardware prototype platform and in a simulated environment. As both testbeds share the same codebase, the evaluation phase enabled us to thoroughly validate our approach, by verifying that the protocol is both suitable for deployment as well as scalable. Because simulation runs are easily reproducible and verifiable, in this section we report the results obtained in synthetic scenarios.

Large scale evaluation scenario

A large-scale empirical validation of the provisioning protocol was performed solely in a simulated DC nanogrid using the Gnucap [8] circuit simulator. To this extent, a dynamic scenario was considered, taking into account different aspects such as the adaptiveness and reliability of the proposed approach, as well as its scalability. The considered initial setup, which is depicted in Figure 6, consists of 7 smart nodes, 3 loads and a 2 power sources (generators). For simplicity, all considered loads require $2.5A$ and each generator can feed at most $15A$. To evaluate the scalability and adaptiveness of our solution the nanogrid is expanded during the simulation: after 250 seconds the number of nodes is increased to 29, and 11 additional loads as well as 3 generators are connected to the system; after 500 seconds, the network size is further increased, reaching 77 nodes, 20 loads, and 10 generators. To validate the robustness of the system, starting from 750 seconds into simulation, several nodes and devices are disconnected. Two disconnection policies are considered: the first one assumes *graceful* disconnection of nodes, and the second one uses *abrupt* disconnection. Graceful disconnection enables the system to resolve alternative paths before nodes are removed, whereas abrupt disconnection simulates a failure. Each simulation run lasts for 1500 seconds, and the presented results are an average over 10 repetitions. Communication between smartnodes is assumed to be reliable, and due to the asynchronous nature of the simulation the average over 10 runs is presented (although no significant difference in the behaviour of the nanogrid was revealed across different runs).

Resilience and adaptiveness

The resilience and adaptiveness of our protocol is determined by the ability to provide power to each load through optimal routing paths. In this regard, Figures 7 and 8 illustrate the percentage of the power received by loads (100% meaning that the required power is provided) as well as the fraction of active transmission lines (which should be minimized). When the network is expanded, new loads must wait until provisioning paths are created. The speed of this process depends on the time required for discovering and activating a new path, which in turn depends on the complexity of the topology and the distance between the load and potential power sources that affects the number of trials required (as iterative deepening is used to propagate request messages). For this evaluation, nodes typically need 15 to 45 seconds to perform the request, hold, and confirm phases. When nodes are disconnected, loads are either minimally affected (graceful disconnection) or significantly affected (abrupt disconnection). However, in both situations, alternative transmission paths are discovered (in a real network a battery backup would be sufficient to eliminate most disruptions). As indicated by the decrease in the number of active transmission lines after 350 and 600 seconds, provisioning paths are not only reconfigured dynamically to adapt to topology changes (for example, following the disconnection of a node) but also to exploit newly discovered shorter paths and power sources that become available when the network is expanded.

[8] www.gnucap.org

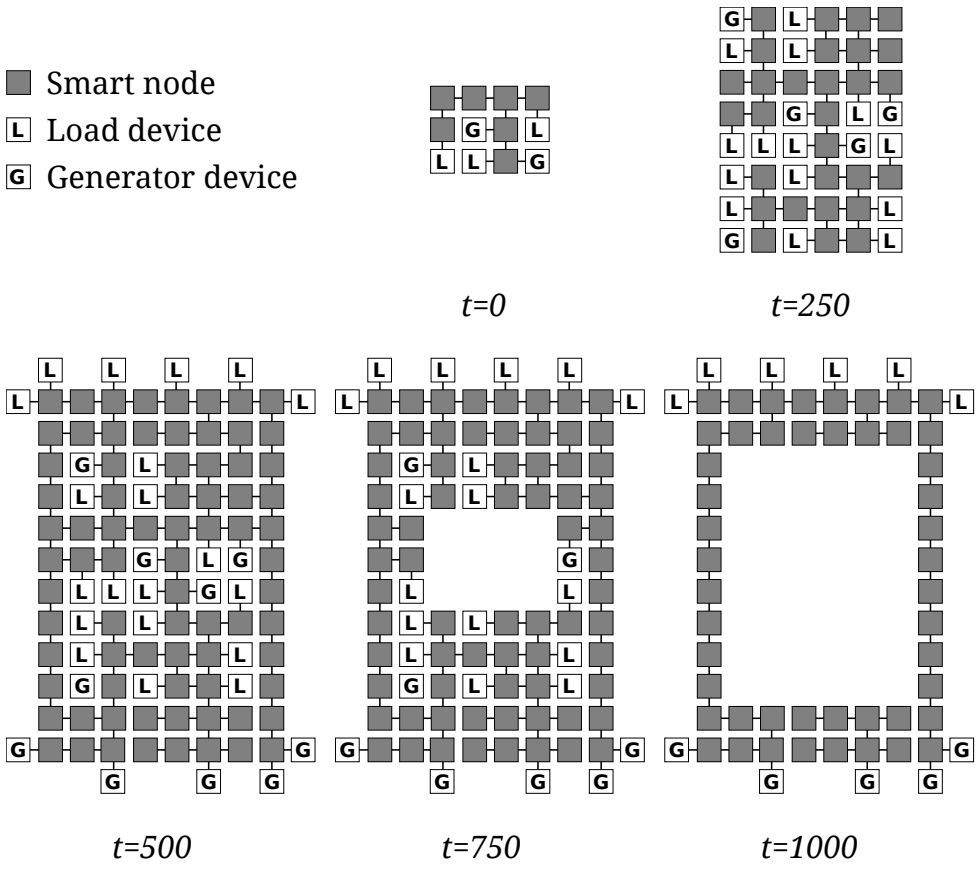

Figure 6: Evaluation scenario with multiple loads and generators: nodes are added and removed throughout the simulation.

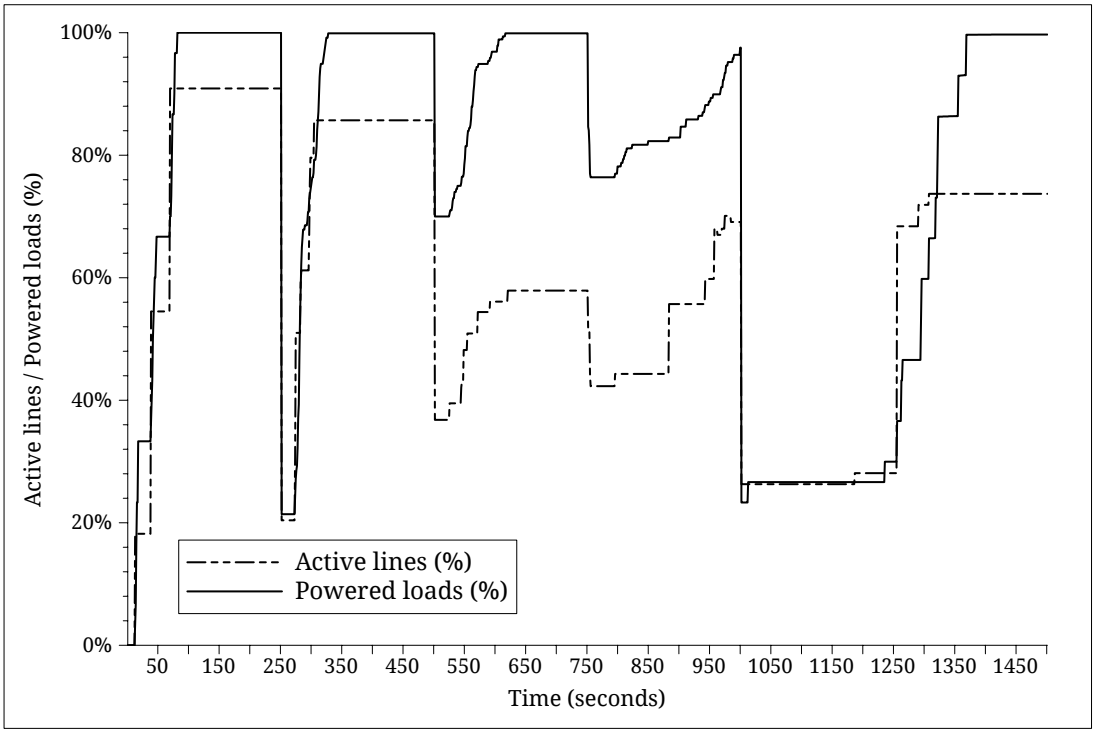

Figure 7: Powered loads and active transmission lines (abrupt disconnection)

Network overhead and scalability

The scalability of the provisioning protocol is determined by the amount of network traffic generated by the algorithm (messages sent), and is illustrated in Figures 9 and 10. The simulator limits the traffic sent through each port to 28kbps, in order to replicate the conditions of the hardware testbed. Up until 1000 seconds into simulation, the average traffic per node (grey line) is about 1kbps, whereas the traffic per load (black line) varies from 2kbps to about 4kbps. Afterwards, the traffic increases significantly, affected by the negative behavior of the improvement phase: requests need to travel many steps in the network to reach power sources and thus contribute a great deal to network overhead. As expected, the traffic generated by the protocol depends on the complexity of the network and the number of transmission lines, however results demonstrate that the protocol is able to scale well and that the observed traffic is sustainable by an ad hoc wireless network [12]. Traffic peaks are evident when the network is expanded or shrinked, as provisioning requests are executed, but there is no significant difference between abrupt and graceful disconnection, because both require the affected nodes to find alternative paths. Reductions of the traffic could be achieved by employing wireless broadcasting (whenever possible) or, at the expense of a less optimized or robust network, by lowering the frequency of the improvement, maintenance and adjustment phases.

Related work

In this section we briefly present some of the research literature related to the two main topics covered by our research: autonomous control of micro- or nanogrids and intelligent power routing. Autonomous control can be achieved using a partially decentralized model, as discussed in [13], where an intelligent and self-configurable microgrid is presented. However, as suggested in [14], the

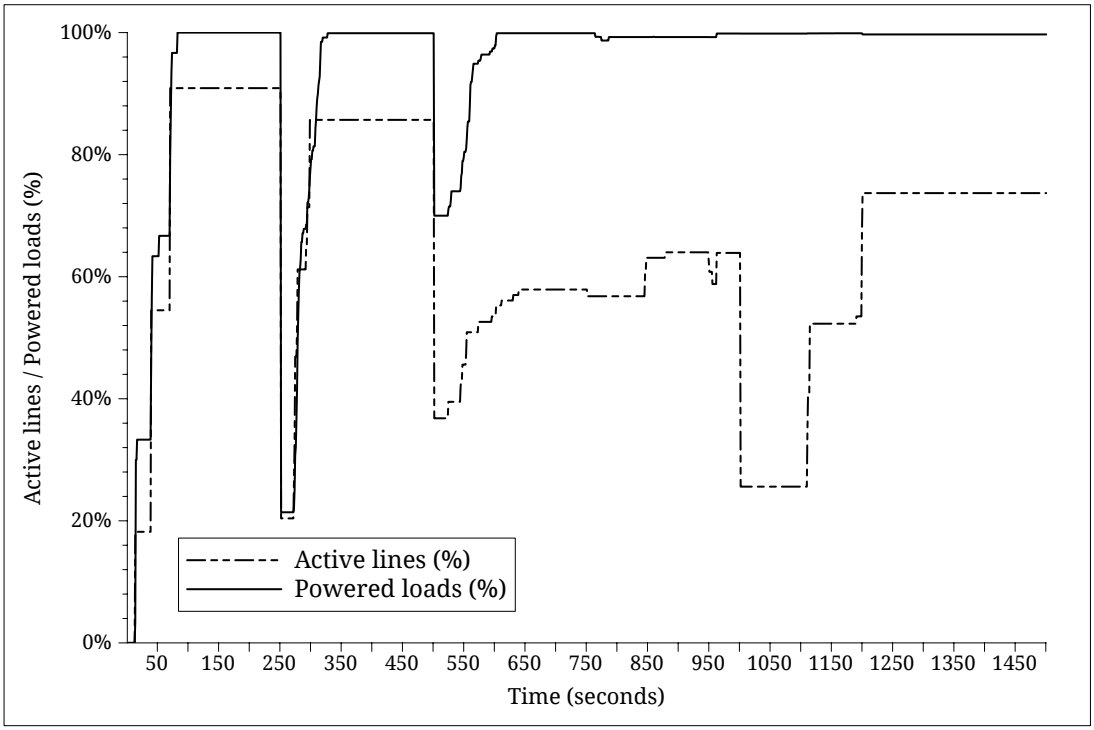

Figure 8: Powered loads and active transmission lines (graceful disconnection)

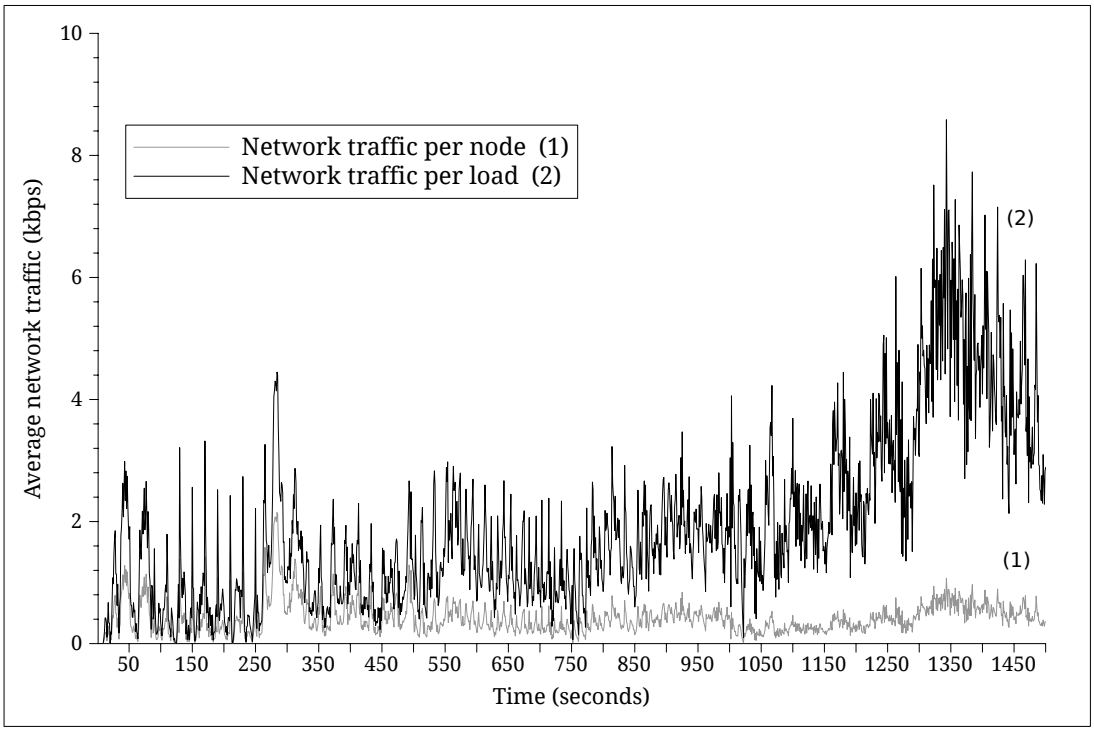

Figure 9: Average network traffic (abrupt disconnection)

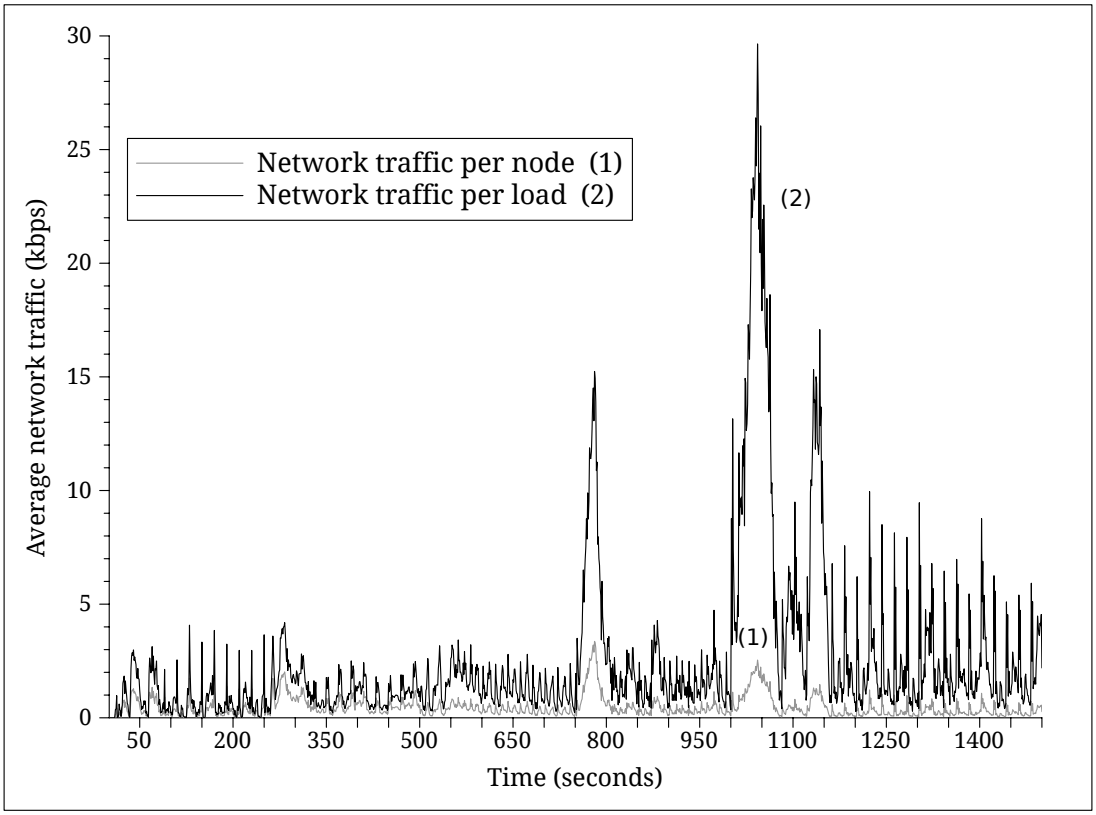

Figure 10: Average network traffic (graceful disconnection)

increasing complexity of energy networks necessitates fully decentralized solutions capable of self-management and adaptive behaviors. Similarly, in [15] decentralized control is deemed essential for microgrids, because it increases robustness and survival chances in the event of failures; to achieve such decentralized control, solutions based on peer-to-peer interaction between components can be employed. In this regard, a simple decentralized control mechanism for nanogrids is presented in [16]: unfortunately it assumes that system parameters are known by each source and is thus inapplicable in our considered scenarios. Decentralized control is frequently related to the idea of providing *energy on demand* and demand side management systems, for example to actively control household appliances [17, 18]. However, the differences in the operating conditions and the objectives of the control strategy make such solutions unsuitable for an ad hoc nanogrid. Concerning power routing, a lot of research work has been done to implement smart power routing mechanisms. For example, [19] advocates the need for distributed and flexible management system, and presents a distributed power routing algorithm to resolve optimal paths in active distribution networks. The research presented in [20] details a secure routing algorithm aimed at energy sharing in smart microgrids, whereas intelligent power routers and distributed coordination have been proposed in [21] to increase the robustness and flexibility of the distribution network. Similarly, in [22] an agent-oriented system for a self-healing smart grid is discussed: energy is dynamically re-routed toward optimal paths depending on the state of the system, in order to overcome disruptions that can lead to unserved demands for power.

Conclusions

In this paper we presented the concept for an ad hoc nanogrid, along with an autonomous on demand provisioning protocol based on a fully distributed power routing algorithm. The system implements active decentralized control by means of intelligent power routing nodes that monitor current flows and collaborate with each other. To avoid the need for an underlying communication infrastructure, information is exchanged across an ad hoc wireless network. The goal of the proposed system is to create an adaptive, scalable, and reliable nanogrid that does not require supervision or control from a central component. The considered application scenarios aim at situations where the main grid is unavailable or severely damaged: electrification of rural areas and support for disaster relief operations. Both rely on distributed energy sources such as photovoltaic or gasoline generators, and present highly dynamic conditions: the topology of the network as well as load requirements might change without notice, and unexpected failures could affect power transmission. Empirical results obtained in a simulator validate the operation of the provisioning protocol as well as its scalability. Current research focuses on parameter sensitivity and security aspects such as authentication and trust management, along with support for energy storage devices. Finally, the proposed provisioning protocol could also be applied to other problems, such as water supply networks.

References

[1] H. Farhangi. The path of the smart grid. *Power and Energy Magazine, IEEE*, 8(1):18 –28, january-february 2010.

[2] P. McDaniel and S. McLaughlin. Security and privacy challenges in the smart grid. *Security Privacy, IEEE*, 7(3):75–77, 2009.

[3] R H Lasseter. Microgrids and distributed generation. *Journal of Energy Engineering*, 133(3):144, 2007.

[4] G. Joos, B.T. Ooi, D. McGillis, F.D. Galiana, and R. Marceau. The potential of distributed generation to provide ancillary services. In *Power Engineering Society Summer Meeting, 2000. IEEE*, volume 3, pages 1762 –1767 vol. 3, 2000.

[5] Faycal Bouhafs and Michael Mackay. Active control and power flow routing in the smart grid. *IEEE Smart Grid*, December 2012.

[6] S. Massoud Amin. Smart grid: Overview, issues and opportunities. advances and challenges in sensing, modeling, simulation, optimization and control. *Eur. J. Control*, 17(5-6):547–567, 2011.

[7] S. Mizani and A. Yazdani. Design and operation of a remote microgrid. In *Industrial Electronics, 2009. IECON '09. 35th Annual Conference of IEEE*, pages 4299 –4304, nov. 2009.

[8] Chris Marnay, Bruce Nordman, and Judy Lai. Future roles of milli-, micro-, and nano- grids. In *CIGRÉ International Symposium The electric power system of the future - Integrating supergrids and microgrids*, Bologna, Italy, 09/2011 2011. LBNL, LBNL.

[9] G. Pepermans, J. Driesen, D. Haeseldonckx, R. Belmans, and W. D'haeseleer. Distributed generation: definition, benefits and issues. *Energy Policy*, 33:787–798, 2005.

[10] T. Ackermann, G. Andersson, and L. Söder. Distributed generation: a definition. *Electric Power Systems Research*, 57:195–204, 2001.

[11] Amos Brocco. Ad-hoc self-organized microgrid for rural electrification and post-disaster response. In *Proceedings of the IEEE Green Technologies Conference 2013*, 2013.

[12] D. Murray, M. Dixon, and T. Koziniec. An experimental comparison of routing protocols in multi hop ad hoc networks. In *Telecommunication Networks and Applications Conference (ATNAC), 2010 Australasian*, pages 159–164, 2010.

[13] A.A. Zaidi and F. Kupzog. Microgrid automation - a self-configuring approach. In *Multitopic Conference, 2008. INMIC 2008. IEEE International*, pages 565 –570, dec. 2008.

[14] Matthias Baumgarten and Maurice D. Mulvenna. Towards intelligent and self-evolving network infrastructures for energy management. In *SASO Workshops*, pages 72–75, 2010.

[15] P. Piagi and R.H. Lasseter. Autonomous control of microgrids. In *Power Engineering Society General Meeting, 2006. IEEE*, page 8, 0-0 2006.

[16] Bryan J., Duke R., and Round S. Decentralised control of a nanogrid. In *Australasian Universities Power Engineering Conference*, Christchurch , New Zealand, 2003.

[17] Y. Suhara, T. Nakabe, G. Mine, and H. Nishi. Distributed demand side management system for home energy management. In *IECON 2010 - 36th Annual Conference on IEEE Industrial Electronics Society*, pages 2430–2435, 2010.

[18] T. Kato, K. Yuasa, and T. Matsuyama. Energy on demand: Efficient and versatile energy control system for home energy management. In *Smart Grid Communications (SmartGridComm), 2011 IEEE International Conference on*, pages 392–397, 2011.

[19] Phuong Nguyen, Wil Kling, Giorgos Georgiadis, Marina Papatriantafilou, Tuan Le, and Lina Bertling. Distributed routing algorithms to manage power flow in agent-based active distribution network. *IEEE PES Conference on Innovative Smart Grid Technologies Europe, Gothenburg, Sweden, October 10-13, 2010*, 2010.

[20] Ting Zhu, Sheng Xiao, Yi Ping, D. Towsley, and W. Gong. A secure energy routing mechanism for sharing renewable energy in smart microgrid. In *Smart Grid Communications (SmartGridComm), 2011 IEEE International Conference on*, pages 143–148, 2011.

[21] A. A. Irizarry-Rivera, M. Rodriguez-Martinez, B. Velez, B. Velez-Reyes, A. Ramirez-Orquin, E. O'Neill-Carrillo, and J. Cedeno. Intelligent power routers: a distributed coordination approach for electric energy processing networks. *International Journal of Critical Infrastructures*, 3(1-2):20–57, 2007.

[22] S. Bou Ghosh, P. Ranganathan, S. Salem, Jingpeng Tang, D. Loegering, and K.E. Nygard. Agent-oriented designs for a self healing smart grid. In *Smart Grid Communications (SmartGridComm), 2010 First IEEE International Conference on*, pages 461 –466, oct. 2010.