Distributed load shedding algorithm for islanded microgrid using fog computing paradigm

Diana Rwegasira[1,2], Imed Ben Dhaou[3,4], ,Syed Kakakhel[4], Tomi Westerlund[4] and Hannu Tenhunen[2,4]

[1]University of Dar es salaam, Tanzania
[2]Royal Instittute of Technology, Sweden
[3]Universty of Monastir, Tunisia
[4]University of Turku, Finland

***Abstract*— Demand Side Management, DSM, is a program supported by the smart-grid which aims at matching the energy consumption and production. Several techniques for demand-side management have been proposed, including load-shedding, time of use pricing, real-time pricing, and critical peak pricing. In this work, we propose a distributed load-shedding algorithm using the multi-agent system. The agents in residential areas collaborate to reduce the energy demands using various forecasting techniques. The computational distributed framework is provided via fog computing to minimize power consumption, costs, and latency when designed using LoRaWAN protocol.**

Keywords: Load Shedding, DC microgrid, fog computing, Lora WAN,

I. INTRODUCTION

The increasing demands on electricity coupled with the need to reduce the operating cost, protect the environment, and improves the grid reliability, have contributed to the need for developing smart grids. The salient features of the smart-grid include distributed energy resources, integration of renewable energy, demand-side management, and resilient operation.

The microgrid is the key enabler for distributed energy resources. It engenders renewable energy generators, energy storage systems, and controllable loads[1]. Microgrids can cover large areas. Three types of microgrids have been identified: residential, commercial and industrial [1]. Residential DC-microgrid is a particular case of a microgrid in which a large number of houses have a roof-mounted and interconnected PV panels. The general architecture of the residential DC residential microgrid is shown in Figure 1.

Renewable energy has emerged as a solution to provide electricity to remote locations that have no access to the primary grid, to reduce carbon dioxide, and to increase the efficiency of the utility grid. The use of renewable technologies such as solar power and wind power have proven to be a cheaper alternative at the household level. DC microgrid systems are focused on (i) increasing power production, (ii) reducing energy dissipation and (iii) decreasing facility cost. DC microgrid systems are also vital during a disaster as they can operate in islanded mode. DC microgrid possesses various features which makes it very attractive. Some features of the DC microgrid are: (i) there is no need for synchronization between different sections of the network (ii) has virtually zero energy loss that occur when converting DC voltage into AC, (iii) operates at low cost, (iv) resilient to skin effect and inductive/capacitive loss thus requires less regulation compared with AC grid systems and (v) enables the connection of DC devices that have high efficiencies compared with AC appliances [3] [4].

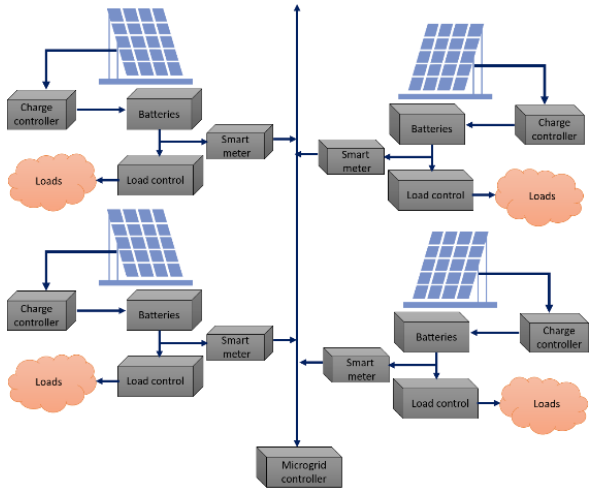

Figure 1: Architecture of a DC residential microgrid

The operation of the DC-microgrid is managed by a microgrid controller that schedules the operations of non-critical loads to operate in the off-peak times. In the islanded microgrid (disconnected from the main grid), the multi-agent system has been proposed as an effective technique for automatic load-shedding. However, the communication aspect and security have received scant attention. In this work, we propose a computational and communication framework provided by a fog computing platform to complement and address the shortcomings of the work described in [9].

The rest of the paper is organized as follows. In Section II, we discuss related work. Section III reports the distributed load shedding algorithm using JADE MAS. In Section IV, we described the fog computing architecture for the implementation of the distributed load-shedding algorithm. The experimental results are reported in Section V. Finally, the paper is concluded in Section VI.

II. RELATED WORK

Traditionally, load shedding algorithms have been based on centralized mechanisms where frequency and voltage levels are closely monitored for fluctuations. Previous studies [2], [3] [4] show several different approaches of shedding fixed amounts of loads whenever particular frequency thresholds are reached. Despite the simplicity of these approaches, they are suboptimal and not flexible for every power imbalance situation [5][6]. As a result, new load shedding schemes

based on adaptive and computational intelligent algorithms have been proposed. These schemes improve the power imbalance estimation process and provide more optimal solutions, even in complex power system structures.

Usman et al. [7] propose a load shedding algorithm based on hybrid optimization techniques. The algorithm uses an evolutionary particle swarm optimization technique to minimize power loss and cost of disconnecting loads. Mortaji et al. [8] propose a load shedding algorithm that relies on direct control of appliances in real-time. The algorithm classifies appliances into static and dynamic types depending on their operational characteristics. This granularity and direct control allow the algorithm to be more sensitive to changes. However, the approach relies on a stable and robust communication architecture.

III. DISTRIBUTED LOAD SHEDDING

Load shedding techniques aim at matching energy supplies and demands. The microgrid controller supervises the transactions of the power in accordance with an auction system. The system operates as follows. First, the microcontroller determines the available power and the demand in the next time interval. If the supply exceeds the demand, then the storage units are instructed to store the surplus charge. If the demand surpasses the supply then the load shedding algorithm executes. In the distributed load-shedding algorithm, the smart-meter determines its bid and sends it to the microgrid controller. The latter allocates the energy quota among the bidders. Our work brings an improvement to the system described in [9]. More precisely, the microgrid controller allocates the least energy per bidder as opposed to the winner takes all solution.

To achieve these activities, we utilize a multi-agent system (MAS) to solve the power allocation problem in the demand-response scheme. Multi-agent systems have become popular in solving distributed control problems [10]. MAS has several attributes, such as autonomous, active, social, and collaborative operations. These attributes implement the control and monitoring actions efficient and reliable.

A. Microgrid Controller Agent

This agent manages the overall activities and agent communication. It links with all agents and uses the controller method with input variables of solar power and storage battery capacity and output of get Power Output. The agent handles the actions of load shedding as described in [12] to the distributed microgrid. The agent will have a database of all houses' demands for the next 24 hours and the history of the power consumptions which will be used in the bid action. Using the history of the controller database, it will be able to assign priority based on critical loads, or the power consumption of a microgrid.

B. Smart Meter Agent

This agent is concerned with the control of the load agent. It keeps on relaying the price of power to the load and commands the load agent for the action to be taken. If the price is high, the smart meter agent will request the smart plug agent to react on the process either to switch off all devices or switch off only the non-important devices. It will also be responsible for charging and discharging activities and reserving the power. Its input variables are voltage, charge capacity, and solar power.

C. Smart Plug Agent

This agent keeps on monitoring the power consumption inside the house. The main function will focus on how to switch on/off upon the load shedding process occurs. Its input variables will be maxLoad and minLoad.

D. Charger Controller Agent

The main task for this agent is to check the status and the battery health with the method percent of Charge Threshold. The distributed load-shedding algorithm is executed on the smart meter. An algorithm to predict the demands, from one hour ahead to a few hours ahead, is essential to control the operations of non-critical appliances. Several real-time algorithms for the forecasting of the electricity demands have been developed [13]. Table 1 describe the main agents and their functions in our system.

Table 1: Functions of the smart DC devices

Device	Functions
The Smart-plug	-Monitors the load inside the house -Controls the operation of the load
The smart-meter	-Predicts the energy demands per house on a given interval (an hour or day ahead) -Rates the appliances (critical, non-critical) -Forecasts the energy demands per appliance (an hour or day ahead) -Proposes a bid -Controls the discharging/ charging of the batteries
Charge controller	-Monitors SoC of the battery -Reports the generated power from the PV panels -Monitors the charging/ discharging of the batteries -Reports on the available power -Determines the battery discharging rate
Microgrid controller	-Activates the load-shedding/balancing process Receives the bids -Allocates the power for each house based on the bids -Determines the available power -Knows/predicts the energy demands for Time (t+1)

The algorithm of our distributed load shedding will follow this flow:

(i) Predict the total amount of energy that needed in the next time interval (prediction) (P_L)

(ii) Estimate using the charging/ discharging rate the total energy available in the next time interval (P_S)

if $P_S<P_L$ initiate the bidding at the load side,

if $P_S>P_L$ initiate the bidding at the storage side.

E. *Load side bidding:*

The auctioneer collects prices from bidders and records all values. In case the bidder with the highest price buys energy $P_b < P_s$ then choose the best-price so that $f = \sum_{i=1}^{n} c_i P_i$ is maximized subject to $P_S \geq \sum_{i=1}^{n} P_i$

F. *Storage side bidding*

The auctioneer collects the selling prices from bidders and records all values. Starting from the lowest to the highest, the auctioneer chooses the total amount of energy that can meet the supplier.

G. *Communications for Distributed Load Shedding*

LoRaWAN is an emerging IoT connectivity protocol. The physical layer of the LoRaWAN uses the spread-spectrum technology with a configurable spreading factor. It uses a low-complex receiver which makes it a suitable technology for M2M communication, IoT, wireless sensor network and the likes. LoRaWAN encapsulates three classes of devices: Class A, class B and Class C. Table 2 shows the feature and application of each class.

Table 2: LoRaWAN device classes

Class	Feature	Application
A	Low-power Often in sleep-mode	End nodes (e.g. fire detection, leakage detection)
B	Battery-powered, low-latency	Sensors, e.g. smart-metering, actuators
C	Mains powered No latency	Low-latency actuators

Numerous research and industrial works have analyzed the latency requirements for the implementation of the demand-response program. However, in the context of smart-grids scant attention has been given to utilizing it in an islanded mode dc-microgrid. Tab 3 summarizes the published data on the latency requirements.

Table 3: Latency requirements for DR programs

Work	Application	latency
Itron Inc	DR-resources	4sec-10 min
Open DR	Regulation or reserve resources	2sec-10 min
Fujitsu	Metering	< 15 secs

IV. FOG COMPUTING ARCHITECTURE

Distributed microgrids and two-way power and information exchange are essential to the future smart grids. There is an intrinsic relationship between fog computing, microgrids, and multi-agent systems. All three are distributed in nature, expected to be autonomous in operation and function even in an islanded mode. Here we will briefly explore the synergy and complementary nature of the three paradigms.

Fog computing is a paradigm that offers computational resources at the user-end of a network. In a fog computing environment, we have scattered computational nodes (routers, switches or surrogate servers) capable of inter-communication and computation across the network path.

This is in complete contrast to cloud computing, where a large pool of resources is merged and managed at a central point. The benefits of having resources closer to the end-user are, greater quality of service for applications and masking cloud/resource outages and scarcity, on the same line as microgrids but for computational infrastructure.

In a fog-microgrid-multi-agent system, fog computing offers the computation and communication layer, microgrid provides the storage and distribution of power, and multi-agent systems manage the distribution and control. Development of such a design offers greater autonomy in operations and management while simultaneously improving efficiency. The benefits of such systems have been proven individually, here we propose an architecture that merges the three.

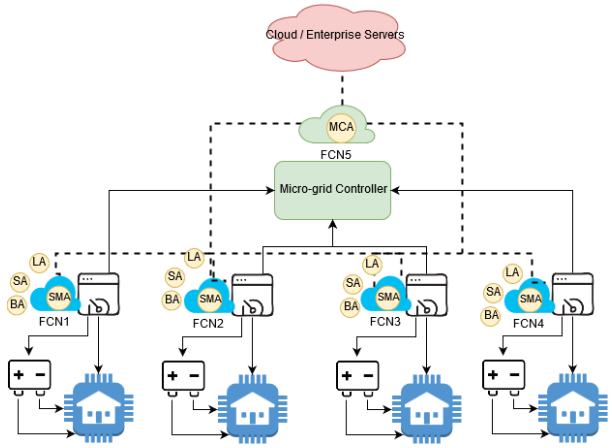

Figure 2: Distributed Microgrid utilizing Fog Computing and MAS

A 2-layer fog architecture for smart microgrids is presented in Figure 2. Each smart-meter is equipped with a computational board (Raspberry Pi 3) capable of acting as a fog node (FCN-1 to FCN-4). Layer-1 consists of these smart meters, where they communicate and offload computation to each other. Layer-2 is a more capable fog-node (FCN-5, surrogate server) with the microgrid controller agent. FCN-5 fog-node offers computational support to the agents running in the microgrid controller, acts as a central peer to layer-1 nodes and gathers, plus transfers data to the cloud/enterprise servers. This fog-node also aids in masking cloud/enterprise server outages. FCN-1 to FCN-4 host the Solar Agent (SA), Battery Agent (BA), Smart Meter Agent (SMA) and Load Agent (LA) independently.

V. EXPERIMENTAL SET-UP AND RESULTS

The application of the distributed load shedding is implemented in the jade platform and raspberry pi device. In this work, the JADE version 4.5.0 and the Raspberry PI 3 are used in the implementation process [11] [12]. Figure depicts the agents' communications whereby each Raspberry Pi has

been installed with jade for load shedding process as well as control strategies. The Raspberry pi 1 is responsible for controlling other microgrids in the system. It is the one which will check which microgrid has enough power to transfer to another, or which one needs to be load shaded. Also, pricing actions are taken care of by this Raspberry Pi. The Raspberry pi 2 focuses on smart meter attributes, smart plug functions, charger control of the battery activities. This is generally the microgrid system. Each microgrid has solar panels, controllers, batteries and loads. Therefore, all attributes of each device are handled by this Raspberry pi. Once the microgrid has extra power to sell, it can share it to the Raspberry pi 3 for communication and hence sell it to the Raspberry pi 1. The third Raspberry Pi is responsible for communication and any faults in the systems. In case, there are any faults among any devices, the Raspberry pi 3 will notify the MG controller for action to take place. The faults in the system are such as low power production which may be due to weather or damage of the solar panels, batteries, or reading errors of the power production.

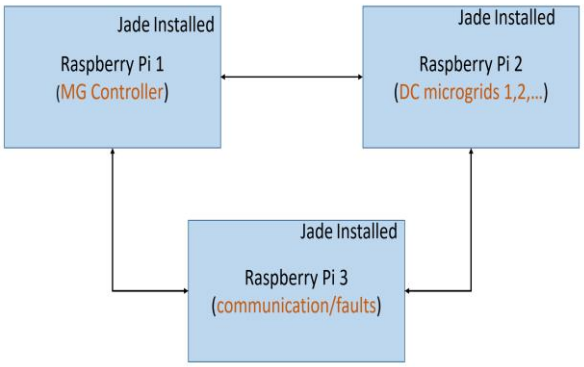

Figure 3: Distributed load shedding implementation

Two classes "TickerBehaviour and CyclicBehaviour" were used for sending and receiving data/messages between agents. For every loop, these classes have to be initiated so that the data can be read and if there is less power in the system, then load shedding will take place. Figure 4 shows one of the DC microgrid used for load shedding technique and demand-side management processes with main supply panels and other solar panels for individual loads. The implementation follows the hardware in loop characteristics to manage the integration between microcontroller and Jade platform. All agents are generated with respect to the characteristics specified earlier. The Raspberry pi 3 output are shown in Figure 5 which demonstrate the initialization process of the main supply agent with the connection of other agents in the system and also the running environment of the load agent.

Figure 4: solar driven DC microgrid at CoICT

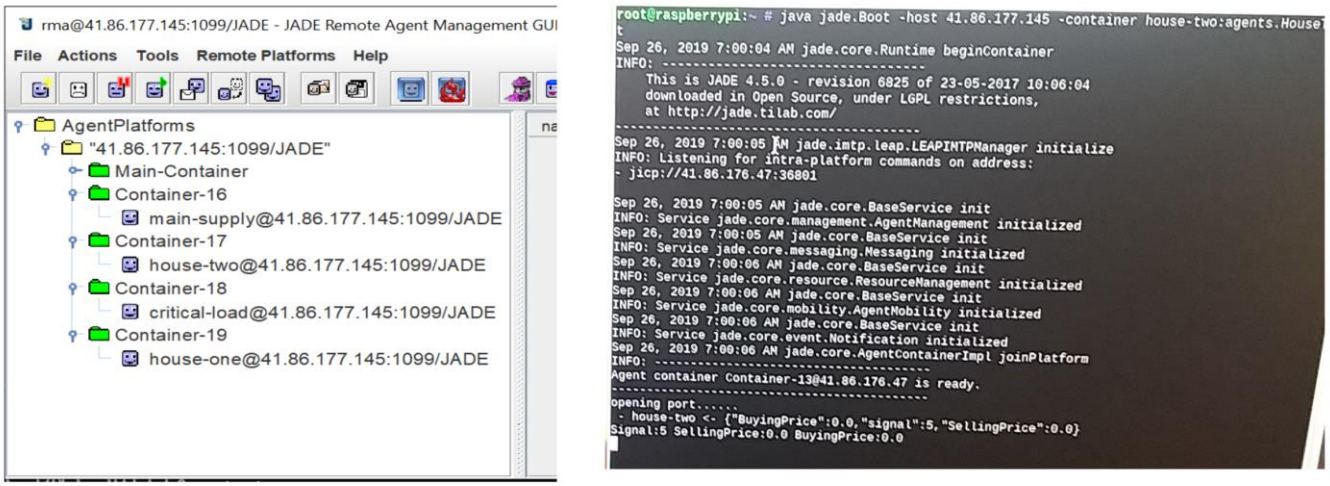

Figure 5: Distributed load shedding implementation (left) showing the Jade environment for all agent and (right) the initialization of the agent

The implementation of the LoRaWAN based fog computing architecture is achieved using Dragino Lora shield v1.4 and the gateway is designed using Dragino LoRa/GPS HAT for RPi v1.4. The gateway was registered at ThingsNetwork.org. The LoRaWAN gateway at the microcontroller listens for the incoming data from the smart meters and forwards the data to an IP server for further processing. In our experiment, the gateway runs the packet-forwarding software only. The smart-meter acts as a LoRaWAN node collects the data from the agents at the smart controllers and the charge controllers and sends it via

LoRaWAN protocol to the LoRaWAN gateway. The experimental results, depicted in Figure 7 shows the network

latency is less than the 20s when using LoRaWAN class B node, and LoRaWAN gateway. A large portion of the latency is attributed to the TTN server.

Besides offering multiple distributed computational nodes, one method that fog computing platforms can utilize for offering more reliability is via computational offloading. In the case of node failures or imminent power outage, the agents or monitoring applications can be migrated to another fog node. The results of our tests for a containerized JVM application (such as JADE), on a single core, 1 GB RAM VM node are presented in Figure 5. With standard migration (shutdown, move, restart) the downtime is 164 to 260 seconds per iteration as seen in Figure 6. Here each iteration is an advanced stage of execution. With live migration, irrespective of the execution stage, the overall downtime stays between 12-15 seconds. This dwontime is well within the tolerance limits expressed in Table 3 for DR applications. Further details of these tests are documented in [13].

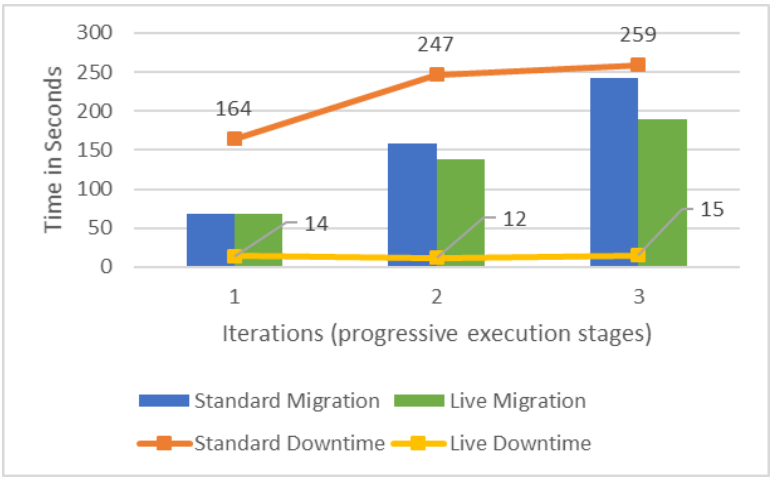

Figure 6: Migration and downtime at various stages of execution

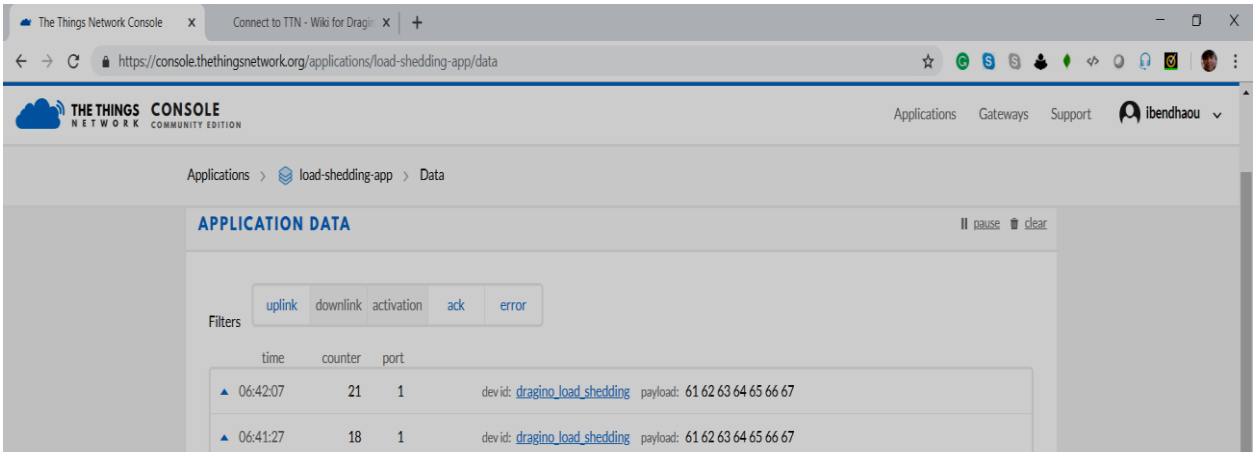

Figure 7: Latency of the LoRaWAN based fog computing

VI. CONCLUSION

In this work, we proposed a fog-based architecture for the implementation of a distributed load-shedding algorithm using a multi-agent system. The architecture uses LoRaWAN for the communication between smart-meter agents and the microgrid controller, and the IP protocol for the communication between the microgrids. The experimental results show that the LoRaWAN protocol and fog computing is a suitable technology for the implementation of edge-centric distributed load-shedding solutions.

REFERENCES

[1] N. Hatziargyriou, Microgrids Architectures and Control. IEEE Press Wiley, 2014.

[2] J. A. Laghari, H. Mokhlis, M. Karimi, A. Halim, A. Bakar, and H. Mohamad, “A New Under-Frequency Load Shedding Technique Based on Combination of Fixed and Random Priority of Loads for Smart Grid Applications,” IEEE Trans. Power Syst., vol. 30, no. 5, pp. 2507–2515, 2015.

[3] D. E. Olivares *et al.*, “Trends in microgrid control,” *IEEE Trans. Smart Grid*, vol. 5, no. 4, pp. 1905–1919, 2014.

[4] H. Wang, H. Jiang, Z. Q. Cai, L. Vinayagam, and Q. Ni, “Development of a DC Microgrid for Enhanced Power System Efficiency and Stability,” 2015.

[5] M. Karimi, H. Mohamad, H. Mokhlis, and A. H. A. Bakar, “Electrical Power and Energy Systems Under-Frequency Load Shedding scheme for islanded distribution network connected with mini hydro,” Int. J. Electr. Power Energy Syst., vol. 42, no. 1, pp. 127–138, 2012.

[6] A. Ketabi and M. H. Fini, “Electrical Power and Energy Systems An underfrequency load shedding scheme for islanded microgrids,” Int. J. Electr. Power Energy Syst., vol. 62, pp. 599–607, 2014.

[7] U. Rudez, S. Member, and R. Mihalic, “Analysis of Underfrequency Load Shedding Using a Frequency Gradient,” IEEE Trans. Power Deliv., vol. 26, no. 2, pp. 565–575, 2011.

[8] J. Tang, G. S. Member, J. Liu, and G. S. Member, “Adaptive Load Shedding Based on Combined Frequency and Voltage Stability Assessment Using Synchrophasor Measurements,” IEEE Trans. Power Syst., vol. 28, no. 2, pp. 2035–2047, 2013.

[9] M. Usman, A. Amin, M. M. Azam, and H. Mokhlis, “Optimal Under Voltage Load Shedding Scheme for a Distribution Network Using EPSO Algorithm,” in 2018 1st International Conference on Power, Energy and Smart Grid (ICPESG), 2018, pp. 1–6.

[10] H. Mortaji, S. H. Ow, M. Moghavvemi, and H. A. F. Almurib, “Load Shedding and Smart-Direct Load Control Using Internet of Things in Smart Grid Demand Response Management,” IEEE Trans. Ind. Appl., vol. 53, no. 6, pp. 1–1, 2017.

[11] Y. Lim, H.-M. Kim, and T. Kinoshita, "

Distributed Load-Shedding System for Agent-Based Autonomous Microgrid Operations", Energies, 7(1), pp.385-401.

[12] D. Rwegasira, I. Ben Dhaou, A. Kondoro, N. Shililiandumi, A. Kelati, and N. Mvungi, “A Multi-Agent System for Solar Driven DC Microgrid,” in International Conference on Control, Electronics, Renewable Energy and Communications (ICCREC), 2017, pp. 252–257.

[13] T. I. Lab, "JAVA Agent DEvelopment Framework," 2000.

[14] D. Sallach, N. Collier, T. Howe, and M. North, “Recursive Porous Agent Simulation Toolkit (Repast),” 2010.

[15] L. Hernandez et al., "A Survey on Electric Power Demand Forecasting: Future Trends in Smart Grids, Microgrids and Smart Buildings," in IEEE Communications Surveys & Tutorials, vol. 16, no. 3, pp. 1460-1495, Third Quarter 2014.