Distribution System Self-Healing Implementation using Decentralized IED-based Multi-Agent System

Jonatas Boas Leite[1], *Member*, *IEEE*, Jose Roberto Sanches Mantovani[1], *Member*, *IEEE* and Mladen Kezunovic[2], Life *Fellow*, *IEEE*

[1]Dep. of Electrical Engineering
São Paulo State University - UNESP
Ilha Solteira, Brasil
mant@dee.feis.unesp.br

[2]Dep. of Electrical and Computer Engineering
Texas A&M University
College Station, USA
kezunov@ece.tamu.edu

***Abstract*—In this paper, local automatic switching plans, considering distribution network topology and hourly loading, are used to implement a self-healing strategy, which minimizes the switching operations to restore the interrupted customers within the shortest time interval. This self-healing implementation is done using decentralized multi-agent system (MAS) where all local switching agents have awareness about the current network state and work cooperatively to reach the global purpose of the distribution network automatic restoration. We are proposing the utilization of intelligent electronic devices (IEDs) as the hosts of the local switching agents. The embedded code for the agent's autonomy, as well as the information exchange among agents are described, implemented and tested.**

***Index Terms*—Electric distribution grid, self-healing, multi-agent system, intelligent electronic device (IED).**

I. INTRODUCTION

The next stage of the future power grid, known as the smart grid, is characterized by two-way flows of electricity and information. The self-healing schemes are meant to work as an immune system able to minimize the service disruption through the coordination of the distributed intelligence in substations and on feeders ensuring high reliability [1]. In the literature, the implementation of self-healing networks employs hybrid control architecture by combining centralized and distributed intelligence [2, 3]. Smart switching devices, communication systems and automatic restoration logic are identified as essential components of many self-healing networks [4] and they may be better utilized when designed as a multi-agent system (MAS).

The multi-agent system is typically not based on a centralized control approach but instead deploys agents that are autonomous, distributed and cooperative. Agents perceive their environment, persist over a prolonged time period, adapt to changes, and create and pursue goals [5]. Several contributions in the literature propose a layered control structure with cooperative MAS for self-healing operations. In [6], agents are associated with two locational layers: zone and feeder. A similar architecture is proposed in [7] where the MAS solution is divided in the three functional layers focused on the response, coordination and organization. These layers compose a centralized agency that works as a computational tool for reducing the processing time of the restoration algorithms executed in the control centers. The internal agents just perform their functions once triggered by the circuit breaker tripping. In a centralized MAS solution, there may be a significant time interval between circuit breaker disarming and restoring the interrupted customer loads because of the time delay resulting from the processing of centralized control actions. An example of the typical centralized MAS architecture is the agency described in [8] where the control agent interacts with other types of internal agents, such as database agent and data analysis agent, in order to manage the available information.

The intelligent electronic device (IED) implementation based on the IEC 61850 standard assumes that the hardware typically has sufficient computational resource for supporting the operation of an agent [9], which is a software entity able to sense and react to changes in the environment. In the decentralized agency, the self-healing coordination agent is expected to have the knowledge of the network topology and hourly loading that are used for modifying the perception of the local switching agents according to their embedded autonomous algorithms located inside IEDs.

In our previous work [10], the self-healing strategy is implemented using a decentralized MAS architecture where the self-healing coordination agent works as the consciousness of the local switching agents by modifying their perception according to their beliefs and design objectives. In this paper, we describe an implementation of the local switching agent using a commercial IED. The theoretical concepts involved in the agent design are converted to control equations that make the realization of the proposed autonomous algorithms possible.

The paper is organized as follows. Section II presents the proposed MAS architecture supporting the self-healing strategy implemented using the local switching agents. Section III discusses the results obtained through two case studies. Section IV contains the conclusions based on the observations resulting from the proposed experiment.

This work was fully supported by the São Paulo Research Foundation – FAPESP (grant: 2014/22377-1 and 2015/17757-2).

978-1-5386-5844-4/18/$31.00 ©2018 IEEE

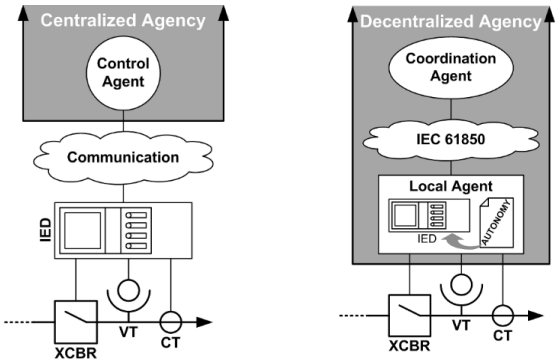

Fig. 1. MAS architectures for self-healing strategies: (a) centralized and (b) decentralized agency.

II. Implementation of the Local Switching Agent

Figure 1 illustrates the difference between centralized vs decentralized agent implementation, referred further as the "agency". In Fig. 1(a) the primary sources for information monitoring are IEDs. The information from these devices is gathered by a control agent through a communication network and shared to other internal agents. Fig. 1(b) presents an expanded agency exceeding the boundaries of the traditional control center implementation, which leads to the development of a decentralized MAS architecture for self-healing networks pursued in our work.

In the layered and decentralized MAS architecture, as illustrated in Fig. 2, there is a two-way information flow where measurements go from the physical (Ph) to the intelligence (I) layer and, then, return as control parameters to the switching agents. The IEC 61850 standard implementation on IEDs provides computational resources that guarantee the information exchange among local switching agents and the self-healing coordination agent [11].

A. Foundations of the Local Switching Agent

In addition to communication with other agents, switching agent monitors instantaneous values, such as current and voltage magnitudes, that are then employed in the fuzzy controller for inferring decisions [12]. At the fuzzification stage, instantaneous values produce grades of membership using linguistic terms of fuzzy sets. Most membership functions have a triangular shape and can be generically represented by the mathematical model given by (1).

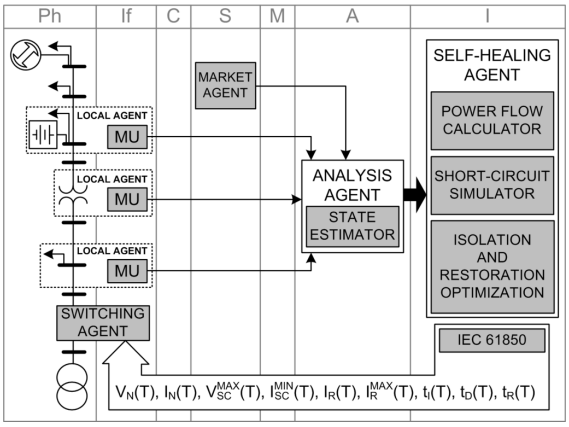

Fig. 2. Self-healing information flow in the architecture of smart grid [10].

$$\mu_G(x,a,b,c) = \begin{cases} 0, & x < a \\ \dfrac{x-a}{b-a}, & a \le x < b \\ \dfrac{x-c}{b-c}, & b \le x < c \\ 0, & x \ge c \end{cases} \quad (1)$$

In (1), μ_G is a generic membership function of the x independent variable where a, b and c are scalar parameters. This model is employed to mathematically represent all membership functions of the local switching agent, *e.g.*, the membership function of the isolation time corresponding to $\mu_{IT}\big(t, t_{TR}(T), t_I(T), t_D(T)\big)$. The next stage of the fuzzy controller design is the inference procedure that generates a numeric response using grades of membership and a set of rules. The premise and proposition of each rule compose the knowledge base and they are mathematically modeled by (2) and (3), respectively.

$$r_k(i,v,t,c) = \min_{1 \le j \le N_k} \{\mu_{G_j}\} \quad (2)$$

$$P_k(\mu_P) = \{p | \mu_P = 1\} \quad (3)$$

The value of the k[th] premise, $r_k(i,v,t,c)$, is calculated using instantaneous measures of current flow, i, voltage profile, v, time, t, and remote command, c, fuzzified in the N_k membership functions. The calculation of the k[th] proposition, $P_k(\mu_P)$, uses resulting membership functions, μ_P, for converting the numeric response into linguistic variables. Both values are used by the fuzzy controller in the defuzzification procedure as given by (4).

$$p_C(i,v,t,c,\mu_P) = \frac{\sum_{k=1}^{n_r} r_k(i,v,t,c) P_k(\mu_P)}{\sum_{k=1}^{n_r} r_k(i,v,t,c)} \quad (4)$$

The defuzzified value, $p_C(i,v,t,c,\mu_P)$, also called the fuzzy centroid, is the numeric response of the fuzzy controller action and should reflect the decisions made by the local switching agent. The fuzzification procedure, premise calculations and defuzzification procedure are theoretical foundations for implementing the switching local agent using available commercial IED computational resources.

Many commercial IEDs can accommodate a set of control equations to customize protection operation, create custom operation elements and automate substation operation. The control equations for programming include advanced features, such as free-form logic, math operations and sequencing timers, that should make the implementation of the local switching agent possible. Particularly, in this work, the development of autonomous algorithms is done using a commercial IED that includes two main setting areas for optional programming [13].

The protection area has six setting groups with 250 lines for free-form control equation programming. The automation area has ten blocks with 100 lines for free-form programming that are executed sequentially from the first block. The protection and automation areas have different intervals of execution. The

former has a fixed interval of execution that equals to 1/8 power signal cycle while the last has an execution time that varies according to the amount of free-form logic expressions in the automation blocks. This way, the fuzzy controller algorithm is inserted into protection area whereas one automation block is responsible for the communication with the self-healing coordination agent via IEC 61850 protocol.

B. Data Exchange Logic

The selected relay supports "vertical" and "horizontal" communications using Ethernet and IEC 61850 standard:

1) Vertical: control service is available for transferring values of remote bits (*RB01-RB32*) from supervisory control and data acquisition (SCADA) system to IED;

2) Horizontal: communication via GOOSE message can be mapped using remote analog inputs (*RA001-RA256*) as incoming messages, and remote analogs outputs (*RAO01-RAO64*) as outgoing messages.

Similarly to free-form programming areas, the vertical and horizontal communications have different execution time, *e.g.*, transmission of digital data begins within 2 ms for GOOSE messages. Thus, the information exchange among local agents should be performed using the GOOSE messages while the communication with the coordination agent should be established via client/server scheme using control service because the adjustment of membership functions occurs continuously for each time interval T that can vary from 15 minutes to 1 hour.

Since the control service just permits the transfer of 32 remote bits and the membership function adjustment requires the use of analog scalar parameters, the transferred parameters should be codified using the IEEE 754R standard that defines the floating-point arithmetic [14]. Fig. 3 shows the codification of 16 remote bits for providing a floating-point with half-precision. The codified information has three parts: sign with 1 bit; exponent with 5 bits; and fraction with 10 bits. In (5), the floating-point, fp, is calculated using the transferred information in decimal format where $s = \{0,1\}$, $-15 \leq pp \leq 15$, and $0 \leq mmm \leq 1023$.

$$fp = (-1)^s \times 1.mmm \times 2^{pp} \tag{5}$$

The relay does not provide support for power math operation, hence the floating-point point equation is modified through the insertion of the Naperian logarithm and exponential functions from the supported set of math functions.

$$fp = (1 - 2s) \times 1.mmm \times e^{pp \times ln2} \tag{6}$$

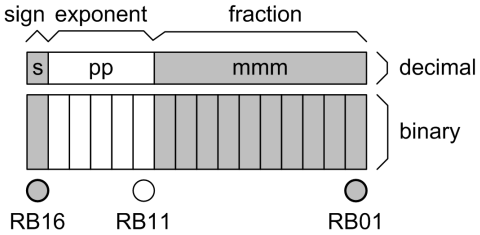

Fig. 3. Codification of the floating-point with half-precision.

In addition to the codification of (6) in the automation area, the definition of an approach for transferring the vector of floating-points, $\boldsymbol{fp}(T) = \{I_R^{MAX}(T), I_R(T), I_N(T), I_{SC}^{MIN}(T)$ $V_{SC}^{MAX}(T), V_N(T), t_{TR}(T), t_R(T), t_{I(T)}, t_D(T)\}$, is necessary. Fig. 4(a) presents a transition diagram of asynchronous serial communication where the remote bit *RB17* works as a flag for starting the storing of the received floating-point message. At the same time, the counter *ACN01CV* is incremented by one unit. When *ACN01CV*, or *i*, is equal to *n*, which is the size of vector $\boldsymbol{fp}(T)$, the transference of all adjustment parameters is finished. The complete vector $\boldsymbol{fp}(T)$ is then sent to the protection area for adjusting the membership functions as shown in Fig. 4(b) where the execution time is the interval needed to run all automation blocks sequentially.

C. Fuzzy Controller Logic

The first step of the fuzzy controller is the fuzzification procedure. Its implementation requires a previous mapping of membership function parameters because control equation elements, such as storage locations, timers and counters, cannot be renamed [13]. Table I presents an example of mapping where scalar parameters $t_{TR}(T)$, $t_I(T)$, and $t_D(T)$ are mapped into the control equation math variables $PMV01$, $PMV02$, and $PMV03$, respectively. The instantaneous measurement of the

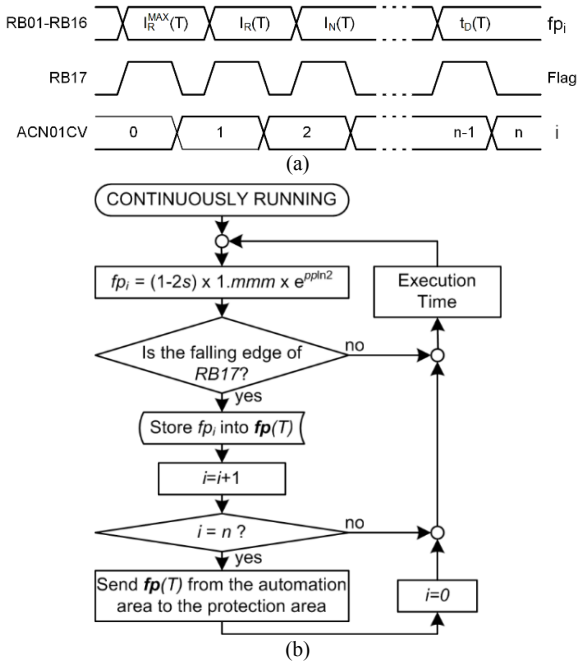

Fig. 4. Asynchronous serial communication: (a) transition diagram, and (b) algorithm flowchart.

TABLE I
MAPPING OF THE ISOLATION TIME MEMBERSHIP FUNCTION.

Control Equation Elements	Parameters of Equation (1)	Membership Function Parameters [10]
$PMV01$	a	$t_{TR}(T)$
$PMV02$	b	$t_I(T)$
$PMV03$	c	$t_D(T)$
$PST01ET$	x	t
$PMV04$	μ_G	μ_{IT}

time, t, is achieved using a sequencing timer that provides the time accumulated through the math variable, $PST01ET$.

After the mapping, the membership function described by (1) must have the graphical representation as illustrated in Fig. 5(a). Discontinuities in the membership function require the use of *IF…*, *THEN…*, *ELSE…* routines that are not included in the set of programming equations. Thus, function discontinuities are marked using Boolean variables ($PSV01$, $PSV02$ and $PSV03$) and comparison operators (>, <, =, <=, >= and <>) to provide logical results, 0 or 1, by comparing two floating-point values. The first continuity interval is obtained by the logical 1 of the $PSV01$. In terms of free-form control equation programming, the first continuity interval is given by (7).

$$PSV01 \coloneqq PST01ET < PMV01 \tag{7}$$

The computation of other variables, such as $PSV02$ and $PSV03$, is similar to $PSV01$; however the second continuity interval is obtained through the logical operation given in Fig. 5(b). Analogously to $PSV04$, the third interval is given by $PSV05$ in Fig. 5(c). After the determination of continuity intervals, the calculation of membership function values, which are different to zero, are achieved as in (8).

$$\begin{aligned} PMV04 \coloneqq & \left(\frac{PST01ET - PMV01}{PMV02 - PMV01}\right) PSV04 \\ & + \left(\frac{PST01ET - PMV03}{PMV02 - PMV03}\right) PSV05 \end{aligned} \tag{8}$$

According to Table I, the $PMV04$ variable represents the linguistic variable of isolation time. The division of the time axis in four intervals suggests the existence of three more linguistic variables that must be obtained using a formulation similar to (8). The fuzzification of measured instantaneous values, such as current flow, voltage profile and remote command, is done by replicating the explained approach.

Premise calculations employ linguistic variables as given by (2) where a function for getting the minimum value among four floating-point values is necessary. Because the relay does not support the minimization function, the implementation of premise calculations is done using the comparison operator (<) and Boolean variables. Fig. 6(a) presents a way for programming the minimization function with free-form control equations. If four linguistic variables, μ_{G_j}, are sequentially

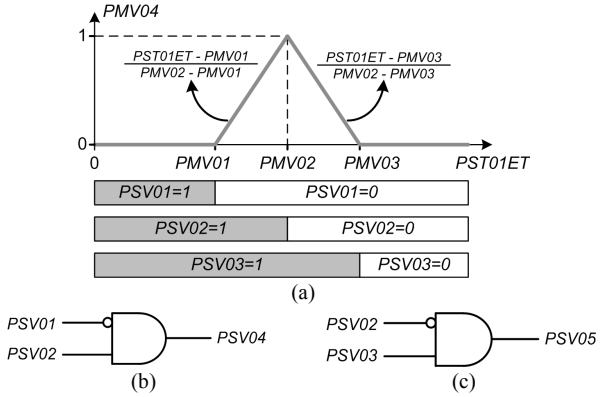

Fig. 5. Fuzzification procedure: (a) graphical representation of the membership function, (b) computation of *PSV04*, and (c) of *PSV05*.

```
01 PSV06 := PMV04 < PMV05
02 PMV08 := PMV04 * PSV06 + PMV05 * NOT PSV06
03 PSV06 := PMV06 < PMV07
04 PMV09 := PMV06 * PSV06 + PMV07 * NOT PSV06
05 PSV06 := PMV08 < PMV09
06 PMV10 := PMV08 * PSV06 + PMV09 * NOT PSV06
```

(a)

```
06 PMV20 := PMV20 + 2.00 * (PMV08 * PSV06 + PMV09 * NOT PSV06)
07 PMV21 := PMV21 + PMV08 * PSV06 + PMV09 * NOT PSV06
```

(b)

Fig. 6. Free-form control equation programming: (a) minimization code, and (b) accumulation of the numerator and denominator.

mapped into control equation elements, from $PMV04$ to $PMV07$, then the math variable, $PMV10$, stores the minimal value among them, *i.e.* the value of the k[th] premise, $r_k(i, v, t, c)$.

Although the calculation of all premises is possible, the minimization code requires many program lines. The non-repeating of comparison operations and the beginning of defuzzification procedure during the premise calculations contribute to reducing the amount of programming lines. In Fig. 6(b), the 6[th] line is modified through the addition of the math variable, $PMV20$, that accumulates all multiplications of premise by proposition. The constant value, equals to 2.00, is the known numeric response provided by linguistic variable of the resulting proposition, $P_k(\mu_P)$. Therefore, the variable, $PMV20$ is mapped as the numerator while $PMV21$ is mapped as the denominator of the fraction given by (4) that calculates the fuzzy centroid and determines the decisions taken by the local switching agent.

III. Results of Laboratory Investigation

The laboratorial implementation of the proposed MAS architecture employs two earlier described relays, which are connected via an Ethernet switch into the smart grid simulation platform, [15] and [16]. Numerical results are obtained through two study cases: one checks the fuzzy controller behavior under a short circuit event simulated in the distribution network. The other one verifies the peer-to-peer communication among local switching agents.

A. Study Case I: Short Circuit

In the short circuit study case, the local switching agent is built using one IED as merging unit (MU) and another one as fuzzy controller, Fig. 7(a). The mapping of the monitored variables are: current magnitude into $RA064$; voltage magnitude into $RA062$; agent timer into $PST01ET$; remote command into $PMV25$; and defuzzified response into $PMV31$. The remote command and defuzzified command are numerical representations of linguistic directives, *e.g.* $PMV25$ equals to 0, the *Open()* directive, equals to 1, *No_Command()*, and equals to 2, *Close()*. Analogously, $PMV31$ equals to 0 is the *Trip()* directive, equals to 1 is *Send_Alarm()*, equals to 2 is *Start_Timer()*, equals to 3 is *Wait()*, and equals to 4 is *Clear_Alarm()*. Fig. 7(b) shows that the switching agent remains in the wait state until the occurrence of the short circuit on the third cycle, when the agent starts its internal timer. After ¼ power signal cycle, the switching agent sends an alarm signal to coordination agent informing it about the abnormal event. Then, the local switching agent makes the tripping as a result of the absence of remote command characterizing an adjacent permanent fault.

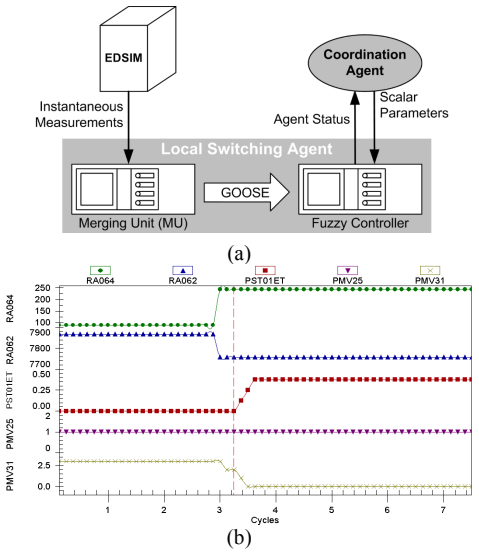

Fig. 7. Local switching agent behavior after the short circuit event.

B. Study Case II: Peer-To-Peer Communication

In the peer-to-peer communication study case, Fig. 8(a), two IEDs work as local switching agents that communicate with the self-healing coordination agent using the client/server model, *i.e.* the SCADA communication. The total time for transferring all scalar parameters is around 14s with time interval of 200ms between the consecutive transference of two scalar parameters because the execution of the automation programming blocks demands an average time of 85ms. In this way, the coordination agent can update 64 local agents whenever the period *T* is equal to 15 minutes. Furthermore, the local switching agents exchange information among them via peer-to-peer communication, *i.e.* GOOSE messages.

Remote commands are transferred using both vertical and horizontal communication. Fig. 8(b) demonstrates the local switching agent behavior when a remote switching agent sends the *Open()* directive using GOOSE message. Initially, the local agent is in waiting state until it receives the GOOSE message that enforces the state change independently from other inputs.

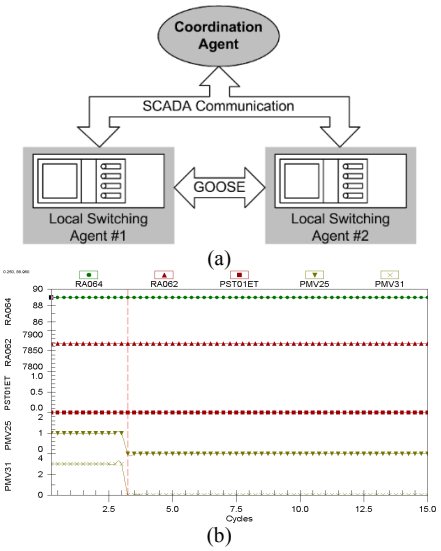

Fig. 8. Local switching agent behavior after the GOOSE message.

Thus, immediately after the message is received, the local switching agent makes the decision of tripping.

IV. Conclusion

The following are the paper contributions:

- The implementation of the local switching agent using a commercial IED based on IEC 61850 protocol reveals the feasibility for expanding the MAS agency beyond the operational control center domain.
- The requirements of communication and artificial intelligence are reached in this paper utilizing the proposed methodology described earlier in [10].
- The implementation illustrates the limitation of the local switching agents realization using the commercial IEDs due to the limited number of available programing lines allowing in our case just the codification of the fuzzy controller for the phase A.

References

[1] S. Borlase, "Smart Grid Technologies," in *Smart Grids: Infrastructure, Technology and Solutions*, 1st ed., vol. 1, Boca Raton: CRC Press, 2013, pp. 67-494.

[2] A. Bernardo, *et al.*, "Preventive Assessment for Combined Control Centre and Substation-Centric Self-Healing Strategies," in *21st International Conference on Electricity Distribution – CIRED 2011*, pp. 1-4.

[3] C. McCarhy, *et. al*, "Smart Distribution through Layered Intelligence for Next Generation Self-Healing Distribution Networks," in *22nd International Conference on Electricity Distribution – CIRED 2013*, pp. 1-4.

[4] L. Maurer, A. Stevens, and W. Reder, "Tales from the Frontline: Keys to Successful Self-Healing Distribution Projects," *IEEE Power Energy*, vol. 10, pp. 100-106, Apr. 2012.

[5] M. N. Huhns and L. M. Stephens, "Multiagent Systems and Socities of Agents," in *Multiagent Systems: A Modern Approach to Distributed Artifical Intelligence*, 3rd ed., vol. 1, MIT Press, Massachusetts, USA, 2001, pp. 79-120.

[6] A. Zidan and E. F. El-Saadany, "A cooperative multiagent framework for self-healing mechanism in distribution systems," *IEEE Trans. Smart Grid*, vol. 3, no. 3, pp. 1525-1539, Set. 2012.

[7] H. Liu, *et. al*, "The control and analysis of self-healing urban power grid," *IEEE Trans. Smart Grid*, vol. 3, no. 3, pp. 1119-1129, Set. 2012.

[8] D. P. Buse, *et. al*, "Agent-Based Substation Automation," *IEEE Power Energy Mag.*, vol. 1, no. 2, pp. 50-55, Mar. 2003.

[9] Ratan Das, *et. al*, "Distribution Automation Strategies: Evolution of Technologies and the Business Case," *IEEE Trans. Smart Grid*, vol. 6, no. 4, pp. 2166-2175, Jul. 2015.

[10] J. B. Leite and J. R. S. Mantovani, "Development of a Self-Healing Strategy with Multiagent Systems for Distribution Networks," *IEEE Trans. Smart Grid*, vol. 8, no. 5, pp. 2198-2206, Sept. 2017.

[11] J. B. Leite and J. R. S. Mantovani, "Study of the IEC 61850 protocol on multiagent systems for power system applications," in *Proceeding IEEE PES General Meeting*, Denver, CO, 2015, pp. 1-6, 2015.

[12] F. M. Mcneill and E. Thro, "Fuzzy systems on the job," in *Fuzzy logic: a practical approach,* 1st ed., Chestnut Hill, MA, Academic Press, 1994, pp. 57-82.

[13] *SEL-451-5 Relay: Protection, Automation and Control Systems*, Schweitzer Engineering Laboratorie, Inc.. 1-1198, Apr. 2013.

[14] *IEEE Standard for Floating-Point Arithmetic*, IEEE Std. 754-2008, Aug. 2008.

[15] J. B. Leite and J. R. S. Mantovani, "Development of a Smart Grid Simulation Environment, Part I: Project of the Electrical Devices Simulator," *J. Control Autom. Electr. Syst.*, vol. 26, n.º 1, pp. 80-95, Feb. 2015.

[16] J. B. Leite and J. R. S. Mantovani, "Development of a Smart Grid Simulation Environment, Part II: Implementation of the Advanced Distribution Management System," *J. Control Autom. Electr. Syst.*, Vol. 26, no. 1, pp. 96-104, Feb. 2015.