25th ABCM International Congress of Mechanical Engineering
October 20-25, 2019, Uberlândia, MG, Brazil

COB-2019-0054
DEVELOPMENT OF SUPERVISION, CONTROL AND CONDITION-BASED MAINTENANCE SYSTEM FOR PHOTOVOLTAIC GENERATION PLANTS WITHIN THE IOT CONCEPT

Antonio Bernardo de Vasconcellos Praxedes
Alberto José Alvares
Programa de Pós-Graduação em Sistemas Mecatrônicos, Departamento de Engenharia Mecânica, Faculdade de Tecnologia, Universidade de Brasília, Asa Norte, Brasília, DF, Brasil.
antoniobernardopraxedes@gmail.com, albertojalvares@alvarestech.com

Abstract. *In current times, where electricity is an essential good for human life, practically every type of work and leisure depends on it. Thus, the consumer invests in micro solar generation with basically two objectives: economy in the invoice and uninterrupted energy, and expects the satisfactory return, which depends fundamentally on the proper functioning of the equipment. However, the complexity of these types of equipment, coupled with the fact that the majority of the consumers are not technicians results in the fragility of the operation and maintenance of the system. Thus, the idea of a low-cost system that could be installed even in small power plants, but that was able to provide the functionalities of supervision, control and condition monitoring in a reliable and friendly way, speeding the detection and resolution of anomalies that may happen. In this way, this work has two main objectives: 1. To develop a system of supervision, control, and condition based maintenance applied to photovoltaic solar generation plants; 2. Develop a communication protocol for supervision and control using IP networks, within the context of condition monitoring and Internet of Things (IoT).*

Keywords: *Internet of Things, IP networks, Fuzzy logic, Condition monitoring, Photovoltaic systems.*

1. INTRODUCTION

The normative resolution 482/2012 of Brazilian National Electricity Agency (ANEEL) allows consumers to generate their electricity from renewable sources, and may even supply the surplus energy to the distribution network of their locality, thereby generating credit with the concessionaire and receiving a rebate on its invoice. Since the publication of this resolution, the capacity of mini and microgeneration distributed in Brazil has grown exponentially, reaching at the beginning of 2019 the official number of more than 46,000 units, with more than 143 new installations made per day. Thus, installed generation capacity jumped from 0.53MW in early 2013 to 572MW in early 2019. Most of the microgeneration plants installed in Brazil are on-grid photovoltaic type, which operates in conjunction with the energy of the concessionaire. In addition to these, some plants operate independently, called "off-grid," which are often not included in the official numbers because they do not require homologation.

A solar generation plant generally has one or more sets of photovoltaic panels connected to one or more equipment intended to convert the energy supplied by the panels as efficiently as possible. In on-grid power plants, which are the most widely used today, a dual function equipment, called "String Inverter", is used, which directly receives the output of a set of photovoltaic panels, searches for the supply point of maximum power (MPPT function), converts to mains voltage (127 or 220VAC), and controls the injection of current directly into the consumer network. In this type of plant, generation problems can go unnoticed, because if there is a reduction or interruption in the photovoltaic generation, the power of the network is automatically used.

Off-grid plants, which operate independently of the utility's network, are more complex. These plants have a main DC bus, where the battery bank, inverters, DC loads and charge controllers are connected. The charge controllers are devices intended to convert the energy supplied by the photovoltaic panels to the voltage and current required by the batteries and the loads as efficiently as possible. This feature is called MPPT function (Pinho, 2014). This type of power plant can be used in locations that do not have a utility grid. In places where the grid is available, this type of power plant usually requires load management, because depending on the solar generation capacity, both inverters can feed the AC loads or the grid. Off-grid power plants are also used as solar UPS, providing uninterrupted power during the day through photovoltaic panels and at night, in the absence of the grid, through the batteries. In addition to the monitoring of the generation capacity of the panels, this type of plant also requires the monitoring and supervision of the other equipment to guarantee a full operation.

Thus, with the objective of avoiding the reduction of photovoltaic generation and damage to the plant, the idea arose of the development of a set of resources that allow the supervision, control, and monitoring of condition of the equipment, that is applicable to photovoltaic generation plants, and that is adherent to the Internet of Things (IoT) concept (Bedi, 2018). The Internet of Things is a paradigm of recent communication that foresees a near future in which objects of everyday life will be equipped with microcontrollers, transceivers for digital communication and adequate protocols that will enable them to communicate with one another. and with users, becoming an integral part of the Internet (Zanella, 2014). Today, the availability of low-cost, high-performance controllers and computers with their communication and connectivity capabilities allow them to integrate the most diverse devices and systems into local networks and the Internet.

The first objective of this work is to carry out the feasibility study of a low cost supervision, control and condition-based maintenance system applied to photovoltaic generation plants. Analysis tools based on fuzzy logic and if-then rules should be used to estimate the operating conditions of each priority equipment. These estimates shall be calculated in real-time and shall be robust enough to monitor the operating conditions of the priority equipment and to take the necessary maintenance actions in the shortest possible time.

The second objective is to study the performance and compatibility of a standardized communication protocol based on standard OSA-CBM binary messages encapsulated in CoAP protocol encapsulated in UDP packets. The idea of assembling this protocol was motivated by the need for a feature that was viable for small controllers like Arduino, while at the same time complying with the standards of communication protocols used in state-of-the-art supervision, control and condition monitoring. Thus this study is to show the viability of a communication feature that can be installed in thousands of plants across the country that can make information available locally or remotely through cloud applications within the concept of the Internet of Things.

2. THE CONDITION BASED MAINTENANCE (CBM) AND THE OSA-CBM STANDARD

Reliability Centered Maintenance (RCM) is a process used to determine the maintenance requirements of any physical asset in its operational context. A subset of the RCM is Condition Based Maintenance (CBM), which is an evidence-based maintenance technique of necessity, and the RCM provides the rules of evidence. CBM is used to actively monitor the health of components or the system in order to optimize operating costs and determine maintenance needs before a major failure occurs. The CBM seeks to reduce operating costs and increase security by optimizing the interval between maintenance and minimizing downtime. The result of these procedures is increased system availability and overall reliability. In a typical CBM environment, monitoring software receiving information from a set of sensors embedded in the process determines when maintenance tasks are necessarily based on the evidence of need (Gillespie, 2015).

The use of open systems is fundamental to modern CBM techniques. The implementation of CBM systems requires the integration of several devices, often from several suppliers, which is made easier through standardization, especially in terms of communication between devices. Therefore, standardization in information exchange results in improved interoperability and reduced costs. Thus, some institutions have been working to consolidate standards that can be used by CBM systems. Among these institutions is the International Organization for Standardization (ISO), the Institute of Electronics and Electrical Engineers (IEEE) and the Society of Automotive Engineers (SAE). However, the Open Systems Architecture (MIMOSA) organization has developed a standardized architecture for data transfer, specifically for the CBM environment called OSA-CBM (Open Systems Architecture for Condition Based Maintenance), defined in the Specification for Binary OSA-CBM Messages Release V1 (Mapping OSA-CBM into a binary format). MIMOSA – Open Standards for Physical Asset Management.

OSA-CBM architecture defines a standard structure based on six functional blocks. Each block is described as well as the interface between these blocks, specifying the inputs and outputs, which facilitates the integration of several components. In summary, the OSA-CBM standard describes a standardized information exchange system for MBC procedures, composing a systematic method to treat sensor signals, processing these signals to obtain the information, generating the system integrity report and maintenance guidelines. The OSA-CBM standard was defined using the Unified Modeling Language (UML) in its format. This standard can be implemented according to the needs of the systems, such as using high-speed protocols in binary format for real-time use, or internet tools. The specification of the OSA-CBM messages that map the abstract UML can be assembled in the form of a binary message, which is much more efficient and compact than a message composed of ASCII characters of the XML format (MIMOSA, 2010). The efficiency in the composition of the standard binary message OSA-CBM encapsulated in User Datagram Protocol (UDP) makes this set possible to be used efficiently by small controllers such as Arduino.

3. THE DEVELOPMENT OF THE COAP-OSA-CBM PROTOCOL

In this work, another communication protocol option was developed for IoT systems: the CoAP-OSA-CBM protocol, which is the Constrained Application Protocol (CoAP) protocol, which is widely used in IoT systems, carrying binary messages standard OSA-CBM. The acronym CoAP (Constrained Application Protocol) refers to the use in small

systems of supervision, control, and automation, even with 8-bit processors and low memory capacity such as Arduino controllers.

The CoAP protocol has the same HTTP access methods (GET, PUT, POST, and DELETE), which facilitates the mapping between the two protocols. The CoAP, defined by RFC 7252 Constrained Application Protocol 2014, is of general use, that is, in its cargo area called Payload, it can carry information regarding various applications and various equipment. Thus, in this case, was defined as a set of messages for supervision, control, and condition monitoring within the OSA-CBM standard. The message of the CoAP protocol is of the binary type, consisting of a fixed 4-byte header followed by three fields: 1) Token; 2) Options; 3) Payload. Using the UDP protocol to load messages requires the implementation of procedures to make packet delivery reliable. Thus, the type of Confirmable message, together with the procedures for verifying the received message and re-transmitting attempts in the event of error detection, provides the reliability of delivering the messages correctly. The Token field is intended to identify each pair of information request messages and the corresponding response. In this work, the Payload field loads the binary messages OSA-CBM with the data of supervision and control, composing the new protocol called CoAP-OSA-CBM. All messages in this protocol are encapsulated in User Datagram Protocol (UDP) packets for traffic on standard IP networks. Table 1 below shows the four fields of a CoAP protocol message. Only the Header field is required for all messages:

Table 1 – CoAP Protocol Messages General Format

Size	Field
4 bytes	Header
Up to 8 bytes	Token – Sincronize requests and responses
Up to 12 bytes	Options
Up to 65000 bytes	Payload – carries CoAP-OSA-CBM messages

The CoAP protocol defines four types of messages regarding the request and response procedures: Confirmable, Non-Confirmable, Recognition, Reset. The codes for message types can cause them to load requests or responses. An acknowledgment message, sent in response to a request made by a confirmatory message, may load the requested data to expedite communication procedures. In this way, the CoAP protocol aims to allow the exchange of information in the most efficient way possible with small controllers and integrate them to the resources of the Internet. Tables 2 and 3 below shows the CoAP protocol layers and the general structure of OSA-CBM messages:

Table 2 – CoAP Protocol Layers

Application Information and Procedures
Requests and Answers Procedures
Composition of CoAP Messages
Encapsulation of CoAP Messages in UDP Protocol

Table 3 – General Structure of the Binary OSA-CBM Standard Message

Message Field	Bytes	Description
Header	1-N	Header bytes are used to indicate the type of message being transmitted and various uses of optional information components.
Message Length	0,4	Contains the total number of bytes of the message.
Number of Blocks	0-2	Contains the amount of data blocks sent. Usually only one block is required.
Message Identifier	1,2,4,8	Identifies a specific message type for a platform.
Platform Identifier	0,3+	Identifies the platform from which a message is generated.
Time	2+	This field contains date / time information in ISO 8601 format.
Information Content Block	3+	This block contains equipment and system data, which includes information on error checking or synchronization. The block has variable length and variable content type. The header byte has the code that identifies the type of information in the block.
...Data Block…	...	There may be up to 65,535 data blocks. In this project only one block is used.

OSA-CBM binary messages of equipment status, measurements, digital status, parameter setting and digital command messages have been implemented.

2.1 The Multiprotocol Gateway Raspberry PI 3

In addition to the CoAP-OSA-CBM protocol, the Raspberry PI 3 server also provides the IEC61850 protocol of the OSGP platform. The Open Smart Grid Platform (OSGP) standard is an open, generic and independent platform that aims to integrate electrical devices into the IoT concept, connecting applications (Web) with equipment through an IP network. These devices can be public lighting, transportation, metering, solar and wind generation, and power quality. The OSGP standard provides the following protocols: 1) Device Language Message Specification (DLMS) for smart metering; 2) IEC61850 for public lighting, microgeneration, and distribution automation; 3) OSLP (Open Street Light Protocol) for microgeneration and public lighting.

Thus, the Raspberry PI 3 also has the Multiprotocol Gateway Function, serving as a protocol converter to allow the integration of this solar power plant with more IoT systems. Figure 1 below shows the communication diagram:

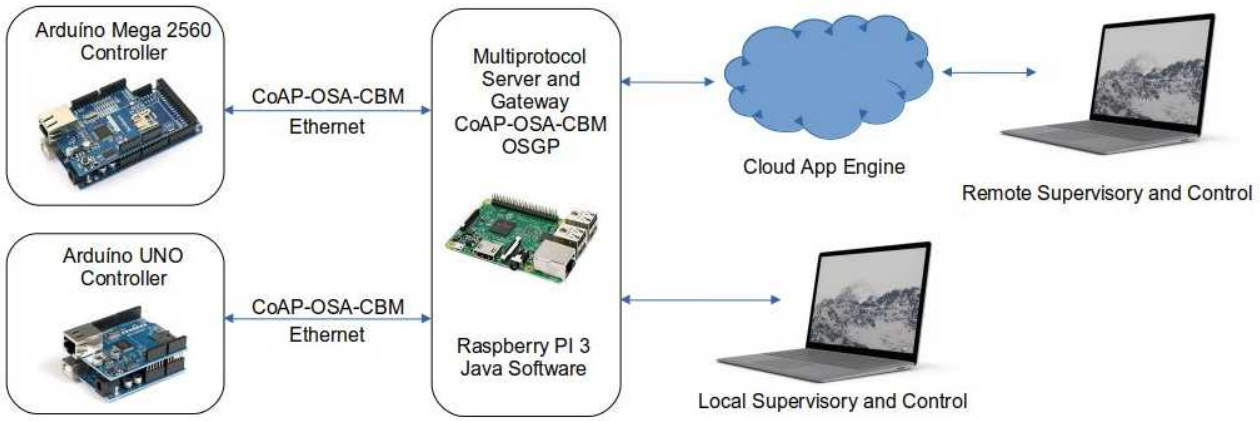

Figure 1. Communication Diagram of Supervisory, Control and CBM System

4. METHODOLOGY AND IMPLEMENTATION

4.1 The Development and assembly of Low-Cost electronic devices for Signal Conditioning

All signal conditioners have been designed and assembled, taking into account the need for reliability and reasonable accuracy coupled with low cost. Thus, all digital and analog supervision signals were converted to read by Controller 1 model Arduino Mega 2560 at a voltage in the range of 0 to +5 volts. Controller 1 has 16 analog inputs with 10-bit resolution for measurement. These inputs provide raw values read by the software, in the range of 0 to 1023 counts, referring to voltage values of 0 and +5 Volts respectively. Also included in the controller are 54 pins of digital inputs or outputs, which are configured by the software according to their function (input or output).

The Controller 2, model Arduino Uno, has an interface converter for dual RS485 standard, which is used for connection to the Charge Controllers through an asynchronous serial interface at the rate of 115200bps. Each controller has an Ethernet interface shield with WiZ5100 integrated circuit, which allows the connection of these devices in the 10/100BaseT Ethernet network.

4.2 The Development of Arduino Controllers Software

The programming of Arduino Controllers is done in C / C ++ language using the Integrated Software Development Environment (Arduino IDE). The Arduino IDE software was installed in the Raspberry PI 3 computer, being accessed through VNC program over the Ethernet network. The programs that run on the Arduino controllers were made within the concepts of real-time programming and state machine. The Controller 1 performs the procedures for reading and processing the signals from all physical points of digital inputs and analog inputs provided by the signal conditioning circuits, and is also capable of control digital command outputs and performing the primary protection procedures of the plant equipment. The Controller 2 makes the communication in MODBUS RTU protocol with two Charge Controllers, reading and converting measurements related to photovoltaic generation.

4.4 The Development of the Raspberry PI 3 Server Software

The software running on the Raspberry PI 3 computer was developed in Java using the Eclipse 2019-03 Integrated Development Environment (IDE). The Eclipse IDE generates a JAR executable file that runs in the Java Runtime

Environment (JRE). The development of this software was done using a computer with Intel® Core ™ i5 1.80GHz × 4 processor, 4GB of RAM and Ubuntu 18.04.2 LTS operating system. This software performs three main tasks: 1) It communicates with the two Arduino controllers using the CoAP-CBM-UDP protocol in 10/100BaseT ethernet network; 2) It Performs fuzzy inference and fault analysis procedures for equipment condition monitoring; 3) It Performs communication and conversion of protocols with the higher levels of supervision and control.

The Fuzzy inference machines, which perform condition monitoring procedures, use the FCL - Fuzzy Control Language, described in the IEC 61131-7 standard. The jFuzzyLogic_v3.0 package installed in the Eclipse Java Development Environment allows programs that execute commands from an FCL type text file. Thus, the variables declaration, fuzzification procedures, defuzzification procedures, and fuzzy inference procedures are written in FCL command language and executed by a standard Java program through the use of fuzzy class objects.

4.3 The Development of Supervisory and Control Software

Supervisory and Control software was developed using Java language in the Eclipse IDE 2019-03. The purpose of this software is to communicate with the Raspberry PI 3 server using the CoAP-OSA-CBM protocol through the network, to present all measurements and states of the plant, also allowing the sending of commands. Figure 2 below shows the screen of the local supervisory and control computer:

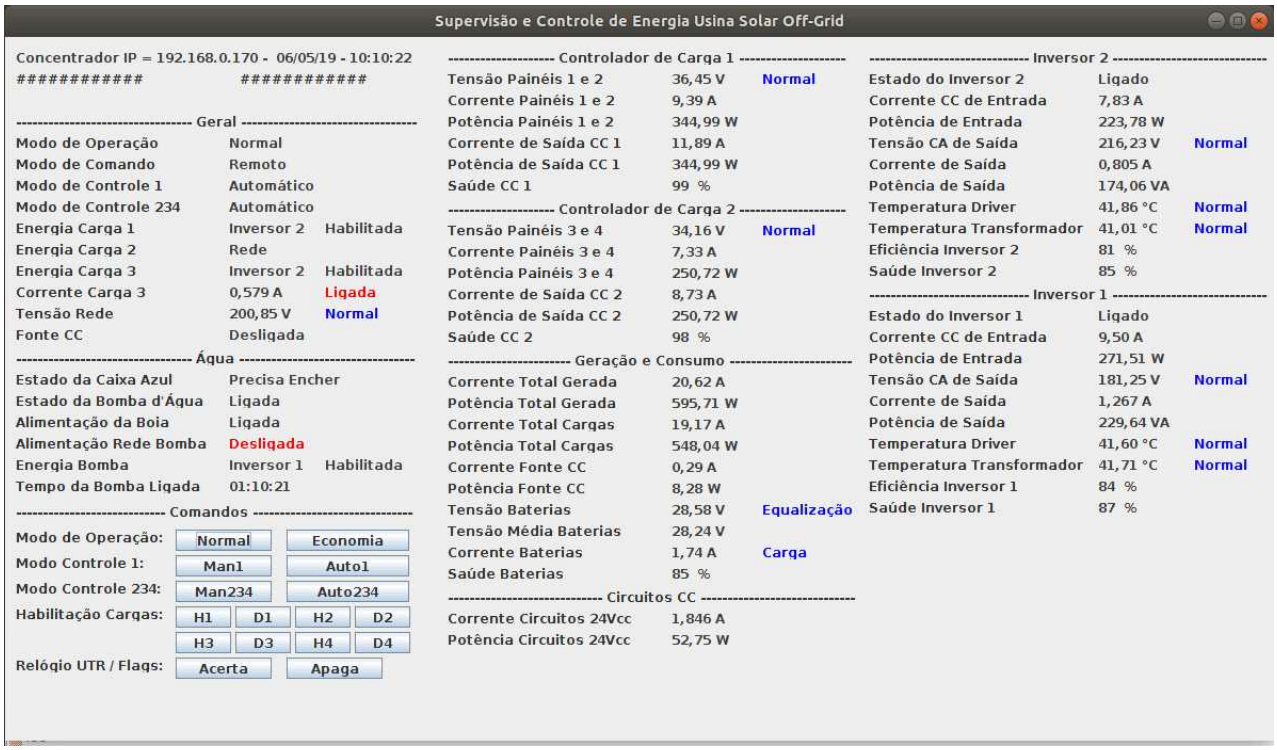

Figure 2. Supervisory and Control Software screen with plant's information

4.5 The Development of the Condition Monitoring Algorithm

The Primary Protection Procedures are performed by Controller 1. These procedures are algorithms for detection of overcurrent, overvoltage, undervoltage, and overtemperature performed by comparison with pre-established limits. In this case, the controller may disconnect the equipment that is at immediate risk to avoid further damage. The devices most prone to failures are the inverters because they are connected directly to the AC network.

However, several types of failures do not cause a severe fault immediately, but they begin with a progressive degradation in the functioning of the equipment, which demands an analysis tool that allows identifying these situations. For this, the Fuzzy Control System Based on Rules was chosen, in which the control outputs are used as device condition monitoring variables (Ross, 2010). The Rules not only model the system through its characteristics but are defined using a technician's knowledge of the process. In this way, the value resulting from an output variable can be presented to the user by informing a particular operating condition and also generating fault identification reports and recommending maintenance procedures. The leading utility of fuzzy inference systems lies in the fact that results are approximated to human language, facilitating user analysis of the system.

In this project, condition monitoring procedures were implemented for the following equipment: 1. Photovoltaic panels; 2. Charge Controllers; 3. Inverters; 4. Battery Bank. Regarding the instrumentation, in addition to the traditional voltage, current and temperature sensors, a small photovoltaic panel was installed whose purpose is to measure the incident light power per area called Irradiance (Markvart, 2003). In this way, it is possible to evaluate the power generation capacity of the photovoltaic panel assemblies in real time. The following is an example of the inverter condition monitoring procedure in its simplified version.

5. IMPLEMENTATION EXAMPLE: INVERTER CONDITION MONITORING

5.1 Definition of the Input Variables and Fuzzy Sets

The first step in the assembly of a fuzzy control system is the definition of the parameters of the system to be modeled. The input variables must provide the information necessary to enable the implementation of the condition monitoring procedures. The Fuzzification is the process of making fuzzy an exact quantity (crisp). Exact quantities carry considerable uncertainty when analyzed from human reasoning. Thus, fuzzy sets allow imprecise properties of pertinence rather than standard sets. In this way, numbers are exchanged for expressions of the type: low, high, medium, normal, etc., where each one represents a fuzzy set.

The basic supervision information for the inverter is the voltage, current and temperature measurements and the on/ off status. The calculated quantities input power, output power and efficiency are used in this situation as they reflect the energy absorbed and supplied by the equipment. Therefore, the VAR_INPUT section of the FCL file for the monitoring of each inverter received the five REAL variables shown below:

```
We:   REAL;  // We = Input Power
Ws:   REAL;  // Ws = Output Power
Ei:   REAL;  // Ei = Inverter Efficiency (Output Power / Input Power)
Td:   REAL;  // Td = MOSFETs Drivers Temperature
Tt :  REAL;  // Tt = Transformer Temperature
```

The fuzzy sets of the input variables were chosen in a standardized way, that is, reflecting the intensity or normality or abnormality condition. The type of function chosen was triangular or trapezoidal, where the lower end is always zero, and the upper one is the maximum value that the variable can assume. Figure 3 shows the fuzzy sets of two input variables: Input Power and Efficiency. The input power of Inverter 1 can vary from 0 to 500 W. Thus; the fuzzy sets were defined as Low, Medium, High and Very High. The inverter used in this design, has an efficiency of around 80%, due to the one stage architecture using a low-frequency transformer. For this reason, the fuzzy set for Normal efficiency has a maximum value equal to 80%. The Normal set decreases to zero at 70%, where the Medium set has its maximum value. The Low set starts at 70% efficiency and has a maximum value of 60% up to zero, as shown in Figure 3 below:

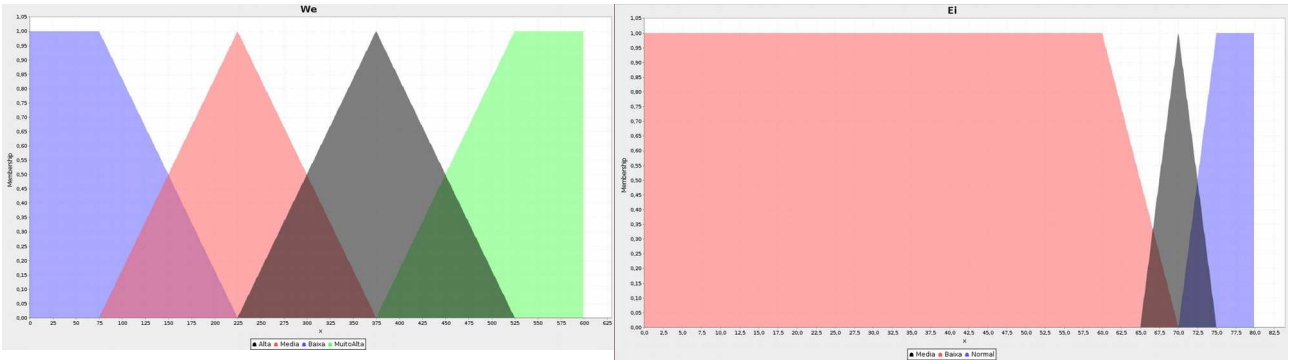

Figure 3. Fuzzy Sets – Input Power (left) and Efficiency (right)

5.1 Definition of the Output Variables and Associated Fuzzy Sets

The criterion for defining the output variables should take into account the proposed objectives within the concept of condition-based maintenance. In this way, the first step is to use expert knowledge to identify the operating context of the equipment. An inverter is a piece of electronic power equipment that receives, converts, and supplies power to various types of loads. The power supplied can vary significantly within the operating range. Operation of the inverter leads to heating of the MOSFETs and transformer heatsinks. Thus, it is desired to detect two essential conditions of abnormality that are the abnormal heating and the fall in efficiency. Using this information was defined as an output variable called InverterHealth that reflects the degree of an anomaly. Four fuzzy sets defined in a range of 0 to 100%

were chosen, indicating the degree of "health" of the inverter (Critical, Alert, Attention, and Normal). Figure 4 below graphically shows the function for the InverterHealth output variable, where the "Critical" set is further to the left near zero and the "Normal" set is rightmost close to 100%. The definition of these sets is done in the DEFUZZIFY block of the FCL file:

```
DEFUZZIFY   InverterHealth
  TERM  Critical   := (0, 1)  (20, 1)  (40, 0);
  TERM  Alert      := (20, 0)  (40, 1)  (60, 0);
  TERM  Attention := (40, 0)  (60, 1)  (80, 0);
  TERM  Normal   := (60, 0)  (80, 1)  (100,1);
END_DEFUZZIFY
```

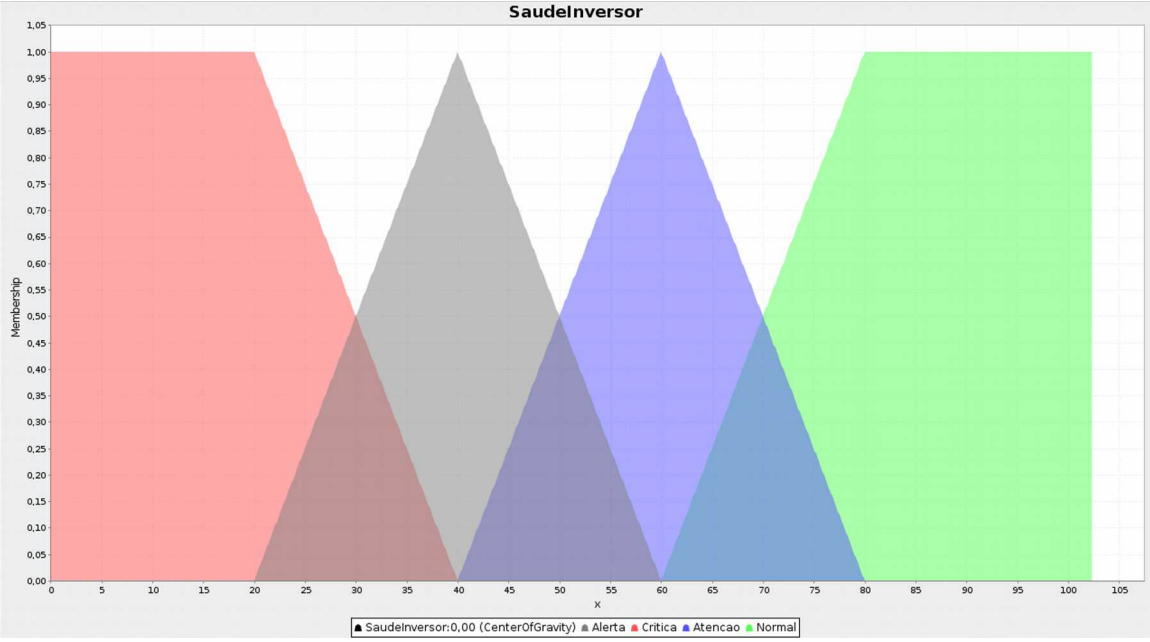

Figure 4. Fuzzy Sets – "Health" of Inverter

5.1 Definition of the Inference Fuzzy Rules

The Fuzzy Inference System has the function of processing the fuzzy input variables using the rules if-then stored in the knowledge base. The definition of fuzzy-control rules of the if-then is not as simple as in the case of classical logic since fuzzy logic allows gradations in the intensity of assertions. Thus, the result of applying an if-then fuzzy rule are membership values to some sets. The value of the intensity or strength of the rule is obtained by a combination of the intensities of the fuzzy input variables. In this example, an output variable called InverterHealth has been defined. Therefore, one must have a set of rules to activate this variable. The following is an example of a simple set of rules in the FCL language:

```
RULEBLOCK Rules1
  AND      : MIN;
  ACT      : MIN;      // Use 'min' activation method
  ACCU     : MAX;      // Use 'max' accumulation method
  RULE 1  : IF ((Ei IS Low) OR (Td IS VeryHigh) OR (Tt IS VeryHigh)) THEN InverterHealth IS Critical;
  RULE 2  : IF ((Ws IS Low) OR (Ws IS Medium)) AND (Ei IS Normal) AND ((Td IS High) OR (Tt IS High)) THEN InverterHealth IS Alert;
  RULE 3  : IF (Ws IS Medium) AND (Ei IS Normal) AND ((Td IS High) OR (Tt IS High)) THEN InverterHealth IS Attention;
  RULE 4  : IF (Ws IS VeryHigh) THEN InverterHealth IS Attention;
  RULE 5  : IF (Ws IS Low) AND (Ei IS Normal) AND ((Td IS Normal) OR (Tt IS Normal)) THEN InverterHealth IS Normal;
  RULE 6  : IF (Ws IS Medium) AND (Ei IS Normal) AND ((Td IS Normal) OR (Tt IS Normal)) THEN InverterHealth IS Normal;
  RULE 7  : IF (Ws IS High) AND (Ei IS Normal) AND ((Td IS Normal) OR (Tt IS Normal) OR (Td IS High) OR (Tt IS High))
                THEN InverterHealth IS Normal;
  RULE 8  : IF (Ws IS VeryHigh) AND (Ei IS Normal) AND ((Td IS Normal) AND (Tt IS Normal)) THEN InverterHealth IS Normal;
END_RULEBLOCK
```

5.1 Defuzification, Presentation and Treatment of the CBM Inverter Information

The defuzzification interface transforms the fuzzy control actions resulting from the application of the Fuzzy Inference Machine rules into a normalized numerical value and also generates membership values to the defined output sets. The value of the membership function for each set of the output variable shows the Fuzzy result of the inference procedure. In the case of the Inverter Operation, the defuzzified value informs the "health" of the equipment, that is, the lower this value, the lower the "health" and the higher the evidence of abnormal functioning. In the supervisor's screen, Figure 2, there is the value of the health variable of the inverters being presented in real time. Under normal conditions, this value is expected to always be above 80%. The following is an example of a report of the result of applying the fuzzy rules and the defuzzification provided by the Java function *fis.getVariable*. The method used for defuzzifiyng was *CenterOfGravity*. The "Latest defuzzified value" parameter refers to the value of the HealthInverter2 output variable in percentage (83.84%). The values of the relevance of the output variable to the fuzzy sets are presented below. In this case, the value of the variable HealthInverter2 belongs 100% to the set "Normal," indicating normal conditions of operation for the Inverter 2.

```
HealthInverter2 :
  Defuzzifier : CenterOfGravity
  Latest defuzzified value: 83.84213036565937
  Default value: 0.0
  Term: Alert      0.0      PieceWiseLinear : (20.0, 0.0) , (40.0, 1.0) , (60.0, 0.0) ;
  Term: Critical   0.0      PieceWiseLinear : (0.0, 1.0) , (20.0, 1.0) , (40.0, 0.0) ;
  Term: Attention 0.0      PieceWiseLinear : (40.0, 0.0) , (60.0, 1.0) , (80.0, 0.0) ;
  Term: Normal   1.0      PieceWiseLinear : (60.0, 0.0) , (80.0, 1.0) , (100.0, 1.0) ;
```

6. CASE STUDY

The case study is an off-grid power plant with a maximum solar generation capacity of 1280Wp provided by four poly-crystalline photovoltaic panels of 320Wp/72 cells and two charge controllers with a maximum power of 780W each. The storage capacity at 24Vdc is 345Ah with sealed acid lead batteries. Two single-phase SPWM Inverters power the 220VAC 60Hz loads. The block diagram is shown in Figure 5 below:

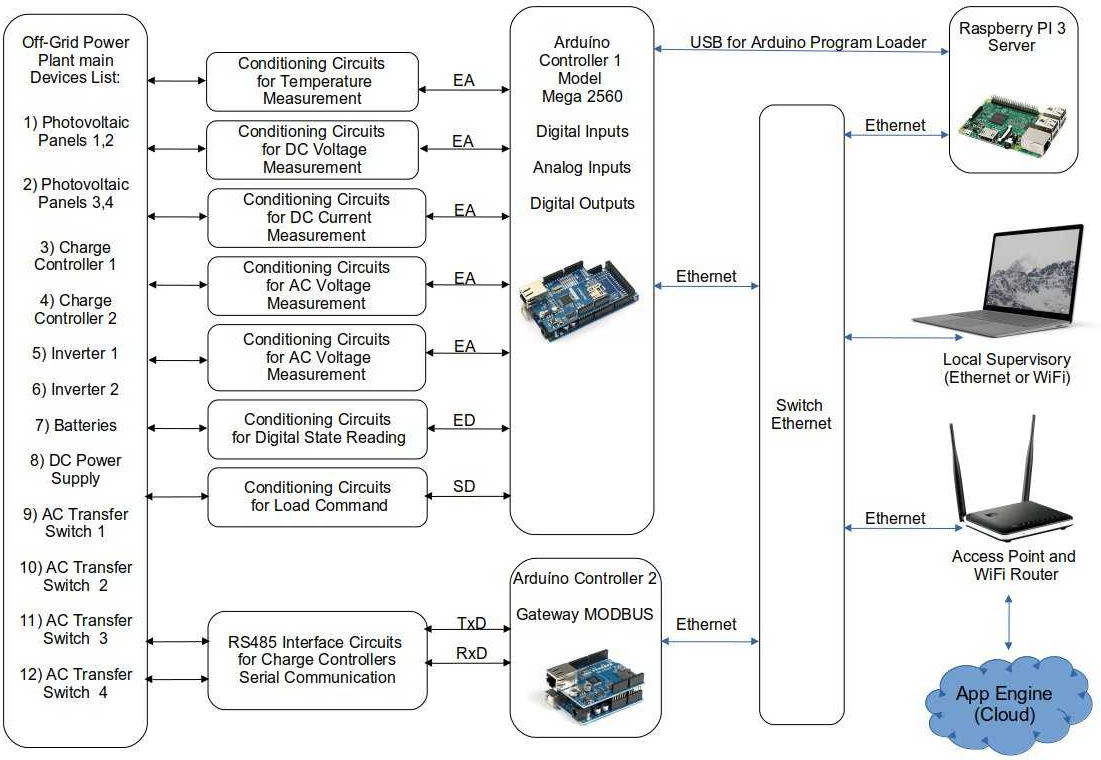

Figure 5. Block Diagram of Supervisory, Control and Communication of the Plant

The system also feeds a set of priority loads in 24Vdc, consisting of security cameras, network infrastructure equipment, supervision, control, and condition monitoring.

This plant works as a solar no-break, feeding a house with two people and a water pump in the rural area of Brasília-DF. The solar power supply is made in conjunction with the utility's supply through a charge management system, which provides uninterrupted power and economy.

7. RESULTS

The condition monitoring functionality allowed the identification of two more severe faults: the detection of an open MOSFET power transistor in the driver of an inverter and the decrease of the generation capacity in a set of photovoltaic panels caused by occasional shadowing. In the case of the inverter, the driver H-bridge, composed of four groups of two MOSFETs connected in parallel, had a MOSFET open, causing abnormal heating and dropping efficiency. This type of abnormality is complicated to detect since the other connected MOSFET in parallel supports all the current and maintains the operation of the equipment. The reduction in the value of the variable InverterHealth to 70% indicated some abnormality, which was identified through a more detailed inspection. The decrease in the capacity of the panel set, detected by reducing the value of the Panel1Health variable, was solved by pruning a tree without further problems. This failure is also challenging to detect without condition monitoring capabilities.

The supervision and control functionality greatly facilitated the operation and maintenance of the plant, also contributing to the development of load management procedures and equipment monitoring. The use of the CoAP-OSA-CBM protocol proved to be quite suitable for use by the Arduino controller, causing no processing overhead even on this eight-bit controller. Raspberry PI 3 server software interrogates Arduino controllers every 500ms, with no significant reduction in processing power of the Arduino controllers. Communication performance testing was done on the Raspberry PI 3 server software by measuring the time between the request and the response received from the Arduino controllers.

The cost of the solar plant used as a case study was R$ 21,000.00, including material and installation. The cost of the supervision, control, and MBC system was R$ 4,000.00, which does not include the computer used for development. This cost is feasible even for a small plant like this, with only four photovoltaic panels.

8. CONCLUSIONS

As of the date of this work, the plant has been in operation for one year, constantly improving and testing within this research work. User benefits include primarily anticipation and resolution of failures, and time savings because with just one view you can check the condition of the entire system and take necessary action if necessary. Specifically in this case study, was also implemented the supervision and control of the water supply system, obtained by pumping a deep tubular well.

The calculation of equipment health through the application of fuzzy rules proved to be efficient in detecting abnormalities in operation. Using the standard FCL text file has greatly facilitated the constant refinement of detection procedures by including new rules.

In short, this feature has become very important in the operation and maintenance of the plant, allowing its full operation and maximum use of available resources.

9. REFERENCES

Bedi, G. et al., 2018. “Review of Internet of Things (IoT) in Electric Power and Energy Systems”. *IEEE Internet of Things Journal*, Vol. 5, No. 2, April 2018.

Gillespie, A., 2015. “Condition Based Maintenance: Theory, Methodology, & Application”. *Conference: Reliability and Maintainability Symposium* at: Tarpon Springs, FL. January 2015.

Markvart, T. and Castafier, L., 2003. *Practical Handbook of Photovoltaics*. Elsevier Ltd. ISBN 1856173909

Pinho, J. T. et al., 2014. *Manual de Engenharia para Sistemas Fotovoltaicos*. Centro de Referência para Energia Solar e Eólica Sérgio de Salvo Brito – CRESESB - Cepel – Eletrobrás.

Ross, T. J., 2010. *Fuzzy Logic with Engineering Applications*. John Wiley & Sons, Ltd. ISBN: 978-0-470-74376-8.

Zanella, A. et al., 2014. “Internet of Things for Smart Cities”. *IEEE Internet of Things Journal*, Vol. 1, No. 1.

10. RESPONSIBILITY NOTICE

The authors are the only responsible for the printed material included in this paper.