Abstract— With the evolution of the Substation control paradigm to distributed architectures, and creation of the IEC 61850 standard, emerges the need to reintroduce in the Substation Automation System (SAS) the automation functions, modeled by Petri Nets (PN), developed in the 80s. Therefore, it is necessary to match the standard with the PN, which motivated the present paper with the objective of developing methodologies of automation functions implementation over the IEC 61850. The method developed to solve the problem uses a logic controller communicating with the IEDs, through GOOSE messages (defined in the IEC 61850) using an Ethernet Local Area Network. The Logic controller is based on a SoftPLC, being programmed directly according to the IEC 61131-3 languages, and therefore allowing the use of automatic translation tools, that translate the PN to the IEC 61131-3 according to formal existing algorithms that guarantee the faithful execution of the PN models. With the objective of proving the correct execution of the system, it was built a laboratorial test environment, in which was simulated part of a Substation through the use of several IEDs and proved the correct execution of the automation functions and good results in real-time performance.

Index Terms— IEC 61850, IEC 61131-3, Petri Nets (PN), Intelligent Electronic Device (IED), Substation Automation System (SAS).

I. INTRODUCTION

The Portuguese Power Utility sector has been evolving in the past few years, being proven by the increasing Power Quality [1,2]. A key element is the constant innovation in the Protection, Command and Control Systems present in Substations, where electromechanical relays gave place to electronic devices, based in microprocessors, and therefore with digital processing, named Intelligent Electronic Device (IED). The IEDs are hardwired to Circuit Breakers (CB), Current Transformers (CT), and Voltage Transformers (VT), performing protection and control functions, as well as, event and oscillographic recording.

The SAS were built to quickly and effectively respond to the Grid incidents, such as faults, under frequency, under voltage, without human operators. In [3] were developed automatisms for SAS of Distribution Substations, specifically to MV feeders. These automatisms are modeled by Petri Nets, which are a powerful mathematical tool for describing relationships between logical conditions and discrete events, therefore supports the modeling and analysis of the control systems [3, 4].

Being properly analyzed, proving the automatisms good behavior and coordination, these PN specifications were implemented in multitask software environment, presented in [6], which originated a centralized SAS prototype, the “Autómato de Subestações”.

However, over the past years the paradigm for power substations control evolved into distributed SAS, using the IED’s capabilities and modular automatisms, outdating the centralized hardwired SAS. In the new methodology, the coordination between automatisms is made through interlocking, which is too restrictive, or even compromising of the correct coordination, this actually is one of the reasons that support the automatisms modeling with PN, as presented in[3, 7]. In the meantime, with the creation of the IEC 61850, regarding to “Communication Networks and Systems in Substations” [8], and by taking advantage of the programmable logic capabilities provided by the IEDs, emerge the need to reintroduce the PN designed automatisms back to the SAS by matching the PN with the IEC 61850.

In [5] in presented a work that tries to solve this problem, however it concludes that the IEC 61850 can’t achieve the proposed objectives, and so it is proposed a modification of the standard, which goes against own concept of international standard.

In this paper is presented the developed solution to solve the problem - a standardized methodology to implement, over the IEC 61850 environment, these automatic control functions, on the distributed SAS present in modern substations, with the faithful execution of the automation functions, modeled by PN.

II. AUTOMATION FUNCTIONS AND PETRI NETS

A. Automation Functions

The Automation Functions discussed in this paper are presented in the EDP Standard-Project [9], as the essential automation functions for MV Lines. The operation of such functions requires information, concerning to the electric grid and substation status, called for specific conditions, and act on the substation through circuit breaker (CB) operational
Under Voltage Load Shedding (UVLS) – This function reacts to the loss of power supply at the bus level, disconnecting simultaneously all the feeders, being reconnected after the normalization of the power supply. The main objective of this function is to avoid the sudden rise of load cause by the simultaneous connection of all loads.

Under Frequency Load Shedding (UFLS) – The UFLS, while acting on the feeders of the distribution Substations, reacts to the Electric Grid’s global disturbances, specifically to the unbalance between generated and consumed power. The Load Shedding aims to restore the balance, with the objective of avoiding blackouts and the disconnection of synchronous generators.

Automatic Reclosure (AR) – This automation function reacts to faults in the distribution lines, trying to eliminate them. Experimental data shows that about 82% of the line faults are eliminated by instant CB trip, followed by a timed pause to allow the arc deionization, and then a fast reclosure; 10% are eliminated by a first trip that allow the feeder temporized protection relay followed by a reconnection after 15 to 45s, representing a slow reclosure; and about 1,5% for a second slow reclosure.

The incidents that trigger the automation functions are random, so it can occurs the simultaneous start of several functions, which can generate decision conflicts that represents a problem, especially in equipment control, specifically to this paper, the CBs from the MV feeder.

To avoid such conflicts, in [3] this functions were improved and specified together in one system, modeled by Petri Nets, with three primary requirements:

- Each Function should operates as freely as possible;
- The CB reclosure is made when every other individual function allows it;
- The reclosure is made when every automation function have finished their operation.

B. PN Definitions

The Petri Nets are a mathematical tool, to describing the relation between conditions and events, allowing the modeling and analysis of automatic control functions [3, 4, 5].

Definition 1: A PN is a five-tuple defined by \( M = (P, T, I, O, \mu_0) \); Where \( P = \{p_1, p_2, ..., p_n\}, n \geq 0 \); is a finite set of places; \( T = \{t_1, t_2, ..., t_m\}, m \geq 0 \); is a finite set of transitions; \( I: P \times T \rightarrow N_0 \), is the input function; \( O: P \times T \rightarrow N_0 \), is the Output function; \( \mu_0: P \rightarrow N_0 \), is the initial marking.

The input and output functions link places to transitions and transitions to places, through oriented arcs, being the value of these functions, the arc weight. To each place is associated a token, representing the place marking. The marking set of all places represents the PN marking.

The graphical representation of a PN is a bipartite graph, being the places represented by circles and the transitions by bars. Oriented arc link places and transitions according to the input and output functions. The marking (token) representation is a dot inside the circles.

Definition 2: a transition \( t \) is fireable to a marking \( \mu_t \) if, \( \mu_t \geq I (.t.i) \), that means, if every input places from that transition have a number of token at least with equal weight of the corresponding arc (pre-condition).

The firing of a transition \( t \), fireable to a marking \( \mu_t \), is made in a single operation, removing from each place in the set \( L I (L, t) \) tokens, and adding to each place in the set \( P O (P,t) \) tokens.

Definition 3: An interpreted PN is a PN \( M = (P, T, I, O, \mu_0) \), an operational domain \( DOP = (D, OP, PR) \) and two functions \( \Psi \) and \( \Phi \), which establish a relation between the PN (control domain) and the DOP (operational domain). D is a set of operational domain’s states, where:

- \( OP = \{op_1, op_2, ..., op_k\}, k \geq 0 \); is a set of operators
- \( opi: D \rightarrow D \);
- \( PR = \{pr_1, pr_2, ..., pr_l\}, l \geq 0 \); set of logical predicates over D;
- \( pr_j : D \rightarrow \{T, F\} \); T and F represent the logical values True and False;
- \( \phi: P \rightarrow OP \), Associates to each place from M an operator over the operational states;
- \( \psi: T \rightarrow (PR, OP) \), Associates to each transition from M an ordered pair (predicate, operator).

An interpreted PN represents a system, in which the state is a pair \( (d, \mu) \), where \( d \in D \) e \( \mu \) is the PN marking.

In the interpreted PN, the firing of a transitions \( t \), is made by verification of the precondition and checking if the logical association of predicates, assigned to the corresponding transition, have the logical value “True” (logic proposition).

C. Modeling with Petri Nets

The modeling or specification of a system, through Petri Nets, is performed by assigning system’s conditions to places and events to transitions. The Place’s marking represents if the system condition is active, being the PN marking the representation of the system’s status.

As in a real system, where the conditions and events dictate the system operation, in a PN, the operation is based on transitions’ firing, which change the PN marking and therefore the system status.

The application of PN in the SAS has been subject for several papers and thesis, standing out [3] regarding the application in Distribution Substations. In the referred thesis, were developed communicating automation functions, modeled by interpreted PN, and analyzed, with computer aid
verification, in a way that the system good behavior was confirmed.

From these analyses were determined some important Petri Nets Properties:

- Safe, for every reachable marking, the number of tokens in one place never exceeds one;
- Live, for every reachable marking, exists at least one fireable transition;
- Deterministic, the PN conflicts are solved;
- Determined, the operations assigned to transitions are compatible.

It was also verified the places and transitions invariance, to assure that some conditions and/or events didn't occur simultaneously.

The reduced version of the referred PN, corresponding only to the feeder automation functions has 88 places and 154 transitions.

III. TRANSLATION AND PROGRAMMING OF PETRI NETS

A. IEC 61131-3

With the evolution of the control engineering, what was controlled by Human operators nowadays is controlled automatically with logical decisions [12]. For this new form of control, were developed dedicated computers, the Programmable Logic Controllers, named PLCs.

These logic controllers gained popularity in the industry, dominating among the other control technologies. In this way, in 1993, is created the IEC 61131, a standard for PLCs development environment. The third part of the standard describes the programming languages [13].

The adoption of the standard unified the programming, lead to the extinction of the manufacturers’ proprietary languages, splitting the programming services from the equipment manufacturing.

The languages defined in the IEC 61131-3 are:

- Ladder Logic Diagrams (LD) – Graphical language;
- Function Block Diagrams (FBD) – Graphical Language;
- Sequential Function Chart (SFC) – Graphical Language;
- Instruction Lists (IL) – Textual Language;
- Structured Text (ST) – Textual Language;

The current paper focuses two specific languages, the LD and IL. The LD because it is the worldwide most used language, and being a graphical language, its learning is simple and quick; adds to the fact that exists a translation from Ladder to logic based on gates AND, OR e NOT, which ahead reveals itself important. The IL because it is a language similar to Assembly, and being textual provides flexibility to be used in any programming tool by simple textual replication.

B. Translation Algorithms

In [10] and [11] are presented proven methodologies for faithful translations from PN to IEC 61131-3 languages. With these methods it is possible to translate and implement the automation functions on an automation standard language, programmable in logic controllers, assuring that they are executed faithfully according to their PN model.

Ladder Logic Diagram

According to the transitions’ firing mechanism, to each place are defined two conditions: Latch Condition – L; Unlatch Condition – U;

Definition 4: The Latch condition is defined by logic OR of sub-conditions, associated with every transition that has the current place in the output function. In the sub-conditions is performed a logic AND between every input places and the logic proposition of the transition.

Definition 5: The Unlatch condition is defined by logic OR of sub-conditions, associated with every transition that has the current place in the input function. In the sub-conditions is performed a logic AND between the logic proposition of the transition and one random input place.

With these definitions it is possible to build a Ladder Diagram that represents a PN Place and its behavior. As shown in the figure 2, the Place Pi is activated when its latching condition is true, and it remains activated while its Unlatching condition is false.

![Fig. 2 – Ladder Logic for Generic Place Pi](image)

In order to complete the implementation are created rungs to evaluate the transitions’ logic proposition and to perform operations assigned to places.

Convert LD to Basic Logic Operations

It is possible to translate simple LD into circuits/equations using only basic logic AND, OR and NOT with the following rules:

- Normally opened contacts connected in series are converted to AND;
- Normally opened contacts connected in parallel are converted to OR;
- To convert the normally closed contact to opened it is added a NOT before the corresponding connection.

An example of this translation is illustrated in figure 3.
The translation from PN to this language is based on two types of instruction blocks: Transition blocks; Place blocks. To each block is assigned a label and the program is built in a specific sequence, as shown in the figure 4.

Existing \( n \) transitions, the label of the first place block will be \( n+1 \).

The initialization performed in the beginning of the program refers to timers and counters.

The transition block is composed by three sub-blocks:

1. **Places’ Validation:** Checks if every input place is marked/active and if every output places are unmarked/inactive, by performing logic AND between the logical value of the input places and negated logical value of the output places. After this operation is added a negated conditional jump to the next block;

2. **Logic Proposition Validation:** Checks the logical value of the logic proposition performing the corresponding logic operations using the transitions’ predicates. After this operation is added a negated conditional jump to the next block;

3. **Transitions’ Firing:** Being checked all the firing conditions, the output places are unmarked and the input places marked, performing resets and sets, to the corresponding places. After this operation is added a jump to the first block place;

The place block is composed by two sub-blocks:

4. **Marking’s Verification:** Checks if the corresponding place is marked. After this operation is added a negated conditional jump to the next block;

1. **Place’s Operations:** Being the place marked, it is performed the corresponding operations.

**C. Programming PN into IEDs**

After the translation of the automation functions, modeled by PN, to the standard automation languages, it is necessary an adaptation to the IEDs’ logic programming language (According to the manufacturers, it is based on the IEC 61131-3).

Therefore, it is necessary to study the details of the IEDs logic languages in order to, indirectly, program the PN models in these devices.

The result of a research performed to IED’s programmable logic from several international manufacturers is presented in the table 1.

**Table 1 – IED’s Programmable Logic**

<table>
<thead>
<tr>
<th>Manufacturer</th>
<th>IED</th>
<th>Configuration Tool</th>
<th>Type of Logic</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABB</td>
<td>REF630</td>
<td>PCM600</td>
<td>Function Blocks &amp; Signal Matrix</td>
</tr>
<tr>
<td>ALSTOM</td>
<td>MiCOM P145</td>
<td>S1 Studio</td>
<td>Basic Logic Gates</td>
</tr>
<tr>
<td>EFACEC</td>
<td>TPU-S420</td>
<td>WinProt 4</td>
<td>Basic Logic Gates</td>
</tr>
<tr>
<td>General Electric</td>
<td>F60</td>
<td>FlexLogic</td>
<td>Logical Equations</td>
</tr>
<tr>
<td>SEL</td>
<td>SEL 351</td>
<td>SELogie</td>
<td>Logical Equations</td>
</tr>
<tr>
<td>SIEMENS</td>
<td>SIPROTEC 4 7SJ64</td>
<td>DIGSI 4</td>
<td>CFC &amp; Signal Matrix</td>
</tr>
</tbody>
</table>

From this table it can be concluded that there are different types of programmable logic, although none is compliant with the IEC 61131-3 standard, but their actual functioning is similar to circuits based on simple logic, so it can be used the algorithms that translate the PN to LD and then convert the LD to Basic Logic Equations or Circuits. For each pair, place and transition, it is required at least the use of one OR operation and three AND operations.

In order to prove the good behavior of the translation, it was programmed a simple PN (two places and two transitions) in
the Siemens’ Siprotec and in the Efacec’ TPU, according to each individual type of logic. For both IEDs the execution of the PN model was proven correct, however, it consumed a great proportion of the IEDs logic resources, for example the TPU has 104 programmable gates, and the programmed PN consumed 8 of those gates, about 8%.

The conclusion of the IEDs’ Logic application in the implementation of PN is that it isn’t a proper solution, since it has insufficient memory capacities and it would require an adaptation the IEC 61131-3 languages, preventing the use of the already existing automatic translation tools.

IV. SYSTEM IMPLEMENTATION

A. System Architecture

The IEDs’ logic reveals limitations that prevent the implementation of the automation functions, so it is necessary the research and development of an alternative method that accomplish the implementation, with the possibility to use directly the code resulting from the translation from the PN to the IEC 61131-3 language.

The solution presented in this paper, aims to the implementation of the automation functions in one dedicated logical controller, programmed directly by the IEC 61131-3, centralizing the digital processing, while uses the IED infrastructure to the predicate generation and the redirection of CB operational commands.

The development of this methodology uses the Instruction Lists as the primary language due to its capacity and flexibility to be used in any kind of logical controller (compliant with IEC 61131-3) by simple code replication, although the main reason is the existence of software translation tools that can perform the translation automatically (SIPN Editor).

Because the execution of the automation functions requires permanent knowledge of the data stored inside the IEDs and the possibility to send commands to CB devices, it is used the IEC 61850, specifically, GOOSE messages [12], that through an Ethernet Local Area Network (LAN) allow the necessary communication between the logic controller and the Substation IEDs.

Because both IEDs already exist, it is not necessary to spend time and cost on their complete reconfiguration. However, they are tied to proprietary software and equipment, making it difficult to centralize the control of the logical devices.

The information transmitted by GOOSE messages are organized in Data-sets, which are a collection of data, ordered and organized in one single entity that allows for a direct monitoring of its values, saving time and bandwidth.

The transmission of GOOSE messages is based on a multicast architecture using a Publisher/Subscriber model, this way the sent messages don’t have a specific destination, they are simply sent to the network, then every device connected to the same network, reads the messages and according to their configuration they decide if the message is processed or not. To perform the transmission of GOOSE messages it is used an Ethernet LAN. The part 5 of the standard defines specific real-time requirements that the GOOSE messages need to fulfill.

C. IEDs Application

Through GOOSE messages, the logic controller receives variables, which it uses by performing logical associations, creating composed variables (or macro-variables) that are later used in the execution of the automation functions. According to this methodology, the controller needs additional processing power to perform these variables compositions, not being
complete dedicated to the execution of the PN models.

In a Distribution Substation there can be up to 20 MV feeders [9], which, together with the remaining equipment, represents a considerable number of IEDs that the controller needs to process and establish communication with. The same can be applied to the number of GOOSE messages that grows proportionally to the IEDs present in the Substation.

In order to optimize the communication system and reduce the processing load of the controller, it is used the IEDs’ logic in the composition of variables required by the automation functions, distributing part of the processing load to the IEDs.

This application of the IEDs logic has several advantages:
- **Modularity**: In the changes at the Substation panel level, the configuration of the controller and GOOSE messages remains the same, only the IEDs logic is modified;
- **Reduction in the number of GOOSE messages**: Being transmitted the composed variables; the number of GOOSE messages required is reduced;
- **Reduction in the message transmitting frequency**: Due to the variables composition, the change of an individual variable doesn’t imply the change of the macro-variable value. Since the GOOSE messages are transmitted at higher frequency when a value of the corresponding Data-set is modified, the frequency of GOOSE publishing is reduced;
- **Reduction of the controller processing load**: Since the variable composition is distributed by the IEDs, the level of processing load assigned to the controller is reduced.

With this additional application, the IEDs, besides the execution of their usual tasks, are incorporated in the automation system by collecting and creating composed variables that are sent to the logical controller, which by its turn, uses directly these variables in the execution of the automation functions.

According to previous presentations, the requirements that the IED needs to fulfill to be applied in the proposed SAS are:
- Performing Protection Functions;
- Data acquisition from the monitored equipment;
- Performing logic operations on the acquired data;
- GOOSE publishing and Subscribing capabilities;
- Allow configuration by the SCL language.

**D. Logical Controller**

It is necessary to use logical controller because the IEDs’ programmable logic has severe limitations. An autonomous logical controller overcome these limitations, because being an automation device, it has enough memory capacity and it is programmed in the IEC 61131-3, which allows the direct use of the code resulting from the translation of the PN that model the automation functions.

Therefore, a PLC could be used as the logical controller device, however to perform the communications between the IEDs and the PLC it is necessary the support for GOOSE messages, so the requirements that the logical controller needs to fulfill are:
- Enough memory capacity to support the complete set of automation functions;
- IEC 61131-3 Support;
- Ethernet Interface;
- GOOSE support, from the IEC 61850.

The IEC 61850 standard is specifically to Substation, so the PLCs manufacturers have low interest in the creation of the corresponding support. The logical controllers with GOOSE support are associated with IEDs manufacturers and are expensive.

These facts suggest an alternative method to the PLC device - the use of a computer to work as the logical controller through automation software (softPLC).

The softPLC is a software that uses the computers’ microprocessor to execute automation programs written in the IEC 61131-3 languages. It supports several communication protocols; it has HMI modules, and can be executed at the Windows kernel level, as a real-time process.

With the processing power of today’s computers, the performance of softPLC is equivalent or superior compared to the traditional PLC, with the advantage that the computer can also be used, besides the automation, to perform other applications, and can establish an internet connection opening the possibility to the execution of automatic tools for data mining and analysis. The softPLC used in the development of this methodology was the CoDeSys from 3S-Software.

Regarding to the GOOSE communications, there are applications, for Windows environments, that allows the publishing and subscription of GOOSE messages, using the already existing computer’s Ethernet Network card, discarding the need for specific communication hardware. The used software is called AX-S4 MMS and it was developed by SISCO with the objective of providing real-time access to the IED’s data, through IEC 61850. For the configuration of GOOSE publishing and subscription it is used the SCD file from the corresponding substation. To interact with other applications the software is provided with an OPC server.

To perform the interface between the two software applications (automation software and GOOSE controlling software) it is used the OPC standard. This protocol was developed to allow standard interconnection between applications in Windows environments. Therefore, a third application is required to work as a bridge between the controller’s OPC server and the OPC server of the GOOSE controlling application, the used in this application is called LinkMaster from Kepware.

Due to the requirements of the electromagnetic compatibility, it is necessary for the Substation computer to fulfill the IEC 61850-3 standard.

The centralized implementation is similar to the “Autómato de Subestações” referred in [6], but instead of connecting the controller through hardware to equipment and non-digital relay, it is used GOOSE messages in an Ethernet LAN.

The resulting communication system applied in the centralized implementation methodology is illustrated in the figure 8.
V. TEST ENVIRONMENT

A. Environment Architecture

In order to prove the correct execution of the system and automation function coordination, the projected SAS was implemented in a simulated partial Substation, composed by the primary equipment: one Power Transformer Panel; two Feeder MV lines. The equipment required to perform the testing was:

- 1x IED Siemens – 7SJ542, Power Transformer Panel;
- 2x IED Siemens – 7SJ645, Feeder MV line;
- 1x Computer, com processador Centrinio Duo T2400 1.83 GHz, 2,5G GB RAM, Ethernet network card 100 Mbps e Operation System Windows XP;
- 1x Switch Ethernet 100 Mbps;
- 1x Omicron testing device.

The IEDs are interconnected, through the Switch in an Ethernet LAN, with the computer, to the exchange of information and operational commands. The testing device is connected through hardwire to the IEDs, generating programmable analog signals of current (to the Feeder IED) and voltage (to the Power Transformer IED), that simulate the CT and VT reading during incidents.

B. System Specifications & Performed Tests

In this testing system there weren’t connections to real CBs, so it was created logic circuits that simulate the CB behavior, and presented on the IED’s display its status. To simplify the interpretation of the events the IED’s LEDs are used to signal the receiving the operational commands and even the start of protection functions.

The automation programs are organized in the SoftPLC by POU (Program Organization Unit) files, each one with about 1900 instruction lines, 56 variables to specific conditions, timers, counters and operation modes, and 88 auxiliary variables to the faithful execution of the Petri Net models.

Logic controller, for each Feeder IED, subscribes to 10 GOOSE messages and publishes to each of these devices 2 GOOSE, from the Power Transformer, it subscribes to 5 messages, with total of 25 GOOSE messages subscribed and 4 published.

The automation programs are executed in averages of 19μs. The communication time between the softPLC and the OPC Bridge is in average 60ms and between the OPC Bridge and the IEDs is in average 8.8ms.

The simulation of incidents is performed by the testing device, in which can be configured as a set of sequential states, the voltage and current amplitude, frequency and duration for each state.

All the individual and coordinated tests of the automation function were performed and completed with success proving the correct execution of the automation functions with real time performances. Therefore the methodology presented in this paper, regarding to the standardized implementation of the automation functions modeled by Petri Nets is validated.

VI. CONCLUSION

This paper presents a successful development of a standardized methodology for the implementation of automation functions, modeled by Petri Nets, in the modern SAS distributed architectures.

It is used the most recent technology in the market, specifically: IEDs with IEC 61850 support; logical controllers, as softPLC; Software applications to publish and subscribe GOOSE messages in an Ethernet LAN. The integrated application of these technologies in one system represents a pioneer project in the Substation Automation Sector.

The architecture of the system is based in the application of the international standards IEC 61131-3 and IEC 61850, regarding to Automation Languages, and Communication Networks and Systems in Substations, which allow interoperability at the automation level, where the same programs can be executed by any logical controllers, and at the communication level, being possible to configure IEDs form any manufacturer with the same SLC file, discarding the need of protocols conversion.

The reuse of programs and configurations suggested by the interoperability has impact in the SAS implementation cost level, making the engineering component cheaper and allowing a faster configuration of the Substation.
Other aspect referring to the introduction of interoperability in the SAS is the possibility to split the engineering from the manufacturing.

Being the automation functions executed in the logical controller, it is possible to modify the programs in a simple, quick and robust way, without changing the configuration of the IEDs, allowing for modifications in the parameters, system maintenance, or its improvement, with the possibility of introducing new functions.

The proposed implementation methodology is presented according to a specific combination of software and hardware, whose objective is the concept demonstration, it is not required its imperative use to perform a standardized implementation with success, there are other solutions, for example, other logical controllers or applications for the GOOSE messages.

REFERENCES