Programmable Logic for IEC 61850 Logical Nodes by means of IEC 61499

ChenWei Yang[1], Valeriy Vyatkin[1], Nirmal-Kumar C Nair[1], Julien Chouinard[2]
[1]Department of Electrical and Computer Engineering, University of Auckland, Auckland, New Zealand
cyan083@aucklanduni.ac.nz, v.vyatkin@auckland.ac.nz, n.nair@auckland.ac.nz,
[2]ISaGRAF, Brossard, Canada, jchouinard@isagraf.com

***Abstract*— IEC 61850 is a recent standard in substation automation system (SAS) aiming mainly at interoperability. Interoperability is achieved by standardizing communications and data between intelligent electronic devices (IEDs). Data in intelligent electronic devices are modelled in IEC 61850 using an object orientated approach as logical nodes. Although the IEC 61850 standardizes the data and communication in substation design, the implementation of control is intentionally left out of the standard. IEC 61499 function blocks has been proposed to fill the logic void of the IEC 61850 standard. In this paper, intelligent logical node (iLN) architecture is used, which combines both description and control within IEC 61850 logical nodes. An IEC 61850 application presented at the IEC TC57WG10 has been modelled using the iLN architecture with the addition of editable logics within the iLN architecture by means of IEC 61499 function blocks implemented in the ISaGRAF development platform.**

Keywords - IEC 61850, IEC 61499, Function Blocks, Logical Nodes, Substation Automation System, intelligent logical node, editable logic

I. INTRODUCTION

In 2007, the total energy consumption according to the United States Energy Information Administration in the US was over 27 trillion kilowatt hours. The main source of energy was generated by non-renewable fossil fuels while renewable energy only accounted for 6.8% of the total energy consumed. The underutilization of renewable energy is a concern as the heavy dependency on fossil fuel could be problematic in the future if the cost of fuel becomes less affordable as it is in the present day. The current infrastructure is not able to maximize the benefits of renewable energy due to the lack of storage and control of the wide variety of different energy sources [1]. The Smart Grid technology incorporates digital communication into the existing power grid which provides real time information for control, and the addition of storage nodes along the network as part of the smart grid architecture could fill the hole that is lacking in the current infrastructure to greater utilize renewable energy sources such as solar, wind and hydro. The real time information provided by the smart grid can empower both the utilities and the consumers to make smarter energy choices such as usage and better management of the flow and the efficiency of power system operations in the distribution grid. A smart grid, in essence, is an Energy Internet which is a move away from the legacy centralized generation and client model to a new decentralized paradigm in energy delivery and management, where the consumers becomes part of the infrastructure in generation and distribution i.e. DRER (Distributed renewable energy source) at the residential and industrial level complemented by DESDs (Distributed energy storage device) such as battery or hydrogen storage, as both a client and supplier [1].

IEC 61850 [2] is the new standard for communication and data representation in substation automation systems (SAS) design. According to the IEC 61850 model, a substation automation system is made up of Intelligent electronic devices (IED) communicating with each other in a network performing functions such as protection, control or monitoring within substations. The standard was published in 2005 and currently there are over one hundred IEC 61850 commissioned substations worldwide [3]. Advantages of IEC 61850 systems includes interoperability between IEDs, simplifying maintenance and the flexibility and scalability of an IEC 61850 system provides a platform for a future proof system design.

Although IEC 61850 standardizes the data and communication in substation design, the implementation of control was intentionally left out in the first edition of the standard. During the first years of standard's application it has been found that the lack of logic descriptions limits the capabilities of developers in terms of encapsulation and reuse of components. The IEC technical committee responsible for the standard development (TC57WG10) has recognized this issue and established new work items in the standard's development agenda towards standardizing the logic representation.

The consideration for agility of IEC61850 systems may have also been overlooked. IEC61850 systems are inherently static. That is once an IED has been defined, the peer systems gather the information in the form of an xml file called CID and SCL file and the system is fixed upon loading these files in the master stations. IEC61499 provides for agility and would enable online reconfiguration of systems. The introduction of IEC61499 will bring substantial value to the industry.

In this paper we continue investigating how to bridge this gap using IEC 61499 [4] - an open standard for distributed automation and control. In particular, IEC 61499 is well

appropriate for implementation of complex logic in substation automation systems and can be combined efficiently with IEC 61850 as it is able to capture the hierarchical characteristics of an IEC 61850 data model from the common data class level to the server level. By combining IEC 61850 and IEC 61499, it is possible to create autonomous logical nodes known as iLN [5] which would be capable of collaborative decision making rather than relying on a central source of control.

The focus of this paper is to demonstrate how the distributed iLN architecture can be implemented with the addition of editable logics in an IEC 61850 application using IEC 61499 function blocks implemented in the ISaGRAF development platform. The ISaGRAF development platform is chosen as it provides support for distribution of single application over multiple devices. In addition, ISaGRAF is developing support for IEC 61850 based applications which includes support for the SystemCorp IEC 61850 protocol stack. The ISaGRAF development platform provides support for a large number of controllers including Advantech and WAGO programmable controllers. The proposed method is illustrated on one industrial use case scenario known as "52 Blocking" [6]. Its benefit is the higher degree of logic encapsulation and, as a result, better reuse of automation solutions which can be tighter coupled with portable logical nodes of IEC 61850. The solution can be easier distributed across multiple networking devices using the built-in mechanisms of IEC 61499 tools.

The paper is structured as follows. Section II presents the Intelligent Logical Nodes architecture implemented in IEC 61499. Section III introduces the idea of Editable logic and how it can be implemented. Section IV presents the"52 Blocking" application and the implementation of the"52 Blocking" application in IEC 61499 function blocks. Section V presents the simulation environment for test validation of the "52 Blocking" application.

II. IEC 61499 Function Blocks for Intelligent Logical Nodes

IEC 61850 introduces a hierarchical data model for power distribution systems. The top level of hierarchy is occupied by the server, and the common data class variables reside at the very bottom. An IEC 61850 server within an IED is made up of a collection of logical devices. A server resides in a physical device, but is made up of virtual data objects. Next level down the hierarchy is the logical device which is made up of a collection of logical nodes performing a specific functionality within the IED. Logical nodes are data set templates specified by the IEC 61850 standard which contains all the data attributes needed to describe a component of a substation (e.g. circuit breaker) or a particular function within a substation design (overcurrent detection or fault detection). Data types within logical nodes are called common data classes and are found at the bottom level of the data hierarchy. The common data class is the basic building block of an IEC 61850 data model and is made up of several data attributes and services akin to the object orientated modelling principle.

The open standard of IEC 61499 is purported for the development of distributed control and automation. In power distribution automation, IEC 61499 is capable of modelling and implementing substations automation in a distributed way where multiple IEDs collaborate communicating horizontally with each other to achieve functionalities such as interlocking or circuit breaker failure protection within large substations. In addition, IEC 61499 is capable of capturing the hierarchal characteristics of IEC 61850 data model from the common data class level to the server level.

The iLN architecture proposed in [6] represents an intelligent logical node as an IEC 61499 function blocks which contains a database function block and an intelligent function block. The database function block contains all the common data class variables that are specified by the logical node definition. The intelligence function block governs the functionality of the logical node. IEC 61499 implemented systems are event-driven and logical connections in an IEC 61850 implemented system are through event signals with associated data connections. Communication between logical nodes within an IED is event triggered which also facilitates data exchanges between logical nodes.

III. Logic in IEC 61850 Logical Nodes

A. Fixed logic and Editable logic

A logical node in the IEC 61850 standard can be seen as a producer and consumer of data. In the iLN architecture, an intelligent logical node will take inputs, then processed by the logic within the logical node, and provide an output based on the implemented logic. There are two layers of logic found within an IEC 61850 implemented logical node. The first layer specifies the semantics of the logical nodes as defined by the IEC 61850 standard. The second layer contains the logic which governs the functionality of the logical node. In the IEC TC57WG10 meeting held in Toronto in June 2010, it was agreed upon by the IEC 61850 committee that the logic part of an IEC 61850 logical node should be made up of two types of logic. They are fixed logic and editable logic as shown in Figure 1. Fixed logic refers to vendor implemented logics which are IED specific and designed for the specification of the substation system. Fixed logic could be protection logic which is implemented by IED vendors. It is not desirable for the end user to modify vendor implemented logic in order to ensure the protection logic works as intended by the IED designer. Fixed logic may not be flexible in terms of reuse in substation designs where the core functionality of the IED is to be kept, however vendor designed IEDs does provide configurability by allowing the end user to modify the settings or controls of logical nodes.

Editable logic was introduced in the "52 Blocking" application [6] to add the flexibility that is lacking in logical nodes where only fixed logic is implemented. In situations where modification to the vendor implemented logic is desirable, editable logic can be used to allow the end user to modify a part of the logic to accommodate substation design where the IEDs are to be reused while the core functionality of the IED is preserved. From the IED designer perspective, it is desirable to implement the core component of the IED logic as

fixed logic, while leaving a small portion of the logic as editable logic to have the flexibility of making small modifications to the IEDs while retaining the core functionality of the IED for reuse instead of complete redesign of the IEDs.

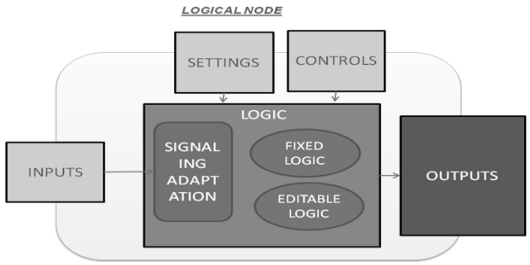

Figure 1. Structure of an IEC 61850 logical node.

B. Implementation of editable logic

Editable logic within logical nodes is implemented through logical equations where Boolean variables are the inputs to the logical operations and logical operators such AND, OR and NOT are used to compute a Boolean result. To implement editable logic within logical nodes, a new common data class is introduced called "Logic Setting" as shown in Figure 2. Each logical equation within an editable logic implementation is represented by the logic setting common data class variable. The setVal data attribute of the logic setting variable contains the logic equation definition for each output variable that is computed by the logical equation.

Common data class specifications for logic setting

Logic Setting

GE class					
Attribute Name	**Attribute Type**	**FC**	**TrgOp**	**Value / Value Range**	**M/O/C**
DataName	Inherited from Data Class (see IEC 61850-7-2)				
DataAttribute					
setting					
setVal	VISIBLE STRING255	SP	dchg	Logic in IEC 61131 literal	M
setVal	VISIBLE STRING255	SG,SE		Logic in IEC 61131 literal	AC_SG_M
configuration, description and extension					
d	VISIBLE STRING255	DC		Description	M
cdcNs	VISIBLE STRING255	EX			AC_DLNDA_M
cdcName	VISIBLE STRING255	EX			AC_DLNDA_M
dataNs	VISIBLE STRING255	EX			AC_DLN_M
Services					
GetDataValues, SetDataValues, GetDataDefinition					

Figure 2. Logic Setting Common Data Class [6]

The data attributes that are referenced for the computation of the logical equations are inputs to the logical node where the editable logic resides. Each of the input data attributes for the logical equation computation is referenced by the InRef attribute of the common data class ORG. Each referenced data attributes are then decomposed to Boolean values through the concept of expected values. Once the referenced variables has been converted to Boolean values, the logic equations then processes the Boolean values and compute the output as specified by the logic equations.

C. Expected Values Algorithm

The concept of expected values is introduced in the "52 Blocking" application [6] to demonstrate how a multileveled common data class data attributes within logical nodes can be decomposed to Boolean variable which can then be applied in the logic equations. Each specific value *v* of the attribute A corresponds to a Boolean variable $B_v \equiv (A=v)$ i.e. whenever the data variable is equal to the specified value, the corresponding Boolean variable is set to *true* while any other attribute values return *false*.

An example to illustrate the idea of expected value is the Pos attribute found in the circuit breaker logical node which tracks the positional value of the circuit breaker. A circuit breaker could have multiple states and a different Pos.stVal value depending on one of its 4 positional states: {*on*, *intermediate-state*, *off*, *bad-state*}, and numbers from 1 to 4 could be used to represent each state respectively. If a Boolean variable is used to represent the value of the circuit breaker when it is on, whenever the position value of the circuit breaker becomes XCBR.Pos[ST][1], the variable will return *true*, while any other position value of the circuit breaker will return *false*. In the implementation of editable logics, all the relevant data attributes in the computation of the editable logics are reduced to Boolean ones in the same manner utilizing the expected values technique.

IV. 52 BLOCKING APPLICATION

"52 Blocking" [6] is the application presented by J. Moreno at the TC57WG10 meeting in Toronto to illustrate the implementation of editable logic within logical nodes. "52 Blocking" describes the safety operation of a circuit breaker by calculating the enable open (EnaOpn) and the enable close (EnaCls) attribute value of the CILO logical node. The EnaOpn and the EnaCls attribute values control the opening and closing condition of the circuit breaker. The variables EnaOpn and EnaCls are part of the editable logic within the "52 Blocking" application and it is implemented in the CILO logical node.

A. Logical Nodes in 52 Blocking

The power system that is designed for the "52 Blocking" application is shown in the single line diagram in Figure 3. The power system contains two busbars which are connected to the circuit breaker labeled as 52. The busbar B is active when the isolator switch 89B and 89AT is closed while the isolator switch 89A is opened. The busbar A is active when the 89B and 89AT isolator switches are opened while 89A is closed. Although there are two busbars in the system, only one busbar is active at an instance while the second busbar acts as the backup to the system. The power system is modelled by four IEC 61850 logical nodes. Logical node XSWI is used to describe the isolator switches 89A, 89AT and 89B. Logical Node XCBR is used to describe the circuit breaker 52 while the CILO logical node governs the interlocking conditions of the circuit breaker. External I/Os into the system are handled by the GGIO logical node.

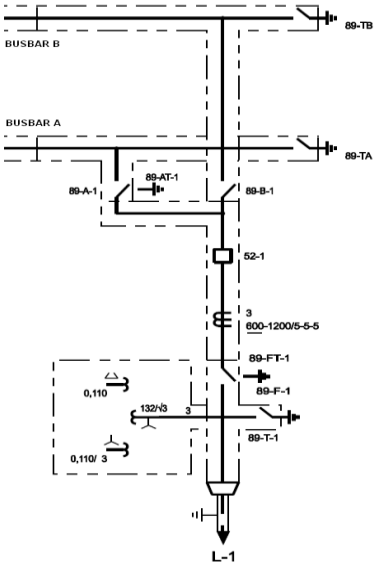

Figure 3. Single Line Diagram of 52 blocking.

B. 52 Blocking Use Cases

There are three use cases presented in the "52 Blocking" application. Each use case is designed to test the opening or the closing algorithm for the safety operating of the circuit breaker under different operating conditions.

The goal of the 3 use cases are as follows:

1. To automatically block the progress of an opening command when the circuit breaker is already opened or the closing command and the circuit breaker is already closed.
2. To automatically block the progress of an opening or closing command when the isolator switches are in the intermediate-state, i.e. the isolator switches are moving.
3. To automatically block the progress of a closing command when the busbar at which the line is connected has a low level of SF6 gas.

Use case 1 analyzes the behaviour of the function 'blocking of the opening of the breaker' when it is already open. In this use case, when the circuit breaker is opened, any additional opening command the circuit breaker receives should be blocked and vice versa for closing commands when the circuit breaker is closed.

Use case 2 analyses the behaviour of the function 'blocking of the closing of the breaker' when at least one of the isolators (89A or 89B) is in the intermediate-state, i.e. from opened to closed or vice versa. In this use case, opening or closing commands issued to the circuit breaker should be blocked while either the 89A or the 89B isolator switch is in the intermediate-state of opening or closing.

Use case 3 analyses the behaviour of the function 'blocking the closing of the breaker' when the busbar at which the L-1 is connected to has a low level of SF6 gas. In this use case, when either busbar A or busbar B trips as a result of a low level of SF6 gas, all the bays connected to the tripped busbar is blocked and the circuit breaker is opened. While the busbar is experiencing a low level of SF6, closing commands issued to the breaker will be blocked until the low level SF6 trip is cleared.

C. Enable Opening and Enable Closing Logic Equations

The blocking of the opening or the closing command to the circuit breaker is facilitated by the enable opening (EnaOpn) and enable closing (EnaCls) data attributes in the CILO logical node. There are two logical equations in the "52 Blocking" application. One logic equation computes the data attribute EnaOpn and the other logic equation computes the data attribute EnaCls. Figure 4 shows the two logic equations implemented in IEC 61131 function block diagrams. The inputs into the logic equations are the resultant Booleans from the expected values algorithm. The logic equations recalculate the EnaOpn and the EnaCls data attribute when there is a change in the input variables.

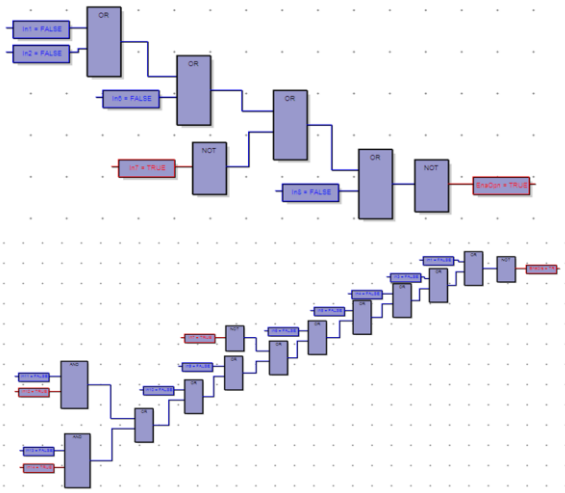

Figure 4. Enable Open (Top), Enable Close (Bottom) logic equations.

D. Implementation of editable logic in iLN

The structure of the logical node implemented in ISaGRAF is based on the iLN architecture which consists of an intelligence function block and a database function block. To add the editable logic to the iLN architecture, editable logic can be implemented as an additional function block next to the intelligence and the database function block within the iLN architecture.

The iLN composite function block which makes up the CILO intelligent logical node is shown in Figure 5. The CILO intelligent logical node contains three main function blocks. The intelligence function block and the database function blocks are as prescribed by the iLN architecture. In addition to the intelligence and the database function block, the third function block in the iLN composite function block is the editable logic function block.

The editable logic function block computes the value of the EnaOpn and the EnaCls attributes each time a change to the system is made. The editable logic function block handles the Boolean data conversion via the expected value algorithm.

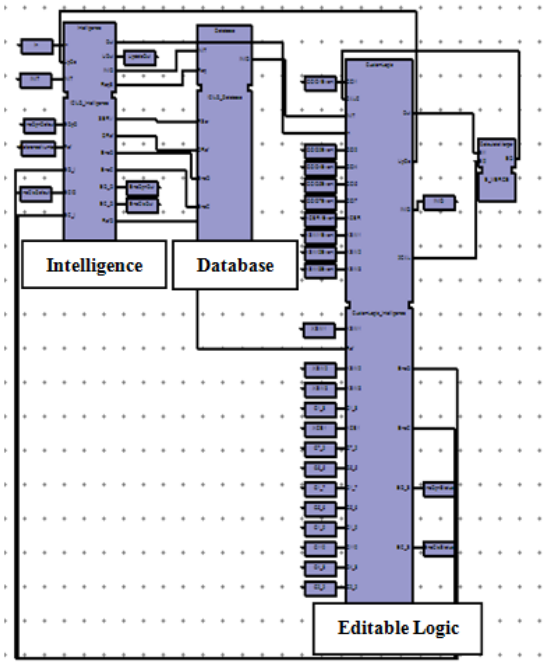

Figure 5. Anatomy of a CILO iLN with editable logic.

The updated EnaOpn and EnaCls attribute value is then passed to the intelligence and the database function blocks of the CILO logical node. As an event driven based system, the editable logic function block will only recalculate the EnaOpn and EnaCls attribute from the logic equation when there is a change in the input variables. Therefore, the editable logic function block does not poll the source logical nodes at regular interval to check whether there is a change in the attribute variable. A change to the data attribute at the source logical node will propagate an output event to the editable logic function block and the updated attribute value will be passed to the editable logic in the CILO function block as an input data for the logic equations.

E. Distributed Devices of 52 blocking

One of the features of an IEC 61499 function block implemented IEC 61850 systems is that the system can be distributed. In the "52 Blocking" application, the function block network is distributed over four devices. Figure 7illustrates how the IEC 61499 logical nodes can be distributed over several devices. The GGIO iLNs are distributed in device 1, the XSWI iLNs are distributed in device 2 and the XCBR and the CILO iLNs are distributed in device 3. The 4th device contains the publisher and subscriber function blocks which are used for communication in the co-simulation environment. With the system being distributed, each intelligent logical node relies on the internal intelligence within to co-ordinate the exchange of data in the distributed system.

F. Data Flow in an iLN network

The iLNs implementing FBs are connected by event and data flow arcs. The behaviour of the system is an event driven system. Figure 6 is a snapshot of the "52 Blocking" application implemented in IEC 61499. Each iLN in the system is connected to the editable logic function block in the CILO iLN with an event signal and their associated data attribute. Whenever there is a change in the data attributes which is part of the logic equations, an output event will be generated out of the source logical node along with the updated value to the CILO iLN. Figure 6 illustrates a sequence of events and data flow when there is a change in the position value of isolator switch. When an isolator switch changes state, for example, from "open" to "close", the XSWI1 logical node, which represents the 89A isolator switch, will generate an output event with the updated data attribute XSWI1.Pos.stVal. When the CILO iLN receives the output event from XSWI1, the updated XSWI1.Pos.stVal is processed by the editable logic function block and the expected value algorithm is applied to the updated value. The resultant Boolean value is then applied to the EnaOpn and EnaCls logical equations which will compute the new EnaOpn.stVal and EnaCls.stVal variables. The CILO function block will then generate an output event

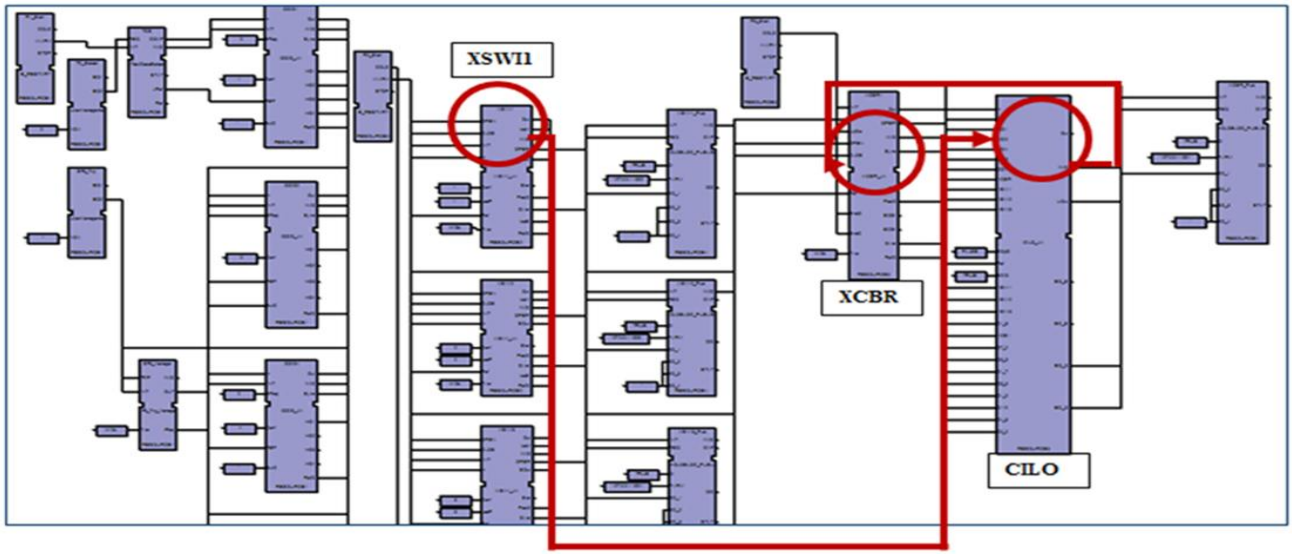

Figure 6. Event flow of 52 blocking implemented in IEC 61499.

with the updated EnaOpn.stVal and the EnaCls.stVal variables to the XCBR circuit breaker iLN. When the XCBR iLN receives the event, it will read the updated EnaOpn and the EnaCls attributes before updating the BlkOpn and BlkCls attributes in the circuit breaker logical node.

V. Simulation Environment

A co-simulation environment was set up similar to the simulation environment in [7] to demonstrate the IEC 61499 implemented "52 Blocking" application. The single line diagram was modelled in Matlab while the IEC 61499 distributed system was implemented in the ISaGRAF development platform. Communication between the two platforms was achieved via Ethernet.

The "52 Blocking" application was also tested on distributed network of IEC 61499 compliant controllers. Hardware platform used for the experiment was composed of 3 controllers (based on ARM7 microprocessor) and communicating via 100MB/s Ethernet as shown in Figure 7. Peripherals of each controller contain 4 LED lights, 2 push buttons and 2 switches. These controllers were represented as devices in ISaGRAF. The GGIO, XSWI, XCBR and CILO intelligent logical nodes were each flashed onto an ISaGRAF controller. The fourth device whose role was to bridge with the Matlab co-simulation environment via publisher and subscriber function blocks was run in ISaGRAF run-time environment on the PC. The communication between the ISaGRAF demonstration kits, the softPLC virtual machine and the Matlab simulation platform was achieved via UDP/IP protocol.

The layout of the ISaGRAF demonstration kits on which the logic nodes I/Os are mapped to is shown in Figure 7. The first device contains the mode selection of the use cases. The combination of the two switches selects the use case. The second device contains the 3 isolator switches (89A, 89B and 89AT) intelligent logical nodes mapped to the I/Os of the demonstration kit controller. Three LED lights are used to represent the status of the three isolator switches. When the LED light is on, it indicates the isolator switch is closed. When the LED light is off, it indicates the isolator switch is opened. The two buttons are used to issue opening and closing commands to the isolator switches. The two switches are used to select which of the three isolator switches receives the opening and closing commands from the input button. The third device contains the XCBR and CILO intelligent logical node mapped to another demonstration kit. One of the LED is used to represent the status of the circuit breaker.

When the LED is on, it represents a closed circuit breaker. When the LED is off, it represents an opened circuit breaker. The two other LEDs represent the value of the EnaOpn and the EnaCls variables. When the LED is on, the associated EnaOpn or the EnaCls value is true. When the LED is off, the associated EnaOpn or the EnaCls value is false.

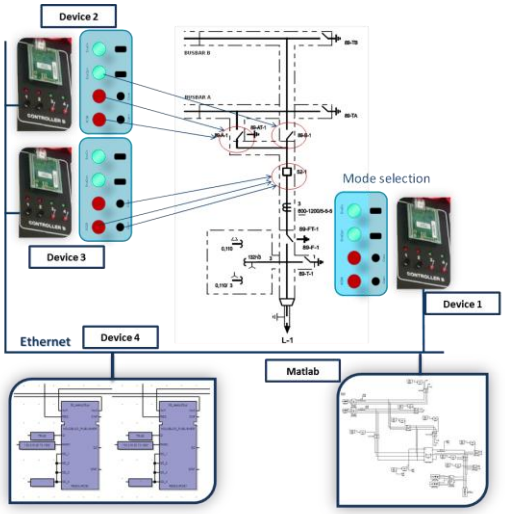

Figure 7. Distributed setup implementing "52 Blocking" scenario.

The three use cases that are specified in the "52 Blocking" applications were tested under the co-simulation environment and the hardPLC of the ISaGRAF demonstration kits. Both were able to provide the expected results as specified by the "52 Blocking" use case goals.

VI. Conclusion

The aim of the "52 Blocking" applications was to show the benefits of adding editable logics to IEDs which only contain vendor implemented logics i.e. fixed logic. The introduction of editable logic alongside fixed logic within IEDs provides flexibility and eliminates the need of proprietary tools or configuration files to configure the IEDs for different applications. With IEC 61499 as the control platform for IEC 61850 distributed systems, the intelligent logical node architecture can be implemented to demonstrate an intelligent smart grid with autonomous agents where decision is left to the discretion of individual logical nodes [8]. The implementation of "52 Blocking" in IEC 61499 demonstrates the ease of implementing editable logic within the iLN architecture.

VII. Acknowledgements

This work has been supported, in part, by FREEDM NSF Centre grant Y3.E.C12 and by ISaGFAF c/n 30747.

The authors are grateful to Karlheinz Schwarz for attracting their attention to the "52 blocking" use case.

VIII. References

1. Huang, A.Q., et al., *The Future Renewable Electric Energy Delivery and Management (FREEDM) System: The Energy Internet.* Proceedings of the IEEE, 2011. **99**(1): p. 133-148.
2. International Electrotechnical Commission, *IEC 61850 Communication Networks and Systems in Substations*. 2003: Geneva, Switzerland.
3. Holbach, D.J., et al., *IEDs exchange signals using numerous GOOSE messages*, in *PAC World*. Autumn 2007. p. 50-58.

4. International Electrotechnical Commission, *IEC 61499 Function Blocks*. 2005.
5. Zhabelova, G. and V. Vyatkin, *Intelligent Logical Nodes of IEC 61850 and IEC 61499 for Multi-agent Smart Grid Automation*. IEEE Transactions on Industrial Electronics 2010, under review, first revision 2011.
6. Moreno, J.G., *MODELING OF LOGICS APLICATION USE CASES - FUNCTION: 52 BLOCKING (FOR OPENING AND CLOSING)*. 2010.
7. Vyatkin V., et al., *Towards Intelligent Smart Grid Devices with IEC 61850 Interoperability and IEC 61499 Open Control Architecture*, in *IEEE CONFERENCE ON TRANSMISSION AND DISTRIBUTION*. 2010: New Orleans.
8. N. Higgins, V. Vyatkin, N. Nair and K. Schwarz, Intelligent Decentralised Power Distribution Automation with IEC 61850, IEC 61499 and Holonic Control, IEEE Transactions on Systems, Machine and Cybernetics, Part C, 40(3), 2010