DEGREE PROJECT IN ELECTRICAL ENGINEERING,
SECOND CYCLE, 30 CREDITS
STOCKHOLM, SWEDEN 2016

Distribution On Load Tap Changer Control Using IEC61850 Client /Server Architecture

ANDRIUS MANEIKIS

KTH ROYAL INSTITUTE OF TECHNOLOGY
SCHOOL OF ELECTRICAL ENGINEERING

Abstract. Distributed generation is transforming the power system grid to decentralized system where separate units like wind power generators or solar panel shall coexist and operate in tandem in order to supplement each other and make one extensive system as a whole so called smart grid. It is utmost important to have a control ability over such units not only on a field level but on a system level as well. To be able to communicate with numerous devices and maintain interoperability universal standard is a must. Therefore, one of the core standards relevant to smart grids is IEC 61850 – Power Utility Automation which comes into assistance and tackles aforementioned challenges. This project uses IEC61850 architecture to implement client/server windows applications for on-load tap changer remote control. The proposed solution and designed applications are tested together with a real time simulator where simple power system is modelled to emulate the system response to control signals in a real time. In this way, the implemented applications can be tried and assessed as if performing in real environment. Consequently, a user of the client application is able to remotely control voltage on the power transformer's secondary side and manipulate the switching equipment simulated in the model.

Keywords. IEC61850, OLTC, transformer, tap changer, distribution, electric, power, automation, client, server.

Acknowledgment. I would like to thank my supervisor Yiming Wu for the guidance and support during the whole project. I would also like to express my deepest appreciation to *FMTP* and *NettedAutomation GmbH* for having an opportunity to work with great experts in the field and sharing that knowledge with me. It was a great encouragement and inspiration to seek set goals with guidance beside. Finally, I would like to thank entire staff in the ICS department at the Royal Institute of Technology for providing the ways and means during my entire education in the faculty.

Referat. Distribuerad generation håller på att förändra transmissionsnätet till decentraliserat system där separata enheter som vindkraftverk eller solpanel skall samexistera och fungera tillsammans för att komplettera varandra och att göra ett omfattande system som helhet så kallade smarta elnät. Det är ytterst viktigt att ha en kontroll förmåga över sådana enheter inte bara på ett fältnivå utan även på systemnivå. För att kunna kommunicera med många enheter och bibehålla interoperabiliten som universell standard är ett måste. En av de grundläggande normer som är relevanta för smarta nät är IEC 61850 - Skydd & Automation, som kommer in i bistånd och möter ovan nämnda utmaningar. Detta projekt använder IEC61850-struktur för att implementera klient/server windows applikation för lindningskopplarens fjärrkontroll. Den föreslagna lösningen och utformade applikationer testas tillsammans med en realtidssimulator där enkelt kraftsystem modelleras för att emulera systemets svar på de givna styrsignalerna i realtid. På detta sätt kan de implementerade programmen prövas och bedömas hur de utföras i verklig miljö. Följaktligen kan användare av klientapplikationen fjärrstyra spänningen på transformatorns sekundärsida och manipulera ställverk som simuleras i modellen.

Table of Contents

1 Introduction

1.1 Background

Nowadays smart grid is a major trend that changes paradigm of a whole energy conversion chain and shifts the generation level down to a consumer. The grid topology transforms from radial networks to meshed networks due to distributed generation and power flows change from unidirectional (downstream the lines from centralized generation to consumers) to bidirectional at distribution level. Therefore, simple protection schemes are not applicable any longer and costlier solutions only used at a transmission level shall be implemented in the distribution grid.

In addition, the distributed generation is more difficult to manage in terms of overall system stability where each system's element must operate synchronously at desired frequency and voltage. These parameters change dynamically depending on consumption and production. Thus, ability to communicate and actively control each system unit is a necessity to make the system operate at optimal levels. On-load tap changer (OLTC) transformers are used to maintain desired voltage and they play an important role improving power system stability [1][2].

Another important aspect is interoperability of the devices in complex power systems. Standardization plays a key role to efficiently share information and communicate between devices from different vendors lest the innovation would be stalled by proprietary standards.

IEC 61850 is one of the core standards defined by IEC committee to create a uniform, future-proof basis for the protection, communication and control of substations. Advantages to use this standard, such as simplification of the design, cost reduction, and increase of reliability have been proved by its application in many substations and devices [3].

1.2 Problem definition

Software development becomes a big part of new technology and it requires knowledge in many different domains, where collaboration between software and hardware engineers might open new possibilities, generate new ideas and solutions. Some engineers might not always have a knowledge or expertise how to develop a tool that would fill their needs or make their work more efficient.

For instance, many solutions have been proposed for changing transformer voltage locally [4][5]. However, the demands and expectations have changed from a customer point of view. They want more and better control over devices they maintain and remote control is a desired feature. This project focuses on a remote control and communication solution. Moreover, to test such control solution a physical device or some replica model would be required. Therefore, to be able to develop a control algorithm and test it in the lab environment without a hardware power transformer would be a great asset.

It is important that provided solution would be able to communicate with various devices. To achieve interoperability, IEC61850 standard was chosen because it covers all parts for substation automation from system description to communication protocols. Hence, the developed product will be compatible with other tools and programs based on this standard.

1.3 Related work

Some master thesis related to IEC 61850 standard had been produced which gave inspiration to this project. Good theoretical background covered in [6] where analysis of the standard had been carried out and outlines of the basic IEC61850 server implementation is delivered. Another related paper [7] to this project was written where the same SystemCORP communication stack was used to implement communication between an IED and IEC61850-compliant controller.

1.4 Objectives

To fulfil the project, several key aspects can be distinguished. They are presented in a list below:

- To provide IEC 61850 standard solution for the OLTC (On-load tap changer)
- To implement server and client applications
- Model power system with a tap changer transformer
- Set reference voltage remotely
- Verification using hardware in the loop concept

The project main aim is to provide a solution based on IEC 61850 standard for OLTC. The thought is to have an ability to control the tap changer set point voltage from a remote location and be able to supervise the processes using tools compliant with IEC 61850 standard. To achieve this task OPAL network simulator is used where simple network model and a power transformer tap changer algorithm is implemented. To have a control possibility of a tap changer, server application is implemented which communicates with a client and provides required data for operation. Then, to ensure the solution was implemented correctly and to verify the results hardware in loop concept is to be used.

1.5 Overview of the report

In this report brief introduction is given about voltage control and basic principal of a voltage transformer device is described. Then, the scope of IEC61580 standard is outlined and general theory of the standard is covered. Since the scope of the standard is very extensive only essential principals relevant to the project will be discussed in this report. For deeper clarification a reader is advised to look into reference. Next case study is presented with project setup description, further discussion of the implementation and received results. After that, final discussion is developed and conclusions drawn about entire project. Finally, the reader can find source code and system models of the client and the server in the Appendix.

2 Introduction to voltage control

2.1 Basics of a transformer

Electrical network system comprises of many different levels in terms of voltage. The majority of electric power still comes from conventional sources [8]. The generators produce electricity that has to be transported over hundreds or even thousands kilometres away from its source. According to physics laws power loss is proportional to squared current, and power is a product of current and voltage. Therefore, to maintain same power, voltage can be increased and current decreased, thus lowering the power losses.

A device that changes alternating voltage is called a transformer. It is based on electromagnetic induction principle, where alternating magnetic flux induces electromotive force which is proportional to magnetic flux change rate and turns ratio (see Figure 1).

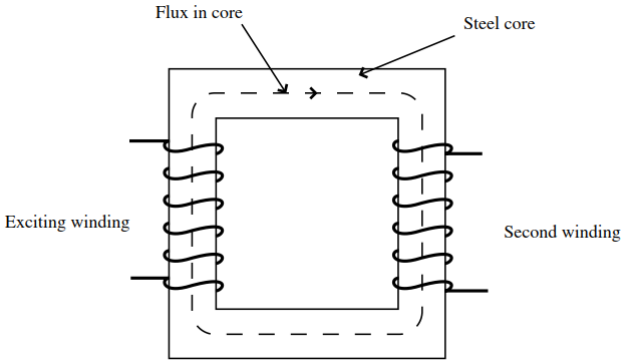

Figure 1 Ideal transformer [9].

Different voltage is induced by applying a secondary winding on a steel core with a different turns ratio.

A mathematical equation can be expressed for an ideal transformer by the following formula:

$$\frac{N_p}{N_s} = \frac{V_p}{V_s} = k = turns\ ratio, \qquad where$$

N_p – number of primary (or exciting) windings;

N_s – number of secondary windings;

V_p – primary voltage;

V_s – secondary voltage.

2.2 OLTC transformer

During an operation on a real transformer it is impossible to change the winding ratio for obvious reasons. Instead, the winding is divided into multiple sections (or taps) and a tap changer switches to the desired section short circuiting the remaining winding. Therefore, by moving the switch up or down the factor k changes, thus changing the induced voltage.

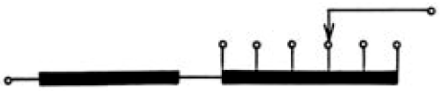

Figure 2 Basic principal of a tap changer

By having ability to send a signal over distance it is possible to control voltage remotely. For that reason, IEC 61850 standard is very beneficial allowing to formalize the control process and making it interoperable with different devices.

3 Overview of IEC61850 standard

IEC 61850 is a standard essentially aiming to provide full infrastructure for substation automation. The core principle of the standard is to make this automation efficient and interoperable.

3.1 Scope of IEC 61850 standard

IEC 61850 is targeted for Power Utility Automation and the general title is Communication networks and systems in substations. It is divided into the following parts [10]:

- IEC 61850-1 Introduction and Overview

  Gives an introduction and overview of the IEC 61850 standard series

- IEC 61850-2 Glossary of terms

  Contains the glossary of specific terms and definitions used in the context of SAS[1] which are standardized in the various parts of the IEC 61850 series.

- IEC 61850-3 General Requirements

  Applies to substation automation systems and more specifically defines the communication between intelligent electronic devices in the substation and the related system requirements.

- IEC 61850-4 System and Project Management

  Describes the requirements of the system and project management process and of special supporting tools for engineering and testing.

- IEC 61850-5 Communication Requirements for Functions and Device Models

  Applies to substation automation systems and standardizes the communication between intelligent electronic devices and the related system requirements. Refers to the communication requirements of the functions being performed in the substation automation system and to device models.

- IEC 61850-6 Configuration Description Language for Communication in Electrical Substations Related to IEDs

  Specifies a file format for describing communication-related IED[2] configurations and IED parameters, communication system configurations, switch yard (function) structures, and the relations between them. The main purpose of this format is to exchange IED capability descriptions, and SA[3] system descriptions between IED engineering tools and the system engineering tool(s) of different manufacturers in a compatible way.

- IEC 61850-7 Basic Communication Structure for Substation and Feeder Equipment
  - IEC 61850-7.1 - Principles and Models

    Provides an overview of the architecture for communication and interactions between substation devices such as protection devices, breakers, transformers, substation hosts, etc.

  - IEC 61850-7.2 - Abstract Communication Service Interface (ACSI)

    ACSI provides description of communication between a client and a remote server, interfaces for data access and retrieval, event distribution, control blocks and logging.

  - IEC 61850-7.3 - Common Data Classes (CDC)

    Specifies common attribute types and common data classes related to substation applications.

  - IEC 61850-7.4 - Compatible logical node classes and data classes

    Specifies the information model of devices and functions generally related to common use regarding applications in systems for power utility automation.

[1] SAS – Substation Automation Systems
[2] IED - Intelligent Electronic Device
[3] SA – Substation Automation

- IEC 61850-8 Specific Communication Service Mapping (SCSM)
    - IEC 61850-8.1 - Mappings to MMS (ISO 9506-1 and ISO 9506-2) and to ISO/IEC 8802-3

      Specifies a method of exchanging time-critical and non-time-critical data through local-area networks by mapping ACSI to MMS
- IEC 61850-9 Specific Communication Service Mapping (SCSM)
    - IEC 61850-9.2 - Sampled Values (SV) over ISO/IEC 8802-3 10 Conformance Testing

      Defines the Specific Communication Service Mapping for the transmission of sampled values according to the abstract specification in IEC 61850-7-2.
- IEC 61850-10 Conformance testing

  Specifies standard techniques for testing of conformance of implementations, as well as specific measurement techniques to be applied when declaring performance parameters.

From part 3 to part 5 the standard identifies general and specific functional requirements for communication in substation automation systems. The core sections of the standard can be found from part 6 to part 9. They will be discussed in more details in the next chapter to provide a reader with basic concept of the standard.

3.2 Basic concepts of system modelling in IEC 61850

3.2.1. The information models of substation automation systems

Well defined information models allow efficient information exchange between different devices. Such information models and the modelling methods are the fundamental parts of the IEC 61850 standard. The standard's approach is to model the common information found in real world devices as depicted in Figure 3 below:

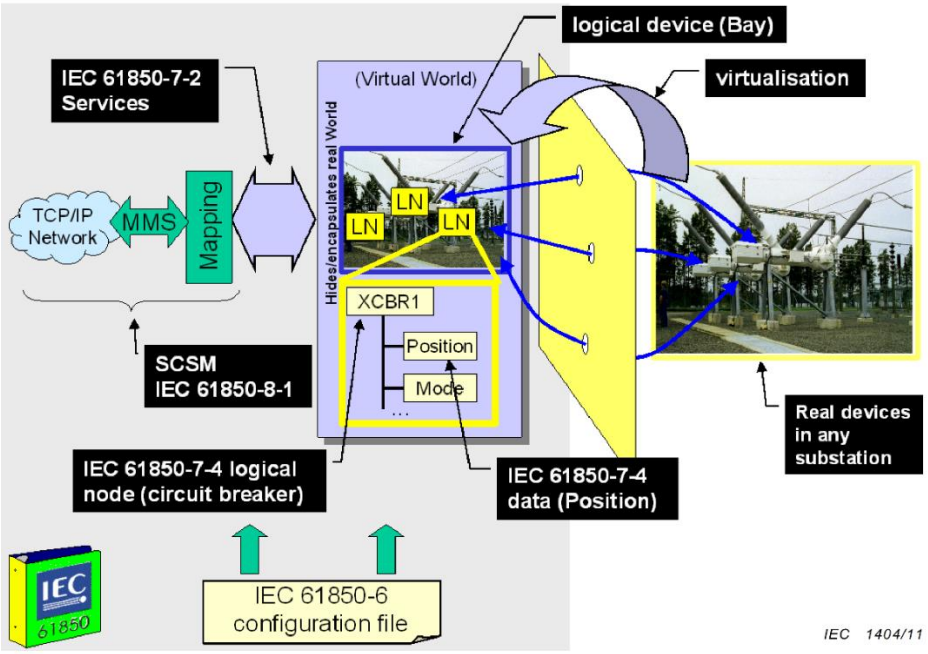

Figure 3 – Conceptual modeling approach of IEC 61850 standard [11].

IEC61850 uses virtualization concept where real devices with many different functions as switching, breaking or protection capabilities share the same attributes in any substation so they can be generalized into abstract model. Then information to be exchanged is independent of specific device and concrete implementation instead is related to the devices of interest for a particular information. In this manner application functions are decomposed into smaller entities to be used for information exchange. Such entities are called logical nodes which contain a list of data with dedicated data attributes. Logical nodes are grouped into logical devices. Hierarchy of the information model is depicted in the Figure 4.

The data to be exchanged has a structure and meaning in the context of systems for power utility automation. The information contained by data and its attributes are communicated by specific communication service mapping (SCSM) using MMS, GOOSE (Generic Object Oriented Substation Event), TCP/IP, and Ethernet and others defined by IEC 61850-8.

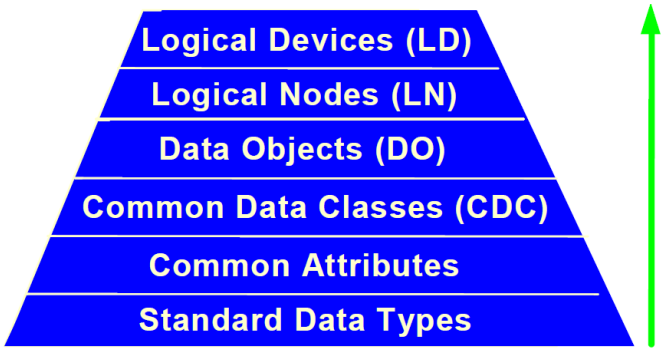

Figure 4 Hierarchy of information model [12].

3.2.1.1 Logical nodes relationship

All logical nodes consist of common logical node information and physical device information independent of the application domain. The former describes logical node such as behavior, nameplate information, operation counters, the latter is related to physical device (LPHD) implementing logical devices and logical nodes. Other information is inherited from Common Logical Node and is specific to an application domain. The relation is depicted in the diagram below:

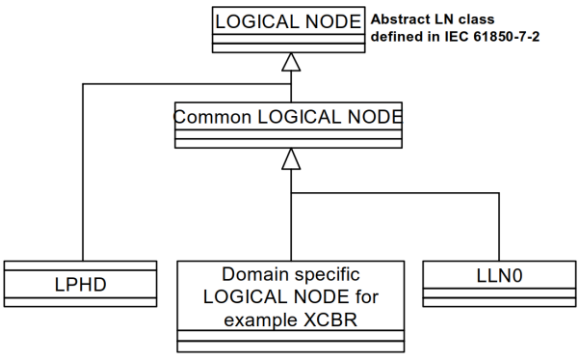

Figure 5 Logical node relationship [13].

3.2.2. Substation Configuration Description Language

The substation model with all devices inside is defined by SCL file. SCL file format is used to describe IED configurations, communication system and capabilities, function structures. The purpose of such file is to provide complete interoperability and data exchange between different devices. It also provides modification capabilities if the system model changes to add additional components or subtract existing ones. Since SCL is based on XML format it allows different system engineering tools to perform these modifications.

SCL format consists of six different file types for specific purposes (see Table 1). SCL file contains the following parts [14]:

- Header
- Substation description
- IED description
- Communication system description
- Data type templates

Table 1 SCL file formats with specific purposes

File extension	Name	Description
.icd	IED capability description	Describes complete capability of IED. The file is intended to be a template for IED type and contains optional communication section and optional substation description.
.ssd	System Specification Description	Contains complete substation automation system functionality. Does not need to include IED descriptions.
.scd	Substation Configuration Description	Contains complete substation automation system functional including IED descriptions.
.cid	Configured IED Description	Contains mandatory communication section for a specific instance of IED.
.iid	Instantiated IED Description	Preconfigured one IED with communication parameters, data templates. Used as an input for system engineering tools.
.sed	System Exchange Description	Provides interface for information exchange and transfers engineering rights between projects.

3.3 Abstract communication service interface (ACSI).

Logical nodes, data, data attributes and service parameters are defined primarily to specify information required to perform a process or an application and for information exchange between different IEDs. Thus, communication systems abstract mainly from application and device point of view and they are closely related. These relations will be discussed in this chapter.

The information to be exchanged is defined by means of services. For instance, to trip the circuit breaker the possible services employed to such task is depicted in the picture below (see Figure 6). All categories of services are described in [15].

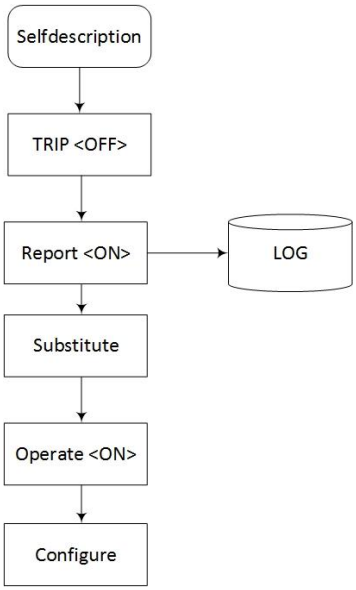

Figure 6 Service example

Initially, description of the device is retrieved and status information exchanged (tripping). Report and logging events are triggered automatically on certain conditions defined in the information model. The substitute service forces a particular data attribute to be changed independent of the process. Operate service reaches the circuit breaker directly, meaning that service operates on the request without any constrains. Depending on state of the machine, more elaborate behavior may be required such as select-before-operate. Finally, online configuration is done to block opening of the circuit breaker.

3.3.1. Mapping to real protocols

Services defined above are called abstract services. They enable the communication between the client and the server. The term abstract means that defined services are required to perform the request. Therefore, the sending and receiving sides behave in identical manner and can understand each other. ACSI only provides semantics of the services. Since abstract services help to achieve interoperability it is crucial to choose an existing, time tested and reliable communication protocols and map to one or more of these protocols. This mapping is called specific communication service mapping (SCSM) and is defined in [16] and [17]. The picture below shows the possible mappings to already widespread protocols in the industry such as MMS, GOOSE, SV.

Most of abstract services defined in ACSI are mapped (defined by IEC 61850-8-1) to Manufacturing Message Specification usually abbreviated as MMS. It is international standard (ISO 9506) for transferring real time process data and control information between devices or applications in the network. MMS provides a set of services (like read/write/report) for peer-to-peer real-time communications over a network and can support the complex naming and service models of IEC 61850. It is based on client/server relation and uses seven layers of OSI model (see Figure 7).

To exchange information for time critical events the abstract services are mapped directly to the data link layer and protocols like GOOSE or SV are used. These protocols rely on publisher/subscriber concept where the source device publishes information and the other device receives transmitted information. The sending device can publish a message to multiple devices in a multicast mode or only to a single device (unicast mode).

GOOSE protocol is mainly used for protection related events. Since it bypasses network layer and it is directly mapped to Ethernet, GOOSE messages are not routable and are only used in horizontal communication in a substation. In order to increase performance and maintain high transmission

speeds, communication confirmation services are not used. Instead, the messages are sent repeatedly to increase probability for successful delivery to the recipient.

SV protocol is used to facilitate high frequency waveform sampling and transmitting in digital communication. The sampling rate is defined by IEC 61850-9-2LE to two distinct sample rates 80 and 256 samples per nominal frequency cycle which translates to 4000 and 12 800 samples per second in 50 Hz system [18] respectively. The former is used common protection functions whereas the latter for high speed functions such as power quality monitoring.

There are other protocols used in IEC 61850 for instance Simple Network Time Protocol (SNTP) which is used for time synchronization over communication networks.

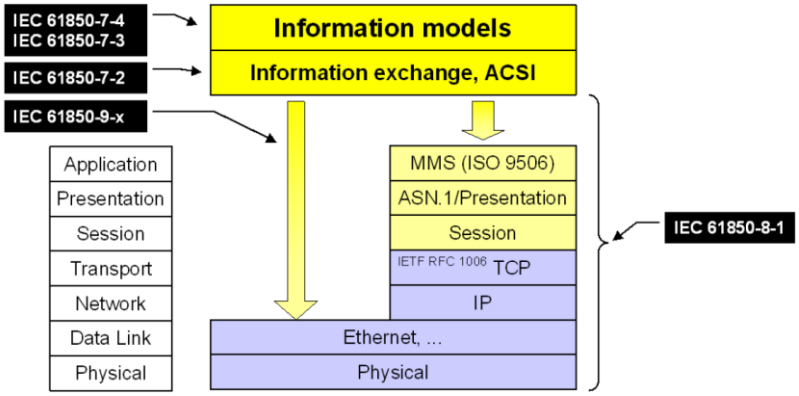

Figure 7 SCSM according to OSI layers [11].

SCSM maps abstract services to concrete protocols that allow communication in the network. These services provide abstract model which enables interoperability between a client and a server. The connection between the client and the server is realized by communication stack (see Figure 9). It links the application layer and application process.

3.3.2. Client/Server relation

Physical device hosts a server which contains data of a described system model and all necessary information defined to be visible and accessible from a communication network. The client provides an interface to access the data in the server:

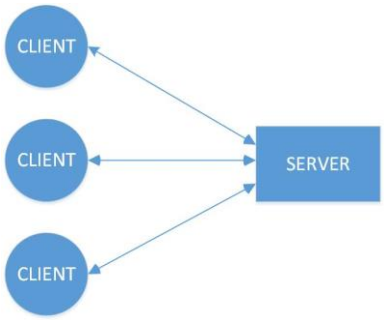

Figure 8 Client/Server roles

The client asks for data or an action from the server and it can only send requests and receive confirmations of services from the server. The server that contains data objects carries out operations on behalf of the client and may issue responses and indications for requested communication services. A communication between a client and a server is realized by the protocol stack which is a piece of software enabling efficient implementation of the communication services (see Figure 9).

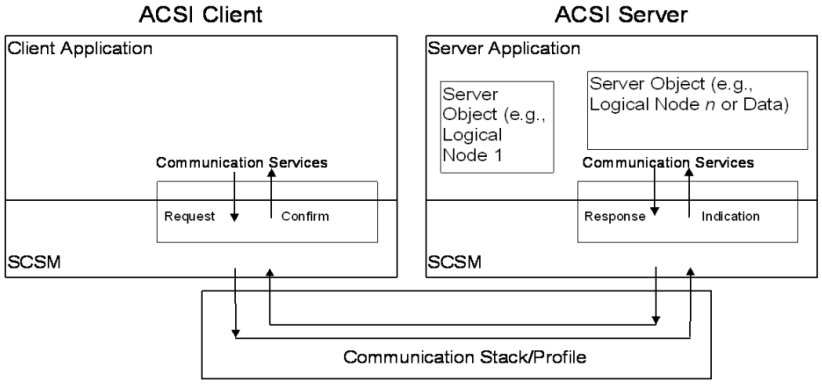

Figure 9 Interaction between client and server [11].

4 Methodology

In section 1.2 problem of the project was defined and section 1.4 goals were set to solve that problem. In order to reach those goals, multiple aspects need to be solved. This chapter explains methods to be used for fulfilling set goals.

4.1 Theory study

First of all, background study of the standard was carried out in order to understand the concept and structure of the standard. Logical nodes required for the model have been examined and selected so it can be represented properly and required data could be held accordingly. In addition, the theory study was conducted about employed tools for building system description and to properly configure network settings. Popular C programming language has been selected for fast and great library variety and the communication stack library, compatible with IEC61850 was chosen. So the implementation instructions have been studied to implement required functions into the applications. For graphical user interface GTK+ toolkit was used which provides effective and rapid application development.

4.2 Model

To be able to apply the theory concepts, an imitation of a system has to be created. Therefore, a model of a transformer with a tap changing capabilities needs to be developed. Furthermore, this tap power transformer cannot work isolated, so the simple power system is connected on both sides of the modelled transformer where generating unit and loads are present. Hence, the actual parameters and relations between voltages and currents can be investigated depending on the load and status of the control settings. For instance, by changing the tap position, the voltage changes on the secondary side accordingly given that the loads are the same. So the developed power system model helps to investigate control mechanisms and later is used in a real time simulator.

4.3 Applications

There are to applications to be created. The purpose of two applications is to have a server and a client where a server application processes necessary information and a client is able to connect to the server and interact in control process. The server application is compatible with a 61850 standard and creates a host so that a client compatible with the standard is able to communicate remotely. Here the client application is able to get a user input of control parameters and deliver it to the server. Consequently, a client user is able to monitor relevant information that is being processed in the server.

4.4 Evaluation of the results

The outcome of the results can be evaluated by running the model in a real time simulator. Integrating the simulator with the server application can give the output in the real time. So performance of the client and the server applications can be assessed. Since the simulator is generating output data in real time according to the model and given control input parameters. The voltage on transformer's secondary side should change depending on the setpoint value. It should also respond to control signal of the circuit breaker. Therefore, the powerflow should change by disconnecting the load. The voltage change speed and extent should be limited by a tap changer since it imitates a mechanical device with limited number of taps.

To test whether the 61850 server was configured correctly and functioning properly, some third party tool could be employed to connect to the server and read the system model. Depending on the tool capabilities, the data objects can be manipulated as well.

5 Case study

In this chapter a case study of an experiment setup will be discussed and main principles outlined for a reader to gain a general understanding of the project. More detailed discussion about the implantation is given in the next chapter.

5.1 Setup of testing platform

Experimenting with real hardware equipment in many cases may be complicated, costly and ineffective. Adjusting different settings might require redesign of the devices which requires additional engineering and financial resources. Therefore, the better solution is to prototype a desired system into mathematical concept and launch in the simulator environment. This approach has its limitations and perfect simulation of real environment is not possible. Thus, some simplifications are introduced by retaining the system fully functioning. As long as a researching engineer is aware of limited power of simulation tool it is reflecting real world dynamics in acceptable level.

Experiment overview is given in the picture below:

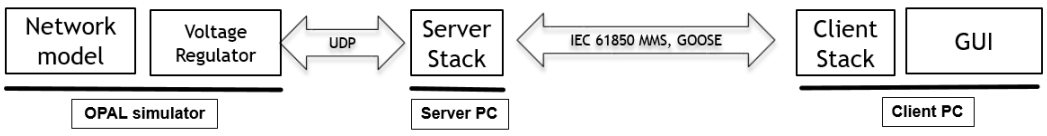

Figure 10 Project experimental setup

To reflect dynamics of the power transformer simple line network model is created and voltage control algorithm implemented in OPAL network simulator. The simulator is communicating with the server where necessary data is being transmitted in between via UDP connection. Server is implemented with a IEC61850 standard. It has a network model described in SCL file and enables connections for clients. The client PC is running on a different machine and is able to send control commands to the server.

5.2 Workflow

The project workflow (see Figure 11 Project workflow) can be divided into four major parts.

- Engineering process starts when real devices, functions, applications are virtualised and converted to abstract model (see Chapter 3.2). Existing elements and functions are described in SCL file. This process can be performed by simple text editor in any operating system. However, many third party tools are available to make the process more efficient and human error free. Final configuration is performed for instantiated IED with proper communication settings.
- Second step is to create a server and client application which is capable of loading SCL model and provide IEC 61850 related services and functions. Software developer does not need to create all possible functions by oneself. Instead, various API libraries can be used, which provide tested IEC 61850 functions, thus saving developing time.
- Next step is to build the system model in the modelling environment which can be compiled and processed by the real time simulator. The modelling is done by combining predefined separate blocks to desired power system. Network settings also should be tailored according to the network specifications in order be able to establish communication link with the server.
- Finally, to test the model and application implementation different scenarios are carried out. The model runs continuously simulating the real time process of the interested power system. Input parameters are given as would be given to the real devices. The simulator processes input parameters and gives output parameters accordingly. Since the focus is on a tap changer behaviour which is a mechanical process and the system model is not extensive the simulation can be run continuously.

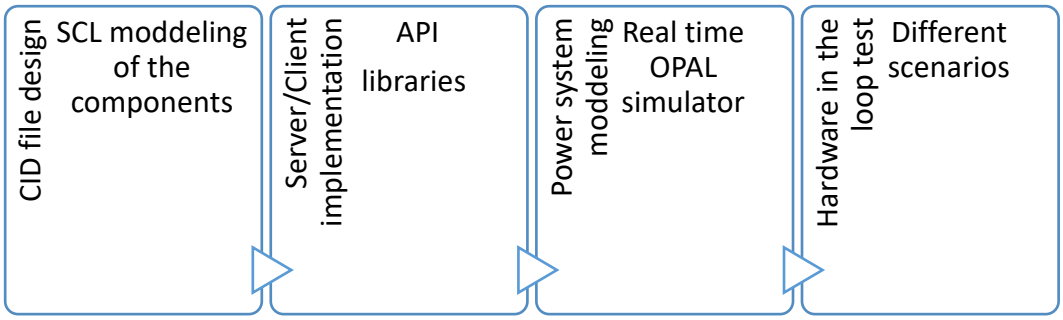

Figure 11 Project workflow

5.2.1. System description in SCL

As it was described in previous section (see Table 1) there are several types of SCL file serving different purposes. To facilitate server application CID file is created where IED capability is described and communication section configured for a particular network. Complete CID files are given in the Appendix sections 9.3 and 9.4 for server and client applications respectively.

IEC 61850 standard provides many predefined data objects grouped in standard logical nodes [13]. They can be used as "bricks" for building custom logical device with desired functionality. In the picture (see Figure 12) below is given a simple power network model used for simulation with logical nodes necessary to accommodate data fields and functions. For instance, to hold a position and to have a control ability of the device XBBR0 logical node is implemented.

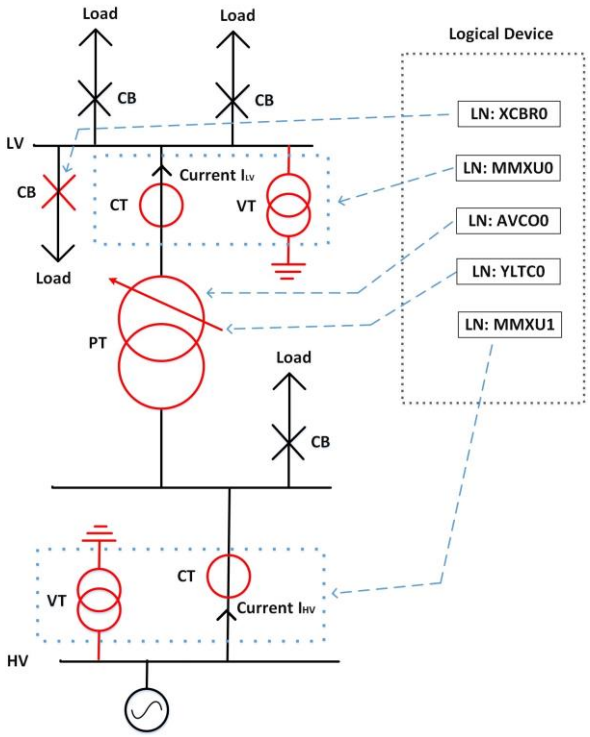

Figure 12 Basic diagram of the system model

Grouping of logical devices is not specified by the IEC 61850 standard and depends on implementation to suit the product functionality the best. Meaning that each logical device may have different combinations of logical nodes. Each logical contains mandatory Data Objects (DO) that is bare minimum required by the standard. There might be DO marked as conditional with note under what conditions it shall be included [13]. Optional DOs could be also included but standard is out of scope to provide the decision and depends on the specifications. For instance, tap changer logical node (YLTC0) consists of the following data attributes given in the table below:

Table 2 YLTC Logical node composition for tap changer

Name of data object	Name of data attribute	Option	Description
Beh		M	Logical node behavior
	stVal		Status
	q		Quality
	t		Timestamp
TapPos		O	Change tap position
	valWTr	M	Tap position
	q		Quality
	t		Timestamp
	ctlModel	M	Control
EndPosR		M	End position raise reached
	stVal		Status
	q		Quality
	t		Timestamp
EndPosL		M	End position lower reached
	stVal		Status
	q		Quality
	t		Timestamp

The particular tap changer logical node (YLTC0) provides mandatory data objects with data attributes holding status values of tap changer higher and lower positions limits. Thus, it allows to receive information when transformer reaches voltage regulation limits. Additional logical node was included to have exact tap position of the power transformer. The distinction for between logical nodes YLTC0 and AVCO0 that former is used for supervision and the latter for control where client can assign the setpoint value to maintain desired voltage.

Other logical nodes (for instance MMXU0) is to accommodate current and voltage measurements converted in RMS values. To be able to control circuit breaker XCBR0 logical node is implemented and control commands can be sent from client.

5.2.2. Server/Client implementation

Server application represents a visible external behavior and multiple clients can access information defined in the server. The server and the client applications are written in C programming language. The client has graphical user interface (GUI) to display information received from server and get input from a user for setpoint value and circuit breaker control.

5.2.2.1 System Corp API DLL

To develop an application a software engineer has to write a code which would realize all desired functions and services. In this way the development process can take a long time and compatibility aspect could be compromised due to vast extent of the aspects to be taken into consideration. Therefore, it is better to use already developed libraries to implement an application. Various APIs are available that provide IEC 61850 features. For this purpose, SystemCORP stack is used which implements IEC 61850 functionalities and allows to integrate them into client/server application [19].

5.2.2.2 Server application

Server application is the information exchange and service interface for the power system events. It interacts with a API DLL library to access provided functions, compliant with IEC61850 standard. API DLL loads the 61850 objects from the CID file to map the user application objects. Using commands from the API, server sends the required data to the physical devices via Ethernet connection over various available protocols in IEC 61850 such as GOOSE, MMS, SV etc. However, in this project communication between the client is limited to GOOSE and MMS messages.

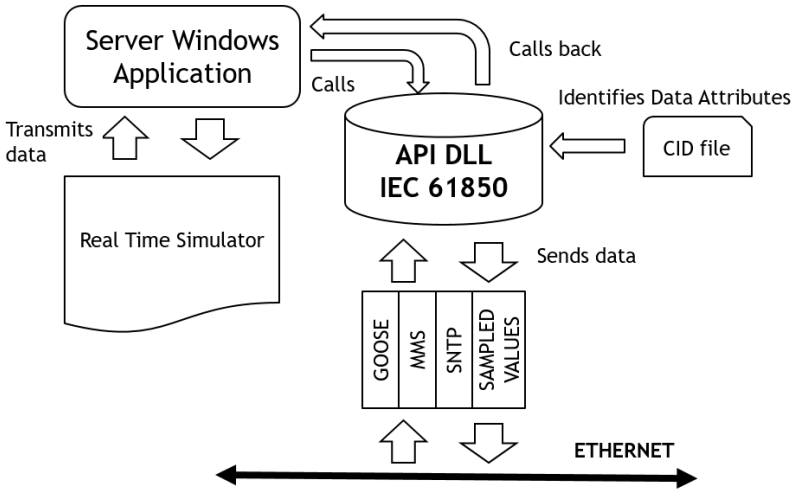

Figure 13 Server implementation overview

Communication between the server and the real time simulator is established via UDP connection. The real time simulator has a function block which controls transmission rate and provides configuration for network settings. The connection is unicast which means that packets are sent directly to a specific host machine with particular IP address and port number. The real time simulator receives data of a setpoint value and operating signal for the circuit breaker which can disconnect the load. The data is sent in packets of 24 bytes which consist of header and data (see Figure 14).

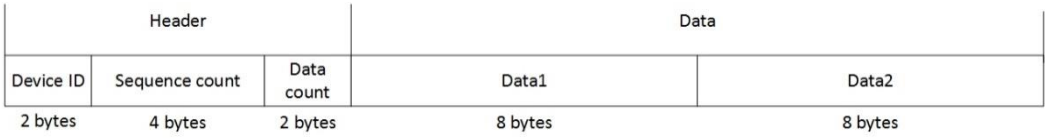

Figure 14 Transmitted UDP packet to simulator

Similar structure packets but with more data points are sent to the server. The data consists of voltage and current measurements from low and high voltage sides and tap position. The measured values are updated with corresponding quality and timestamp attributes.

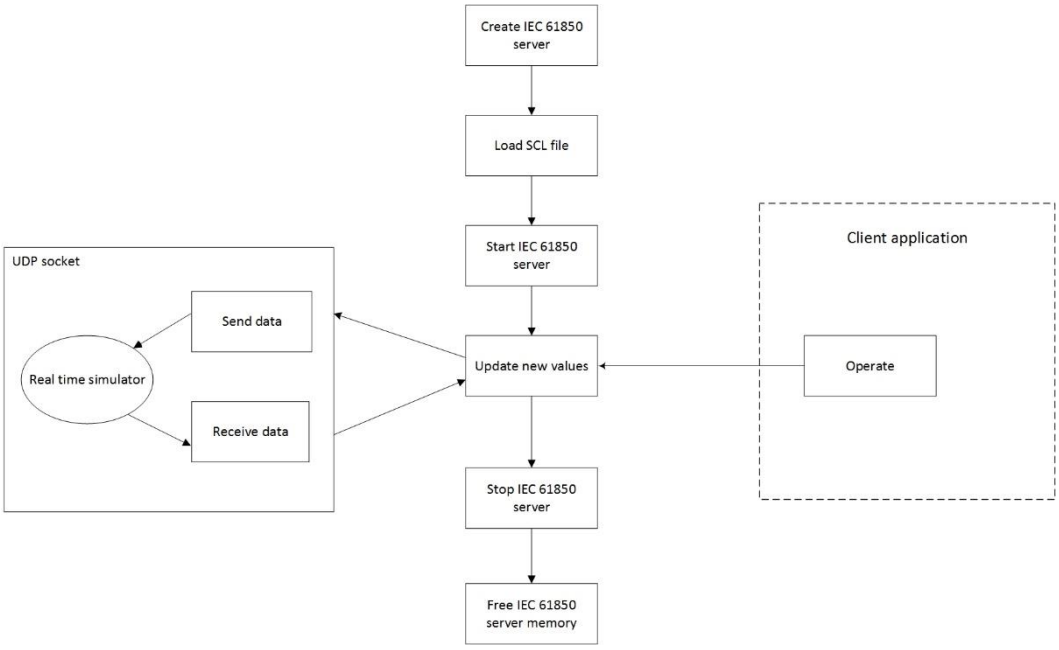

Figure 15 Server application flowchart

The flowchart of the server application is depicted above (see Figure 15). Initially, IEC61850 server object is created with certain parameters and callback functions are designated. SCL file with data attributes IDs is loaded to the server object in order to identify objects in the system model and be able to bind them with variables in the application. After the server has started, additional thread is added to run in continuous loop where UDP socket is created and connection established between the real time simulator. The server listens and sends data packets as described in the previous paragraph (see Figure 14). The operate callback function can be initiated by the client. Entire process of the circuit breaker control will be discussed in the next chapter (see Figure 17). Finally, when user quits the application the server is stopped and the memory is freed of the server object.

5.2.2.3 Client application

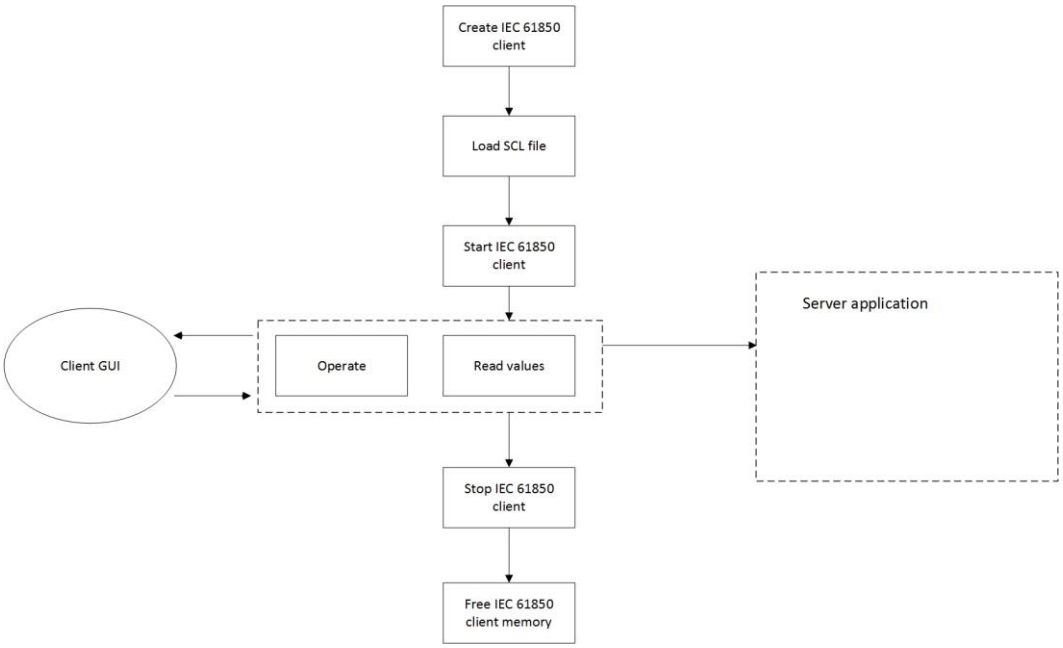

Figure 16 Client application flowchart

The process flow of creating a client is similar to explained in the previous chapter for server application. Expect that the client is a "shadow server" and it is not visible from a network point of view. The client is polling a server for data attribute values and updates them in the GUI.

A user can directly control voltage setpoint and switch off/on circuit breaker. Directly means that the operate command can be performed instantly without any feedback or confirmation from the controlled object. In some instances, for example circuit breaker tripping might not be possible if the spring has not been charged after reclose. Therefore, switchgear cannot accept the command even if the trip signal has been sent instantly. So to facilitate more complex control schemes IEC61850 standard defines other control models in [15].

In this project only direct control will be considered. For instance, if the user wants to change the setpoint value, control function is called and certain value is passed to the function (see Figure 17). The desired value is sent to the server where control callback function is invoked by communication stack. The server application updates the variable and sends new value to the real time simulator where the status is changed accordingly. Then the update callback function is invoked by the communication stack in the client.

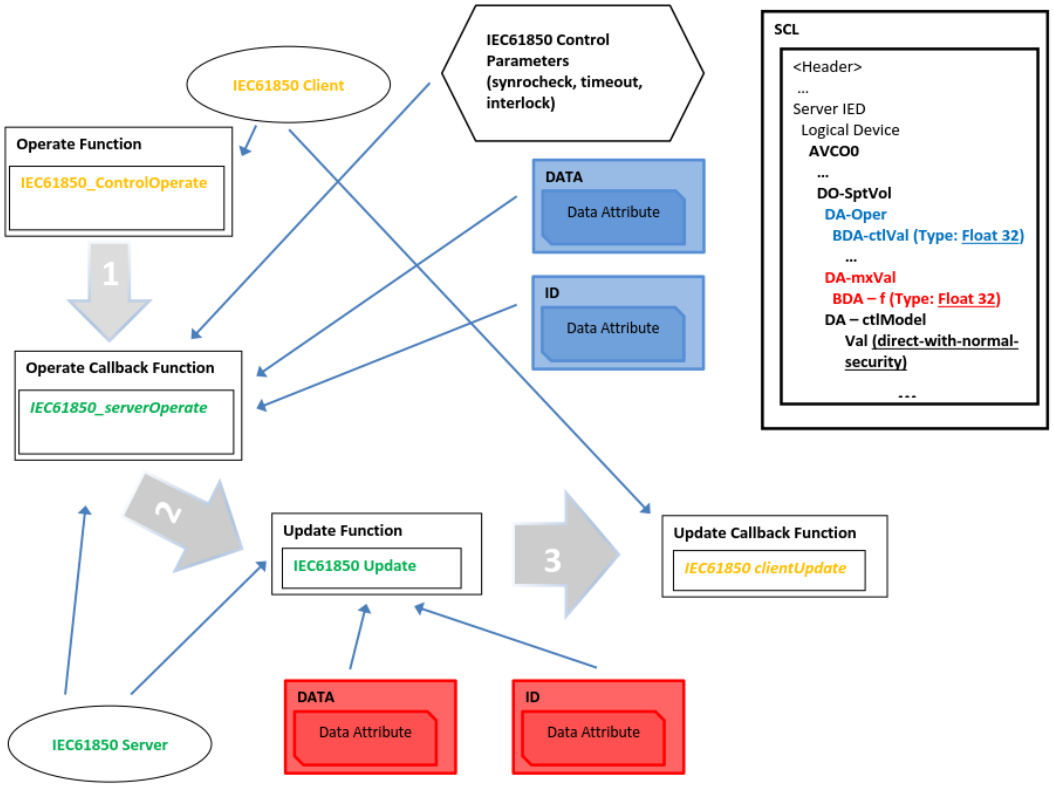

Figure 17 PIS-10 API Client changing measure value (With direct operation and normal security)

5.2.1. Hardware in the loop method

To fully develop a product before launching it into a market it is important to test and make sure that it is functioning properly as intended. Making a hardware prototype of a device or even a physical model of a power system might be expensive and complicated. Ability to change the prototype during or after the developing process is also a welcomed feature which could be too costly. Therefore, designing a mathematical model of a prototype and running a real time simulation with given parameters allows to test without a hardware implementation. Among many benefits like cost reduction or a development time this method also allows to safely test the system without a risk of damaging the equipment with overburden [20].

In this project to investigate communication stack and control algorithm hardware in the loop method is used. Power transformer model with a tap changer is designed and simple power line is created to investigate voltage control and to see dynamics of a power flow and voltage drop over the line. Additionally, switching equipment can be manipulated to disconnect the load.

5.2.2. OPAL-RT simulator

Opal-RT is a real time simulation software that performs simulation in discrete time with constant steps. Time required to solve all equations within step frame can be less, equal or greater then real time frame [21]. However, in this project simulation progresses in equal time duration in each step and the time necessary to perform all computations is less than the time step. Thus, computation time is synchronized with the time clock and the output is always of the same length.

Opal-RT is fully integrated with MATLAB and Simulink. Hence, modeling can be done in graphical environment where Simulink provides extensive customizable block libraries. When the model is built

Opal-RT takes the Simulink model and creates executable C code. User instructs to load the model and OPAL-RT automatically manages synchronization and communication processes. Finally, user can start the simulation and interact with a simulation via graphical user interface.

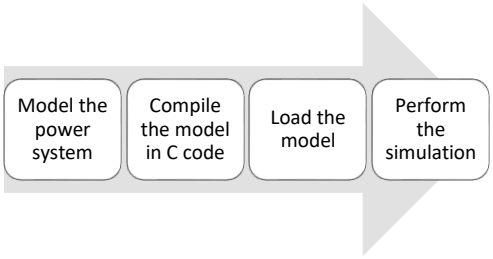

Figure 18 Workflow of power system simulation process

The power system design is depicted in the Figure 19. Power is supplied from a 3-phase voltage source. There are multiple loads connected on the power line to consume generated energy and imitate power flow on the line. Depending on the load the tap changer algorithm tries to maintain set voltage by switching voltage transformer's taps accordingly. Current and voltage measurements are taken in locations B1 and B2 for high voltage and low voltage respectfully.

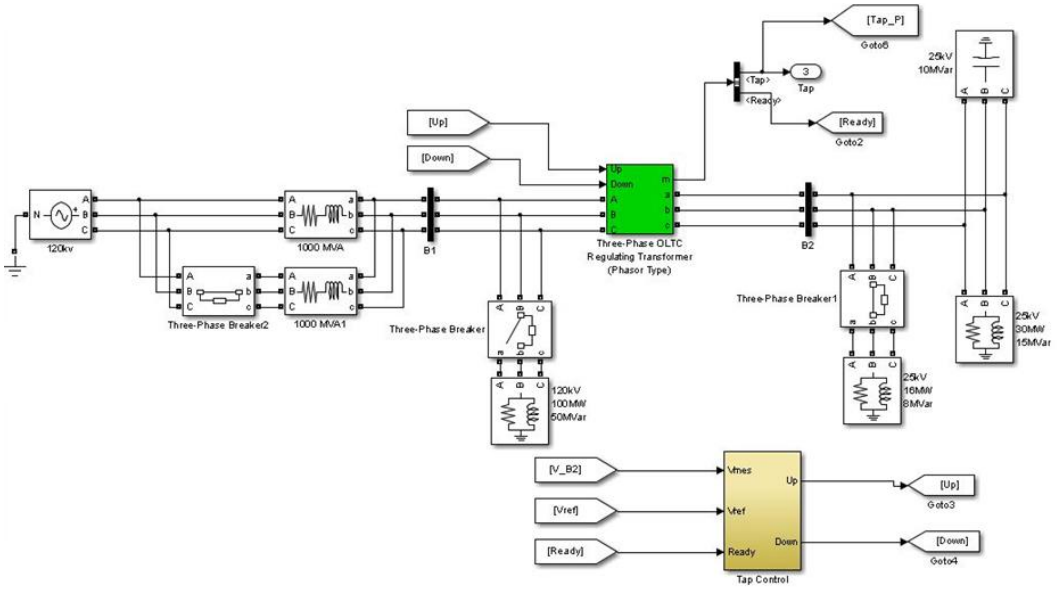

Figure 19 Simulink power system model

5.3 Results

The entire process of a voltage regulation can be investigated by installing a scope function block in the system model where all measurements are acquired and can be inspected in real time during the simulation. Different scenarios are tested changing voltage setpoint value from a client application. Initially, voltage setpoint is set to a nominal value 1 (per units) then the value is increased to 1.05 by the user input from the client application and later the value is changed to 1.1. After that the user lowers setpoint below nominal value to 0.9 and later lowers even more up to 0.85. The results are depicted in Figure 20 below:

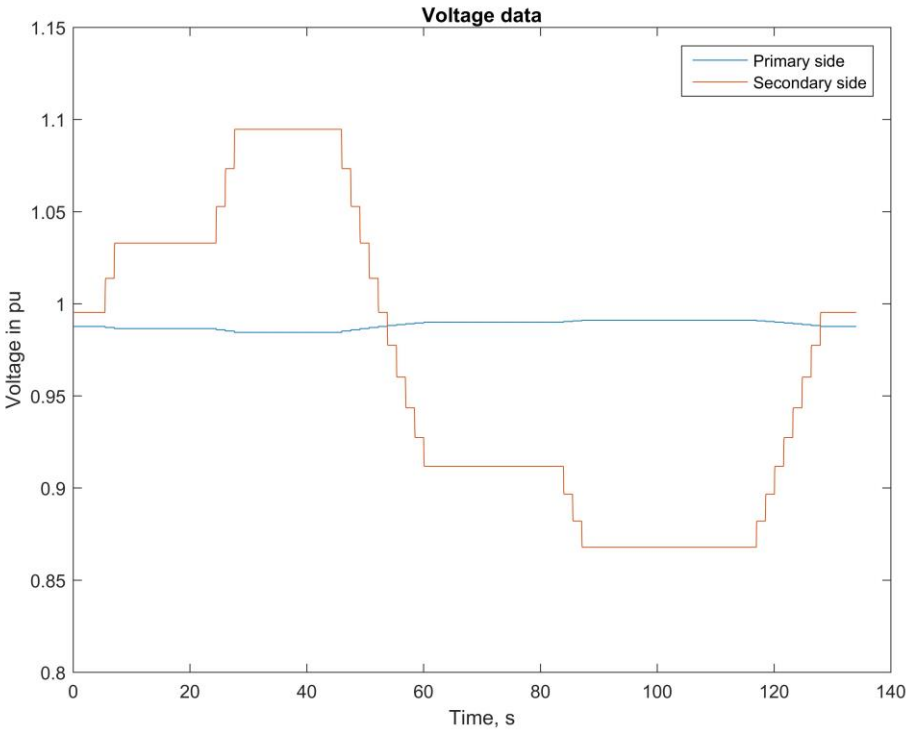

Figure 20 Voltage measurement from the real time simulator when circuit breaker is open.

It can be seen that to regulate the voltage, tap changer position moves accordingly to adjust the required value. As the tap changer can only switch positions with limited number of discreet steps, the voltage does not always equal to the setpoint value. The model in real time simulator of the tap changer is modelled to behave as a real physical device so the movement between positions are not immediate but rather behaves like a mechanical device with a slight delay. The tap changer data is depicted in the figure below:

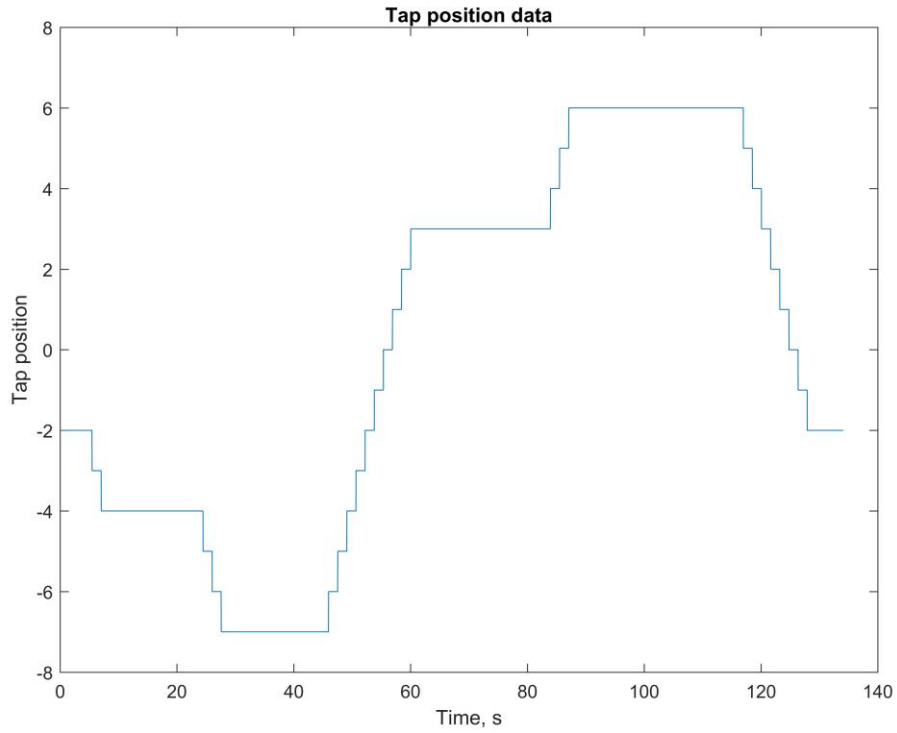

Figure 21 Tap changer position data.

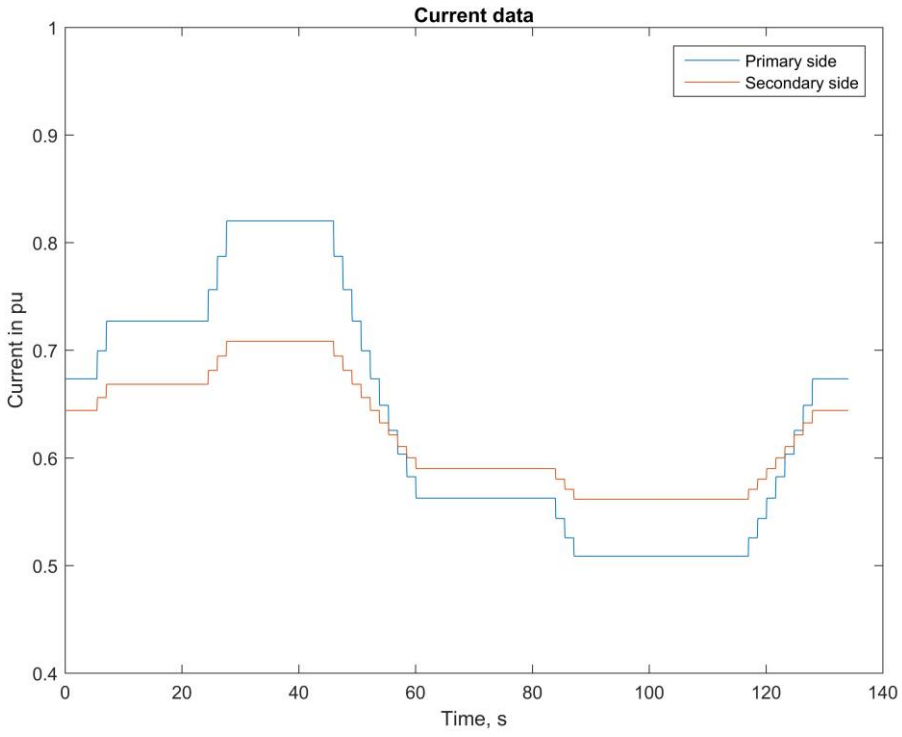

Figure 22 Current measurement from the real time simulator when circuit breaker is open.

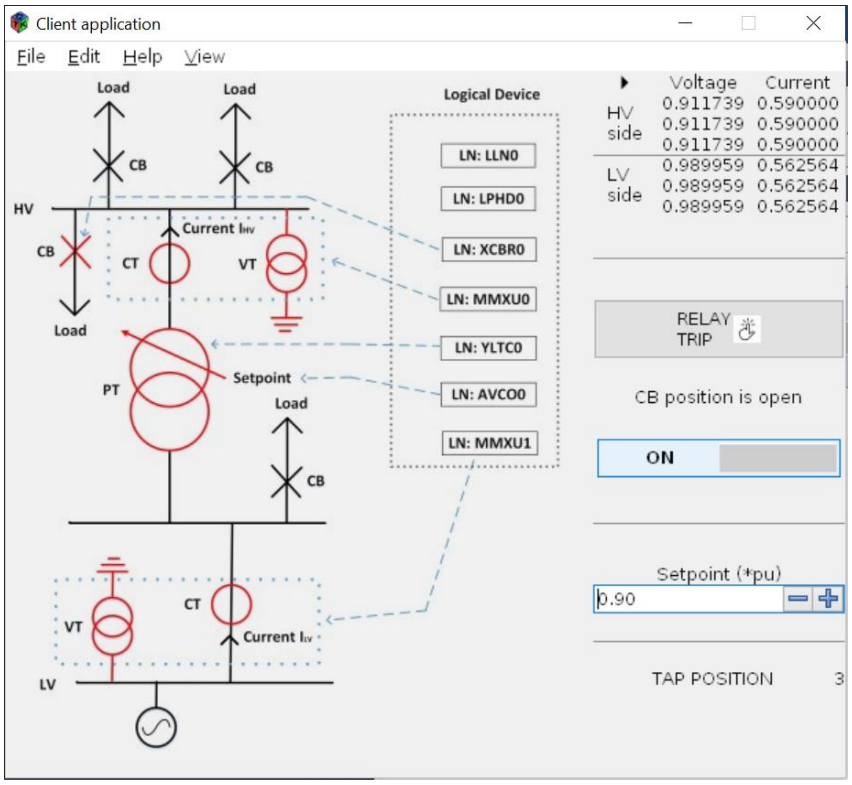

Figure 23 Client application

The client application shows received data from the server (see Figure 23) in top right corner for current and voltage measurements, in bottom right corner for tap position.

A third party tool IED scout manager is used to investigate IEC 61850 devices. In this project it is used to investigate the IEC61850 server. IED scout manager allows to connect the server and inspect its data model (see Figure 24).

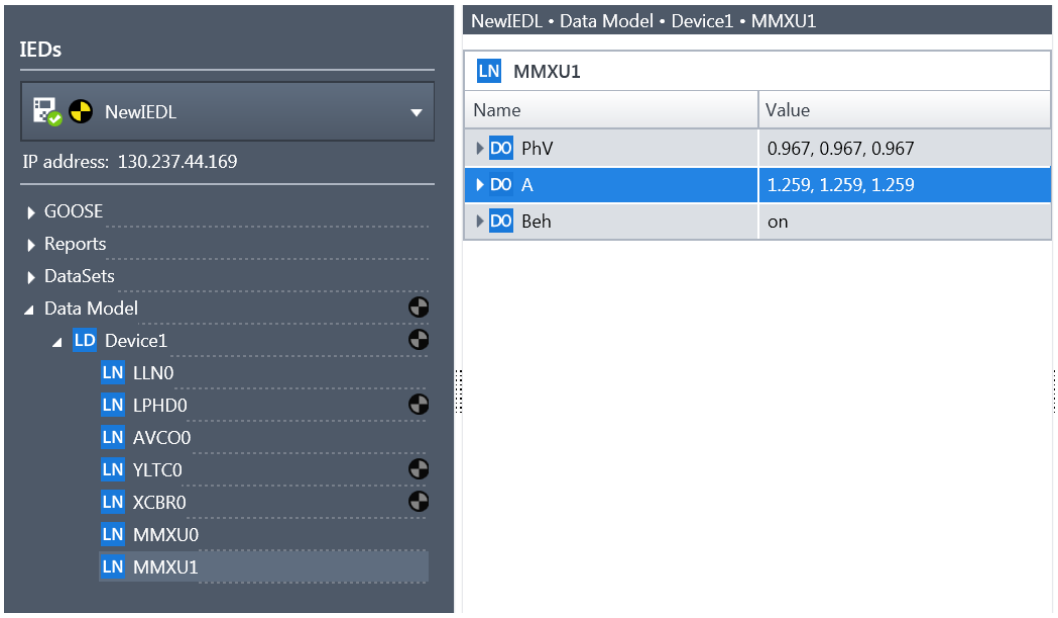

Figure 24 Data model in IED scout manager

The readings of data attributes values can also be compared to the ones received by the client (see Figure 25 and Figure 23). IED scout manager allows to select any individual data attribute from the model. In this example position of the circuit, tap changer position, setpoint value and measurements of voltage and current for one phase were selected.

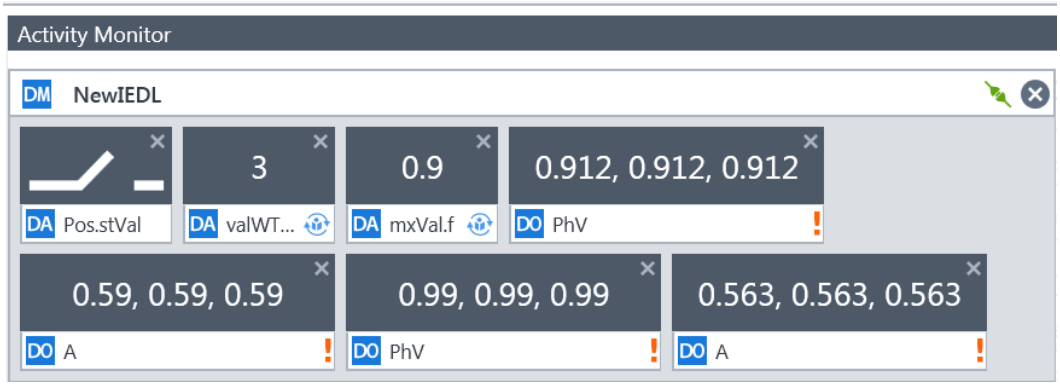

Figure 25 Readings from the server in IED scout manager

5.4 Discussion

The developed power system model and running it in simulator enabled to emulate real processes without physical devices. The simulated processes of course are not ideal representation of real processes since simplifications introduced during the design. However, it is acceptable since the main focus is to generate data to the software applications not to investigate electrical processes in detail. During the simulation output data is sent from the simulator as measurements and position indication. This data is processed in the server and made available for devices compliant to the IEC61850 standard. Likewise, digital signals received from external systems or user inputs are used to control the real processes.

IEC61850 standard provides information model to describe device models and functions commonly used in substation automation and specifies compatible logical node names. Logical nodes grouped into different categories where first character denotes the category. Logical nodes can be arranged in any order and depends on the designer. In this project it was grouped in one single logical device without further consideration. On the other hand, in an extensive power system model it might be worth to consider a systematic way of grouping logical nodes into separate logical devices depending on their functions. However, this grouping is not determined by the standard so it might differ among various vendors.

When the server application loads the SCL file of a virtualized power system model and communication settings. Server makes it available for accessing from remote machines. Different clients compatible to the IEC61850 standard can connect the server. The clients are able to read provided information and operate commands for instance to manipulate switching equipment position. It does not require any additional protocol or layer so full interoperability of different clients is possible.

6 Conclusion

This thesis project demonstrated basic principals of IEC61850 standard and gave concrete solutions based on the standard. The whole cycle of hardware in the loop method works. It allows to develop control algorithms in laboratory environment without a need for physical hardware. The server application is able to communicate with the real time simulator, process data and provides access to different clients. From the user input the client application is able to control the processes in real time simulator from a remote location.

This project showed that to create a control application for a power system device it does not necessary need to be an exclusive proprietary software. The IEC61850 encompasses needed information models and protocols within the field to develop such application. It is very important to have the same language used among all engineers in the field when one designer solutions and can still maintain interoperability with another. In addition, having widely acceptable and time tested standards allow to develop tools which ultimately translates into increased efficiency and improved engineering solutions.

7 Future studies

In the future it would be interesting to study various topics in greater depth which have not been covered during this project. One very acute topic nowadays is security measures, how to protect the power system devices which are connected to the internet from intruders and malicious attacks. For instance, authentication is a very important aspect to consider when a server would allow only specific client to establish connection and gain access to the information inside.

Another future study would be very interesting to investigate communication quality and how it affects the performance. Specific case studies could be carried out to research under what circumstances the quality degrades most and what network capacity is required for a good performance. In addition to that, how network configuration affects the performance and what are the best settings for optimal solution.

8 REFERENCES

[1] H. Radmanesh, S. S. Heidari Yazdi, S. U. Mosazadeh and G. B. Gharehpetian, Studying Voltage Stability in Power System Considering Load Dynamics, Indian Journal of Science and Technology, Vol 6(11), 5487–5494, November 2013

[2] A.Abu-Siada S. Islam and E.A. Mohamed, Adaptive Setting of OLTC to Improve Power Transfer Capability of Power Systems, International Conference on Condition Monitoring and Diagnosis, Beijing, China, April 21-24, 2008

[3] Reducing substation wiring costs with IEC 61850, ABB. 2011

[4] Anders Hagehaugen, Voltage Control in Distribution Network with Local Generation, NTNU, June 2014

[5] Chao Gao, Voltage Control in Distribution Networks using On-Load Tap Changer Transformers, University of Bath, May 2013

[6] Eirik Hammer, Eilev Sivertsen, Analysis and implementation of the IEC 61850 standard, March, 2008

[7] John-Charly Retonda-Modiya, Development Of An Embedded System Actuator Node For Integration Into An Iec 61850 Based Substation Automation Application, November, 2012

[8] http://ec.europa.eu/eurostat/statistics-explained/index.php/Electricity_production,_consumption_and_market_overview [Accessed 2016.04.20]

[9] James H. Harlow, Electric power transformer engineering, CRC Press LLC, 2002

[10] http://www.iec.ch/smartgrid/downloads/smartgridpubs.xls [Accessed 2016.04.23]

[11] IEC 61850 Part 7.1 Basic Communication Structure for Substation and Feeder Equipment: Principles and Models

[12] IEC 61850 Part 7-420 Communication Networks and Systems for Power Utility Automation for Distributed Energy Resources (DER) (Final draft)

[13] IEC 61850 Part 7-4: Basic communication structure – Compatible logical node classes and data object classes

[14] IEC 61850 Part 6 Configuration description language for communication in electrical substations related to IEDs

[15] IEC 61850 Part 7-2: Basic information and communication structure – Abstract communication service interface (ACSI)

[16] Part 8-1: Specific communication service mapping (SCSM) – Mappings to MMS (ISO 9506-1 and ISO 9506-2) and to ISO/IEC 8802-3

[17] Part 9-2: Specific communication service mapping (SCSM) – Sampled values over ISO/IEC 8802-3

[18] An IEC 61850 Process Bus Solution, PAC.SUMMER.2009. Available online at https://www.pacw.org/fileadmin/doc/SummerIssue09/Powerlink_summer09.pdf [Accessed 2016.06.04]

[19] http://api.systemcorp.com.au/PIS10API/V2.06.28/intro.html [Accessed 2016.03.01]

[20] http://www.opal-rt.com/about-hardware-loop-simulation [Accessed 2016.05.01]

[21] J. Bélanger, P. Venne and J.-N. Paquin, The What, Where and Why of Real-Time Simulation, 2010

9 Appendix

9.1 Server application source code

```
/*
Source code of a server application.

Master thesis project:
"DISTRIBUTION OLTC CONTROL USING IEC 61850 CLIENT/SERVER ARCHITECTURE"
Author:
Andrius Maneikis

Communication stack API used:
SystemCORP Embedded Technology Pty Ltd PIS10 V2.06.28

*/

#include <stdio.h>
#include <stdlib.h>
#include <IEC61850API.h> //IEC61850 communication stack
#include <winsock2.h> //UDP connection
#include <windows.h> //Multiple threads

#define SUPPORTED_PROTOCOL IEC61850_SYSTEMCOROP_GENERIC

#define numberObjects 17 // number of objects used in the model

#define numberOfTaps 8

IEC61850 myServer;

/* ********************************Variables for server
data********************************** */
Integer8 tap_position;

Float32 voltage_HV, voltage_LV;
Float32 current_HV, current_LV;

Float32 voltage_HVa,  voltage_HVb,  voltage_HVc;
Float32 voltage_LVa,  voltage_LVb,  voltage_LVc;
Float32 current_HVa,  current_HVb,  current_HVc;
Float32 current_LVa,  current_LVb,  current_LVc;

Unsigned8 usiposition_CB;
Boolean endPositionL, endPositionR;

Float32 setpointEXP;
Boolean CBtripsignalEXP;


//Store data attributes and data attributes IDs
struct IEC61850_DataAttributeData dataObjects[numberObjects];
struct IEC61850_DataAttributeID_Generic dataObjectsID[numberObjects];

//Data to be passed to functions

struct dataBase
{

    Float32 *psetpointEXP;
    Boolean *pCBtripsignalEXP;

    struct IEC61850_DataAttributeData dataObjects[numberObjects];
    struct IEC61850_DataAttributeID_Generic dataObjectsID[numberObjects];

} dataStored1;

typedef struct
{

    Float32 *prefVoltage;
    Boolean *CBtripsignal;

} UserData;

DWORD WINAPI ThreadFunc(void* data)
{
    WSADATA wsadata;                    /* Used to open Windows connection */
    unsigned short port_number;             /* The port number to use */
```

```
    SOCKET sd;                              /* The socket descriptor */
    int server_length;                      /* Length of server struct */
    struct hostent *hp;                     /* Information about the server */
    struct sockaddr_in server;              /* Information about the server */
    struct sockaddr_in client;              /* Information about the client */
    int a1, a2, a3, a4;                     /* Server address components in
xxx.xxx.xxx.xxx form */
    int b1, b2, b3, b4;                     /* Client address components in
xxx.xxx.xxx.xxx form */
    char host_name[256];                /* Host name of this computer */
    char *message[100];

//disable padding in order to have exact same size of bytes in the structure with
different type variables
#pragma pack(push,1)
    struct DatatoSend
    {
        short DevID;
        unsigned int counter;
        short dataNumber;
        double referenceValue;
        double CBtrip;

    } dataset1;
#pragma pack(pop)

    dataset1.DevID = 1;
    dataset1.counter = 1;
    dataset1.dataNumber = 16;

    printf("size of dataset1: %d\n", sizeof(dataset1));
    printf("size of dataset1.DevID: %d\n", sizeof(dataset1.DevID));
    printf("size of dataset1.counter: %d\n", sizeof(dataset1.counter));
    printf("size of dataset1.dataNumber: %d\n", sizeof(dataset1.dataNumber));
    printf("size of dataset1.referenceValue: %d\n", sizeof(dataset1.referenceValue));

    struct reiceivedData
    {
        double header;
        double v1a, v1b, v1c;
        double v2a, v2b, v2c;
        double tap;
        double i1a, i1b, i1c;
        double i2a, i2b, i2c;
    } measure1;


    port_number=27016;

    /* Open windows connection */
    if (WSAStartup(MAKEWORD(2,2), &wsadata)!=0)
    {
        fprintf(stderr, "Could not open Windows connection.\n");
        exit(0);
    }

    /* Open a datagram socket */
    sd = socket(AF_INET, SOCK_DGRAM, 0);
    if (sd == INVALID_SOCKET)
    {
        fprintf(stderr, "Could not create socket.\n");
        WSACleanup();
        exit(0);
    }

    /* Clear out server struct */
    memset((void *)&server, '\0', sizeof(struct sockaddr_in));

    /* Set family and port */
    server.sin_family = AF_INET;
    server.sin_port = htons(port_number);

    /* Set server address */
    server.sin_addr.s_addr = inet_addr("192.168.0.11");
```

```
    /* Clear out client struct */
    memset((void *)&client, '\0', sizeof(struct sockaddr_in));

    /* Set family and port */
    client.sin_family = AF_INET;
    client.sin_port = htons(port_number);


    /* Get host name of this computer */
    gethostname(host_name, sizeof(host_name));
    hp = gethostbyname(host_name);

    /* Check for NULL pointer */
    if (hp == NULL)
    {
        fprintf(stderr, "Could not get host name.\n");
        closesocket(sd);
        WSACleanup();
        exit(0);
    }

    /* Assign the address */
    client.sin_addr.S_un.S_un_b.s_b1 = hp->h_addr_list[0][0];
    client.sin_addr.S_un.S_un_b.s_b2 = hp->h_addr_list[0][1];
    client.sin_addr.S_un.S_un_b.s_b3 = hp->h_addr_list[0][2];
    client.sin_addr.S_un.S_un_b.s_b4 = hp->h_addr_list[0][3];


    /* Bind local address to socket */
    if (bind(sd, (struct sockaddr *)&client, sizeof(struct sockaddr_in)) == -1)
    {
        fprintf(stderr, "Cannot bind address to socket.\n");
        closesocket(sd);
        WSACleanup();
        exit(0);
    }
    printf("Client is listening\n");


    server_length = (int)sizeof(struct sockaddr_in);

    /*
*********************************************************************************************
    quality and time data and data IDs */

    //For quality

    unsigned short int usiObjectQuality;
    usiObjectQuality= (IEC61850_QUALITY_GOOD);
    //usiObjectQuality= (IEC61850_QUALITY_FAILURE | IEC61850_QUALITY_INVALID |
IEC61850_QUALITY_OLDDATA);

    struct IEC61850_DataAttributeID_Generic  qualityDataIDtemp;
    struct IEC61850_DataAttributeData qualityDatatemp;

    qualityDatatemp.ucType = IEC61850_DATATYPE_QUALITY;
    qualityDatatemp.uiBitLength = IEC61850_QUALITY_BITSIZE;
    qualityDatatemp.iArrayIndex = -1;
    qualityDatatemp.pvData = &usiObjectQuality;

    //For time
    struct IEC61850_TimeStamp  tTime;
    struct IEC61850_DataAttributeID_Generic  timeDataIDtemp;
    struct IEC61850_DataAttributeData timeDatatemp;

    timeDatatemp.ucType = IEC61850_DATATYPE_TIMESTAMP;
    timeDatatemp.uiBitLength = sizeof(tTime)*8;
    timeDatatemp.iArrayIndex = -1;
    timeDatatemp.pvData = &tTime;

    while (1)
    {
        //Update setpoint value to be sent
        dataset1.referenceValue = setpointEXP;
        //dataset1.CBtrip = CBtripsignalEXP;
```

```
        if (CBtripsignalEXP == 0){

        dataset1.CBtrip = 1;
    } else {

        dataset1.CBtrip = 0;
    }

        enum IEC61850_ErrorCodes error = IEC61850_ERROR_NONE;

        error = IEC61850_Update(myServer, (struct IEC61850_DataAttributeID *)&dataStored1.
dataObjectsID[14], &dataStored1.dataObjects[14], 1);

        if(error != IEC61850_ERROR_NONE)
        {
            printf("Update failed: %s (%u)\n", IEC61850_ErrorString(error), error);

        }

        //Set value of the setpoint quality dataID attribute
        memcpy(&qualityDataIDtemp, &dataStored1.dataObjectsID[14], sizeof(
qualityDataIDtemp));
        qualityDataIDtemp.uiField3=1;

        //update quality of voltage setpoint
        error = IEC61850_Update(myServer, (struct IEC61850_DataAttributeID *)&
qualityDataIDtemp, &qualityDatatemp, 1);
        if(error != IEC61850_ERROR_NONE)
        {
            printf("Update failed: %s (%u)\n", IEC61850_ErrorString(error), error);
        }

        //update setpoint value time

        if(IEC61850_GetTime(myServer, &tTime) != IEC61850_ERROR_NONE)
        {
            printf("Get time failed: %s (%u)\n", IEC61850_ErrorString(error), error);
        }
        memcpy(&timeDataIDtemp, &dataStored1.dataObjectsID[14], sizeof(timeDataIDtemp));
        timeDataIDtemp.uiField3=2;

        error = IEC61850_Update(myServer, (struct IEC61850_DataAttributeID *)&
timeDataIDtemp, &timeDatatemp, 1);
        if(error != IEC61850_ERROR_NONE)
        {
            printf("Update failed: %s (%u)\n", IEC61850_ErrorString(error), error);

        }


        /* Send reference voltage value for the tap changer */

        if (sendto(sd, &dataset1, (int)sizeof(dataset1), 0, (struct sockaddr *)&server,
server_length)==-1)
        {
            fprintf(stderr, "Error sending datagram.\n");
            closesocket(sd);
            WSACleanup();
            exit(0);
        }

        if (recvfrom(sd, (void*) &measure1, sizeof(measure1), 0, (struct sockaddr *)&
server, &server_length) < 0)
        {
            fprintf(stderr, "Error receiving data.\n");
            closesocket(sd);
            WSACleanup();
            exit(0);
        }

        /* Save data received from the simulator */

        voltage_LVa = measure1.v1a;
        voltage_LVb = measure1.v1b;
        voltage_LVc = measure1.v1c;
```

```
        voltage_HVa = measure1.v2a;
        voltage_HVb = measure1.v2b;
        voltage_HVc = measure1.v2c;

        current_LVa = measure1.i1a;
        current_LVb = measure1.i1b;
        current_LVc = measure1.i1c;
        current_HVa = measure1.i2a;
        current_HVb = measure1.i2b;
        current_HVc = measure1.i2c;

        tap_position = measure1.tap;

        //Update voltages and currents received from the simulator
        int i;

        for (i = 0; i < 12; i++ ){


              //Update  voltage
        error = IEC61850_Update(myServer, (struct IEC61850_DataAttributeID *)&dataStored1.
dataObjectsID[i], &dataStored1.dataObjects[i], 1);

        if(error != IEC61850_ERROR_NONE)
        {
            printf("Update failed: %s (%u)\n", IEC61850_ErrorString(error), error);

        }

        //Set quality value of secondary voltage dataID attribute
        memcpy(&qualityDataIDtemp, &dataStored1.dataObjectsID[i], sizeof(qualityDataIDtemp
));
        qualityDataIDtemp.uiField3=1;

        //update quality of secondary voltage
        error = IEC61850_Update(myServer, (struct IEC61850_DataAttributeID *)&
qualityDataIDtemp, &qualityDatatemp, 1);
        if(error != IEC61850_ERROR_NONE)
        {
            printf("Update failed: %s (%u)\n", IEC61850_ErrorString(error), error);

        }

        //update secondary voltage time value

        if(IEC61850_GetTime(myServer, &tTime) != IEC61850_ERROR_NONE)
        {
            printf("Get time failed: %s (%u)\n", IEC61850_ErrorString(error), error);
        }
        memcpy(&timeDataIDtemp, &dataStored1.dataObjectsID[i], sizeof(timeDataIDtemp));
        timeDataIDtemp.uiField3=2;

        error = IEC61850_Update(myServer, (struct IEC61850_DataAttributeID *)&
timeDataIDtemp, &timeDatatemp, 1);
        if(error != IEC61850_ERROR_NONE)
        {
            printf("Update failed: %s (%u)\n", IEC61850_ErrorString(error), error);

        }
           } // for loop end

        /* ************************************************ */

        printf("Tap position: %d\n", (int) measure1.tap);

        tap_position=(int)measure1.tap;

        //Update tap position
        error = IEC61850_Update(myServer, (struct IEC61850_DataAttributeID *)&dataStored1.
dataObjectsID[13], &dataStored1.dataObjects[13], 1);

        if(error != IEC61850_ERROR_NONE)
        {
            printf("Update failed: %s (%u)\n", IEC61850_ErrorString(error), error);
```

```
        }

        //Set value of the tap position quality dataID attribute
        memcpy(&qualityDataIDtemp, &dataStored1.dataObjectsID[13], sizeof(
qualityDataIDtemp));
        qualityDataIDtemp.uiField3=1;

        //update quality of tap position
        error = IEC61850_Update(myServer, (struct IEC61850_DataAttributeID *)&
qualityDataIDtemp, &qualityDatatemp, 1);
        if(error != IEC61850_ERROR_NONE)
        {
            printf("Update failed: %s (%u)\n", IEC61850_ErrorString(error), error);

        }

        //update setpoint value time

        if(IEC61850_GetTime(myServer, &tTime) != IEC61850_ERROR_NONE)
        {
            printf("Get time failed: %s (%u)\n", IEC61850_ErrorString(error), error);
        }
        memcpy(&timeDataIDtemp, &dataStored1.dataObjectsID[13], sizeof(timeDataIDtemp));
        timeDataIDtemp.uiField3=2;

        error = IEC61850_Update(myServer, (struct IEC61850_DataAttributeID *)&
timeDataIDtemp, &timeDatatemp, 1);
        if(error != IEC61850_ERROR_NONE)
        {
            printf("Update failed: %s (%u)\n", IEC61850_ErrorString(error), error);

        }

        //******************************

        //Tap end position indication
        if (tap_position == numberOfTaps)
        {

            endPositionR = 1;
            qualityDataIDtemp.uiField3=1;
            qualityDataIDtemp.uiField2=0;
            error = IEC61850_Update(myServer, (struct IEC61850_DataAttributeID *)&
dataStored1.dataObjectsID[15], &dataStored1.dataObjects[15], 1);
            error = IEC61850_Update(myServer, (struct IEC61850_DataAttributeID *)&
qualityDataIDtemp, &qualityDatatemp, 1);
            qualityDataIDtemp.uiField3=2;
            IEC61850_GetTime(myServer, &tTime);
            error = IEC61850_Update(myServer, (struct IEC61850_DataAttributeID *)&
timeDataIDtemp, &timeDatatemp, 1);
            if(error != IEC61850_ERROR_NONE)
            {
                printf("Update failed: %s (%u)\n", IEC61850_ErrorString(error), error);

            }

        }
        else if ((tap_position < numberOfTaps) && endPositionR == 1)
        {

            endPositionR=0;
            qualityDataIDtemp.uiField3=1;
            qualityDataIDtemp.uiField2=0;
            error = IEC61850_Update(myServer, (struct IEC61850_DataAttributeID *)&
dataStored1.dataObjectsID[15], &dataStored1.dataObjects[15], 1);
            error = IEC61850_Update(myServer, (struct IEC61850_DataAttributeID *)&
qualityDataIDtemp, &qualityDatatemp, 1);
            qualityDataIDtemp.uiField3=2;
            IEC61850_GetTime(myServer, &tTime);
            error = IEC61850_Update(myServer, (struct IEC61850_DataAttributeID *)&
timeDataIDtemp, &timeDatatemp, 1);
            if(error != IEC61850_ERROR_NONE)
            {
                printf("Update failed: %s (%u)\n", IEC61850_ErrorString(error), error);
            }
```

```
        }
        else if (tap_position == -numberOfTaps)
        {

            endPositionL = 1;
            qualityDataIDtemp.uiField3=1;
            qualityDataIDtemp.uiField2=1;
            error = IEC61850_Update(myServer, (struct IEC61850_DataAttributeID *)&
dataStored1.dataObjectsID[16], &dataStored1.dataObjects[16], 1);
            error = IEC61850_Update(myServer, (struct IEC61850_DataAttributeID *)&
qualityDataIDtemp, &qualityDatatemp, 1);
            qualityDataIDtemp.uiField3=2;
            IEC61850_GetTime(myServer, &tTime);
            error = IEC61850_Update(myServer, (struct IEC61850_DataAttributeID *)&
timeDataIDtemp, &timeDatatemp, 1);
            if(error != IEC61850_ERROR_NONE)
            {
                printf("Update failed: %s (%u)\n", IEC61850_ErrorString(error), error);
            }

        }
        else if ((tap_position > -numberOfTaps) && endPositionL == 1)
    {

        endPositionL=0;
        qualityDataIDtemp.uiField3=1;
        qualityDataIDtemp.uiField2=1;
        error = IEC61850_Update(myServer, (struct IEC61850_DataAttributeID *)&dataStored1.
dataObjectsID[16], &dataStored1.dataObjects[16], 1);
            error = IEC61850_Update(myServer, (struct IEC61850_DataAttributeID *)&
qualityDataIDtemp, &qualityDatatemp, 1);
            qualityDataIDtemp.uiField3=2;
            IEC61850_GetTime(myServer, &tTime);
            error = IEC61850_Update(myServer, (struct IEC61850_DataAttributeID *)&
timeDataIDtemp, &timeDatatemp, 1);
            if(error != IEC61850_ERROR_NONE)
            {
                printf("Update failed: %s (%u)\n", IEC61850_ErrorString(error), error);
            }
        }

        printf("Received. Voltage v1: %f  %f  %f\n", measure1.v1a, measure1.v1b, measure1.
v1c);
        printf("Received. Voltage v2: %f  %f  %f\n", measure1.v2a, measure1.v2b, measure1.
v2c);
        printf("Received. Tap position: %d\n", (int) measure1.tap);
        printf("Received. Current i1: %f  %f  %f\n", measure1.i1a, measure1.i1b, measure1.
i1c);
        printf("Received. Current i2: %f  %f  %f\n", measure1.i2a, measure1.i2b,  measure1
.i2c);
        printf("**********\n");
        printf("Sent. Setpoint value: %f\n", dataset1.referenceValue);
        printf("Sent. CB control signal to simulator: %d\n", (int)dataset1.CBtrip);
    }

    closesocket(sd);
    WSACleanup();

    return 0;
}

/*Callback function prototypes ************************************************
****************************************************************************** */

//Read Callback function

int myReadFunction (void *ptUserData, struct IEC61850_DataAttributeID *ptDataAttributeID,
struct IEC61850_DataAttributeData *ptReturnedValue);


//Operative Test Callback function
enum eCommandAddCause OperativeTestCallbackHandler(void * ptUserData, struct
IEC61850_DataAttributeID* ptControlID, struct IEC61850_CommandParameters*
ptOperativeTestParameters);
```

```
//Opeare callback function
int myOperateFunction (void *ptUserData, struct IEC61850_DataAttributeID *ptControlID,
const struct IEC61850_DataAttributeData *ptOperateValue, struct IEC61850_CommandParameters
 *ptOperateParameters);

int main(int argc, char *argv[])
{

    enum IEC61850_ErrorCodes error = IEC61850_ERROR_NONE;

    //Check compatibility
    if(strcmp((PIS10_VERSION), IEC61850_GetLibraryVersion()) ==0)     //Test for Library
and header version matching. Important!
    {
        printf("Library Version: %s\n Header Version: %s\n", IEC61850_GetLibraryVersion(),
PIS10_VERSION);
    }

    char filepathServer[]="files\\server v1.01.cid";
    printf("File path for the Server ICD file: \n%s\n\n", filepathServer);


    //Create server

    struct IEC61850_Parameters tServerParam = {0};
    memset(&tServerParam, 0, sizeof(struct IEC61850_Parameters));

    do
    {
        tServerParam.ClientServerFlag = IEC61850_SERVER;
        tServerParam.Ed1_Ed2_Flag=IEC61850_Edition2;
        tServerParam.ptReadCallback = NULL;
        tServerParam.ptWriteCallback = NULL;
        tServerParam.ptUpdateCallback = NULL;
        tServerParam.ptOprTestCallback = OperativeTestCallbackHandler;
        tServerParam.ptOperateCallback = myOperateFunction;

        myServer = IEC61850_Create(&tServerParam, &error);

        if (myServer == NULL)
        {
            printf(" Failed : %i", error);
            break;
        }

        //Load SCL configuration file of the server
        printf("\r\n Server load SCL");
        error = IEC61850_LoadSCLFile(myServer, filepathServer); //Load ICD file
        if (error != IEC61850_ERROR_NONE)
        {
            printf(" Failed : %i, \t%s", error, IEC61850_ErrorString(error));
            return;
        }


        /*
*****************************************************************************

***************************************************************************** */
        //Starting server
        printf("\r\n Server Start");
        error = IEC61850_Start(myServer);
        if(error != IEC61850_ERROR_NONE)
        {
            printf(" Failed : %i, \t%s", error, IEC61850_ErrorString(error));
            break;
        }
    }
    while(0);

    printf("\r\n Server started.\n");

    //Initialize setpoint reference voltage
    usiposition_CB = 128; //position of circuit breaker is closed
```

```
setpointEXP=1;
CBtripsignalEXP = 0; //on state signal to circuit breaker

//Initialize data attributes
//MMXU0 Voltage measurements
//Voltage measurement of HV side phase A
struct IEC61850_DataAttributeData voltage_HVaData;
voltage_HVaData.ucType = IEC61850_DATATYPE_FLOAT32;
voltage_HVaData.uiBitLength = sizeof(voltage_HVa)*8;
voltage_HVaData.iArrayIndex = -1;
voltage_HVaData.pvData = &voltage_HVa;

dataStored1.dataObjects[0]=voltage_HVaData;

struct IEC61850_DataAttributeID_Generic  voltage_HVaDataID;
voltage_HVaDataID.Generic_type = IEC61850_DAID_GENERIC;
voltage_HVaDataID.uiField1  = 1;
voltage_HVaDataID.uiField2 = 1;
voltage_HVaDataID.uiField3 = 0;
voltage_HVaDataID.uiField4 = 0;
voltage_HVaDataID.uiField5 = 0;

dataStored1.dataObjectsID[0]=voltage_HVaDataID;

//Voltage measurement of HV side phase B
struct IEC61850_DataAttributeData voltage_HVbData;
memcpy(&voltage_HVbData, &voltage_HVaData, sizeof(voltage_HVaData));
voltage_HVbData.pvData = &voltage_HVb;

dataStored1.dataObjects[1]=voltage_HVbData;

struct IEC61850_DataAttributeID_Generic  voltage_HVbDataID;
memcpy(&voltage_HVbDataID, &voltage_HVaDataID, sizeof(voltage_HVaDataID));
voltage_HVbDataID.uiField5 = 1;

dataStored1.dataObjectsID[1]=voltage_HVbDataID;

//Voltage measurement of HV side phase C
struct IEC61850_DataAttributeData voltage_HVcData;
memcpy(&voltage_HVcData, &voltage_HVaData, sizeof(voltage_HVaData));
voltage_HVcData.pvData = &voltage_HVc;

dataStored1.dataObjects[2]=voltage_HVcData;

struct IEC61850_DataAttributeID_Generic  voltage_HVcDataID;
memcpy(&voltage_HVcDataID, &voltage_HVaDataID, sizeof(voltage_HVaDataID));
voltage_HVcDataID.uiField5 = 2;

dataStored1.dataObjectsID[2]=voltage_HVcDataID;

//MMXU0 Current measurements
//Current measurement of HV side phase A
struct IEC61850_DataAttributeData current_HVaData;
memcpy(&current_HVaData, &voltage_HVaData, sizeof(voltage_HVaData));
current_HVaData.pvData = &current_HVa;

dataStored1.dataObjects[3]=current_HVaData;

struct IEC61850_DataAttributeID_Generic  current_HVaDataID;
memcpy(&current_HVaDataID, &voltage_HVaDataID, sizeof(voltage_HVaDataID));
current_HVaDataID.uiField2 = 2;


dataStored1.dataObjectsID[3]=current_HVaDataID;

//Current measurement of HV side phase B
struct IEC61850_DataAttributeData current_HVbData;
memcpy(&current_HVbData, &voltage_HVaData, sizeof(voltage_HVaData));
current_HVbData.pvData = &current_HVb;

dataStored1.dataObjects[4]=current_HVbData;

struct IEC61850_DataAttributeID_Generic  current_HVbDataID;
memcpy(&current_HVbDataID, &voltage_HVbDataID, sizeof(voltage_HVbDataID));
current_HVbDataID.uiField2 = 2;
```

```
dataStored1.dataObjectsID[4]=current_HVbDataID;

//Current measurement of HV side phase C
struct IEC61850_DataAttributeData current_HVcData;
memcpy(&current_HVcData, &voltage_HVaData, sizeof(voltage_HVaData));
current_HVcData.pvData = &current_HVc;

dataStored1.dataObjects[5]=current_HVcData;

struct IEC61850_DataAttributeID_Generic  current_HVcDataID;
memcpy(&current_HVcDataID, &voltage_HVcDataID, sizeof(voltage_HVcDataID));
current_HVcDataID.uiField2 = 2;

dataStored1.dataObjectsID[5]=current_HVcDataID;

//MMXU1 Voltage measurements
//Voltage measurement of LV side phase A
struct IEC61850_DataAttributeData voltage_LVaData;
memcpy(&voltage_LVaData, &voltage_HVaData, sizeof(voltage_HVaData));
voltage_LVaData.pvData = &voltage_LVa;

dataStored1.dataObjects[6]=voltage_LVaData;

struct IEC61850_DataAttributeID_Generic  voltage_LVaDataID;
memcpy(&voltage_LVaDataID, &voltage_HVaDataID, sizeof(voltage_HVaDataID));
voltage_LVaDataID.uiField1 = 2;

dataStored1.dataObjectsID[6]=voltage_LVaDataID;

//Voltage measurement of LV side phase B
struct IEC61850_DataAttributeData voltage_LVbData;
memcpy(&voltage_LVbData, &voltage_HVaData, sizeof(voltage_HVaData));
voltage_LVbData.pvData = &voltage_LVb;

dataStored1.dataObjects[7]=voltage_LVbData;

struct IEC61850_DataAttributeID_Generic  voltage_LVbDataID;
memcpy(&voltage_LVbDataID, &voltage_HVbDataID, sizeof(voltage_HVbDataID));
voltage_LVbDataID.uiField1 = 2;

dataStored1.dataObjectsID[7]=voltage_LVbDataID;

//Voltage measurement of LV side phase C
struct IEC61850_DataAttributeData voltage_LVcData;
memcpy(&voltage_LVcData, &voltage_HVaData, sizeof(voltage_HVaData));
voltage_LVcData.pvData = &voltage_LVc;

dataStored1.dataObjects[8]=voltage_LVcData;

struct IEC61850_DataAttributeID_Generic  voltage_LVcDataID;
memcpy(&voltage_LVcDataID, &voltage_HVcDataID, sizeof(voltage_HVcDataID));
voltage_LVcDataID.uiField1 = 2;

dataStored1.dataObjectsID[8]=voltage_LVcDataID;

//MMXU1 Current measurements
//Current measurement of LV side phase A
struct IEC61850_DataAttributeData current_LVaData;
memcpy(&current_LVaData, &voltage_HVaData, sizeof(voltage_HVaData));
current_LVaData.pvData = &current_LVa;

dataStored1.dataObjects[9]=current_LVaData;

struct IEC61850_DataAttributeID_Generic  current_LVaDataID;
memcpy(&current_LVaDataID, &current_HVaDataID, sizeof(current_HVaDataID));
current_LVaDataID.uiField1  = 2;

dataStored1.dataObjectsID[9]=current_LVaDataID;

//Current measurement of LV side phase B
struct IEC61850_DataAttributeData current_LVbData;
memcpy(&current_LVbData, &voltage_HVaData, sizeof(voltage_HVaData));
current_LVbData.pvData = &current_LVb;
```

```
dataStored1.dataObjects[10]=current_LVbData;

struct IEC61850_DataAttributeID_Generic  current_LVbDataID;
memcpy(&current_LVbDataID, &current_HVbDataID, sizeof(current_HVbDataID));
current_LVbDataID.uiField1  = 2;


dataStored1.dataObjectsID[10]=current_LVbDataID;

//Current measurement of LV side phase C
struct IEC61850_DataAttributeData current_LVcData;
memcpy(&current_LVcData, &voltage_HVaData, sizeof(voltage_HVaData));
current_LVcData.pvData = &current_LVc;

dataStored1.dataObjects[11]=current_LVcData;

struct IEC61850_DataAttributeID_Generic  current_LVcDataID;
memcpy(&current_LVcDataID, &current_HVcDataID, sizeof(current_HVcDataID));
current_LVcDataID.uiField1  = 2;

dataStored1.dataObjectsID[11]=current_LVcDataID;

//Circuit breaker position
struct IEC61850_DataAttributeData circuit_breaker;
circuit_breaker.ucType = IEC61850_DATATYPE_DBPOS;
circuit_breaker.uiBitLength = sizeof(usiposition_CB)*8;
circuit_breaker.iArrayIndex = -1;
circuit_breaker.pvData = &usiposition_CB;

dataStored1.dataObjects[12]=circuit_breaker;

struct IEC61850_DataAttributeID_Generic  circuit_breakerID;
circuit_breakerID.Generic_type = IEC61850_DAID_GENERIC;
circuit_breakerID.uiField1  = 3;
circuit_breakerID.uiField2 = 0;
circuit_breakerID.uiField3 = 0;
circuit_breakerID.uiField4 = 0;
circuit_breakerID.uiField5 = 0;

dataStored1.dataObjectsID[12]=circuit_breakerID;

//Tap changer position
struct IEC61850_DataAttributeData tap_changer;
tap_changer.ucType = IEC61850_DATATYPE_INT8;
tap_changer.uiBitLength = sizeof(tap_position)*8;
tap_changer.iArrayIndex = -1;
tap_changer.pvData = &tap_position;

dataStored1.dataObjects[13]=tap_changer;

struct IEC61850_DataAttributeID_Generic  tap_changerID;
tap_changerID.Generic_type = IEC61850_DAID_GENERIC;
tap_changerID.uiField1  = 4;
tap_changerID.uiField2 = 2;
tap_changerID.uiField3 = 0;
tap_changerID.uiField4 = 0;
tap_changerID.uiField5 = 0;

dataStored1.dataObjectsID[13]=tap_changerID;

//Voltage setpoint

struct IEC61850_DataAttributeData voltage_setpoint;
voltage_setpoint.ucType = IEC61850_DATATYPE_FLOAT32;
voltage_setpoint.uiBitLength = sizeof(setpointEXP)*8;
voltage_setpoint.iArrayIndex = -1;
voltage_setpoint.pvData = &setpointEXP;

dataStored1.dataObjects[14]=voltage_setpoint;

struct IEC61850_DataAttributeID_Generic  voltage_setpointID;
voltage_setpointID.Generic_type = IEC61850_DAID_GENERIC;
voltage_setpointID.uiField1  = 5;
voltage_setpointID.uiField2 = 0;
```

```
    voltage_setpointID.uiField3 = 0;
    voltage_setpointID.uiField4 = 0;
    voltage_setpointID.uiField5 = 0;

    dataStored1.dataObjectsID[14]=voltage_setpointID;

    //Voltage supervision EndPosR

    struct IEC61850_DataAttributeData EndPosRData;
    EndPosRData.ucType = IEC61850_DATATYPE_BOOLEAN;
    EndPosRData.uiBitLength = sizeof(endPositionR)*8;
    EndPosRData.iArrayIndex = -1;
    EndPosRData.pvData = &endPositionR;

    dataStored1.dataObjects[15]=EndPosRData;

    struct IEC61850_DataAttributeID_Generic  EndPosRDataID;
    EndPosRDataID.Generic_type = IEC61850_DAID_GENERIC;
    EndPosRDataID.uiField1  = 4;
    EndPosRDataID.uiField2 = 0;
    EndPosRDataID.uiField3 = 0;
    EndPosRDataID.uiField4 = 0;
    EndPosRDataID.uiField5 = 0;

    dataStored1.dataObjectsID[15]=EndPosRDataID;

    //Voltage supervision EndPosR

    struct IEC61850_DataAttributeData EndPosLData;
    EndPosLData.ucType = IEC61850_DATATYPE_BOOLEAN;
    EndPosLData.uiBitLength = sizeof(endPositionL)*8;
    EndPosLData.iArrayIndex = -1;
    EndPosLData.pvData = &endPositionL;

    dataStored1.dataObjects[16]=EndPosLData;

    struct IEC61850_DataAttributeID_Generic  EndPosLDataID;
    EndPosLDataID.Generic_type = IEC61850_DAID_GENERIC;
    EndPosLDataID.uiField1  = 4;
    EndPosLDataID.uiField2 = 1;
    EndPosLDataID.uiField3 = 0;
    EndPosLDataID.uiField4 = 0;
    EndPosLDataID.uiField5 = 0;

    dataStored1.dataObjectsID[16]=EndPosLDataID;

    UserData UserDataset;
    UserDataset.prefVoltage = &setpointEXP;
    UserDataset.CBtripsignal = &CBtripsignalEXP;


    error = IEC61850_SetUserData(myServer, &UserDataset); //provides user data for the
callback functions

    /*
*****************************************************************************/

    dataStored1.psetpointEXP = &setpointEXP;
    dataStored1.pCBtripsignalEXP = &CBtripsignalEXP;

    HANDLE thread = CreateThread(NULL, 0, ThreadFunc, &dataStored1, 0, NULL);

    /* ********************************************************************** */

    error = IEC61850_Update(myServer, (struct IEC61850_DataAttributeID *)&
circuit_breakerID, &circuit_breaker, 1);
                if(error != IEC61850_ERROR_NONE)
                {
                    printf("Update failed: %s (%u)\n", IEC61850_ErrorString(error), error);

                }

    /* ***************************************************************************
        ***************************************************************************
*/
```

```
    char enter = 0;
    printf("Press enter to continue\n");

    while (enter != '\r' && enter != '\n')
    {
        enter = getchar();
    }

    //Shutting down the server
    printf("Closing server...\n");
    error = IEC61850_Stop (myServer);
    if(error != IEC61850_ERROR_NONE)
    {
        printf("Failed : %i,", error);

    }
    printf("\nServer is free\n");
    IEC61850_Free(myServer);

    printf("Final tap position is: %d", tap_position);

    return 0;
}

int myReadFunction (void *ptUserData, struct IEC61850_DataAttributeID *ptDataAttributeID,
struct IEC61850_DataAttributeData *ptReturnedValue)
{

    enum IEC61850_ErrorCodes error = IEC61850_ERROR_NONE;

    return error;
}

enum eCommandAddCause OperativeTestCallbackHandler(void * ptUserData, struct
IEC61850_DataAttributeID* ptControlID, struct IEC61850_CommandParameters*
ptOperativeTestParameters)
{
    enum eCommandAddCause eErrorCode = IEC61850_COMMAND_ERROR_NONE;
    /*Do Operative Test Code Here*/
    return eErrorCode;
}
int myOperateFunction (void *ptUserData, struct IEC61850_DataAttributeID *ptControlID,
const struct IEC61850_DataAttributeData *ptOperateValue, struct IEC61850_CommandParameters
 *ptOperateParameters)
{
    UserData myData;
    memcpy(&myData, ptUserData, sizeof(UserData));

    enum eCommandAddCause eErrorCode = IEC61850_COMMAND_ERROR_NONE; //Create return value
and init to No Error

    struct IEC61850_DataAttributeID_Generic *ptObjectData = NULL;
    enum IEC61850_ErrorCodes error = IEC61850_ERROR_NONE;

    if(ptControlID->daid_type == IEC61850_DAID_GENERIC)
    {

        ptObjectData = (struct IEC61850_DataAttributeID_Generic  *) ptControlID;

        if((ptObjectData->uiField1 == 5) && (ptObjectData->uiField2 ==0 ) && (ptObjectData
->uiField3 ==0) && (ptObjectData->uiField5 ==1))
        {
            struct IEC61850_DataAttributeData SetpointDataReceived;
            struct IEC61850_DataAttributeID_Generic SetpointDataIDReceived;

            SetpointDataReceived=dataStored1.dataObjects[14];
            SetpointDataIDReceived=dataStored1.dataObjectsID[14];

            printf("Setpoint value has been received %f\n", *(Float32*)(ptOperateValue->
pvData));
            memcpy(&setpointEXP, ptOperateValue->pvData, sizeof(Float32));


            error = IEC61850_Update(myServer, (struct IEC61850_DataAttributeID *)&
SetpointDataIDReceived, &SetpointDataReceived, 1);
```

```
            if(error != IEC61850_ERROR_NONE)
            {
                printf("Update failed: %s (%u)\n", IEC61850_ErrorString(error), error);

            }

            printf("Setpoint value has been updated: %f\n", setpointEXP);

        }
        if((ptObjectData->uiField1 == 3) && (ptObjectData->uiField2 ==0 ) && (ptObjectData
->uiField3 ==0) && (ptObjectData->uiField5 ==1))
        {

            printf("Circuit breaker control value has been received %d\n", *(Boolean*)(
ptOperateValue->pvData));

            struct IEC61850_DataAttributeData CBpositionData;
            struct IEC61850_DataAttributeID_Generic CBpositionDataID;


            CBpositionData=dataStored1.dataObjects[12];
            CBpositionDataID=dataStored1.dataObjectsID[12];


            if (*(Boolean*)(ptOperateValue->pvData)==0)
            {
                usiposition_CB = IEC61850_DB_POS_TRUE;
                CBtripsignalEXP = 0;
                error = IEC61850_Update(myServer, (struct IEC61850_DataAttributeID *)&
CBpositionDataID, &CBpositionData, 1);
                if(error != IEC61850_ERROR_NONE)
                {
                    printf("Update failed: %s (%u)\n", IEC61850_ErrorString(error), error);
                }

                printf("Circuit Breaker position been updated: %d\n", usiposition_CB);
            }
            else
            {
                CBtripsignalEXP =1;
                usiposition_CB = IEC61850_DB_POS_FALSE;
                error = IEC61850_Update(myServer, (struct IEC61850_DataAttributeID *)&
CBpositionDataID, &CBpositionData, 1);
                if(error != IEC61850_ERROR_NONE)
                {
                    printf("Update failed: %s (%u)\n", IEC61850_ErrorString(error), error);
                }

                printf("Circuit Breaker position been updated: %d\n", usiposition_CB);
            }

            /*
*********************************************************************************
            quality and time data and data IDs */

            //For quality

            unsigned short int usiObjectQuality;
            usiObjectQuality= (IEC61850_QUALITY_GOOD);

            struct IEC61850_DataAttributeData qualityDatatemp;

            qualityDatatemp.ucType = IEC61850_DATATYPE_QUALITY;
            qualityDatatemp.uiBitLength = IEC61850_QUALITY_BITSIZE;
            qualityDatatemp.iArrayIndex = -1;
            qualityDatatemp.pvData = &usiObjectQuality;

            CBpositionDataID.uiField3 = 1;

            error = IEC61850_Update(myServer, (struct IEC61850_DataAttributeID *)&
CBpositionDataID, &qualityDatatemp, 1);
            if(error != IEC61850_ERROR_NONE)
            {
                printf("Update failed: %s (%u)\n", IEC61850_ErrorString(error), error);
                printf("Generic fields: %d:%d:%d:%d:%d \n", CBpositionDataID.uiField1,
```

```
CBpositionDataID.uiField2, CBpositionDataID.uiField3, CBpositionDataID.uiField4,
CBpositionDataID.uiField5);

            }

            //For time

            struct IEC61850_TimeStamp  tTime;

            struct IEC61850_DataAttributeData timeDatatemp;

            timeDatatemp.ucType = IEC61850_DATATYPE_TIMESTAMP;
            timeDatatemp.uiBitLength = sizeof(tTime)*8;
            timeDatatemp.iArrayIndex = -1;
            timeDatatemp.pvData = &tTime;

            CBpositionDataID.uiField3 = 2;

        if(IEC61850_GetTime(myServer, &tTime) != IEC61850_ERROR_NONE)
        {
            printf("Get time failed: %s (%u)\n", IEC61850_ErrorString(error), error);
        }

            error = IEC61850_Update(myServer, (struct IEC61850_DataAttributeID *)&
CBpositionDataID, &timeDatatemp, 1);
            if(error != IEC61850_ERROR_NONE)
            {
                printf("Update failed: %s (%u)\n", IEC61850_ErrorString(error), error);
                printf("Generic fields: %d:%d:%d:%d:%d \n", CBpositionDataID.uiField1,
CBpositionDataID.uiField2, CBpositionDataID.uiField3, CBpositionDataID.uiField4,
CBpositionDataID.uiField5);
            }
        }
    }
    return eErrorCode;
}
```

9.2 Client application source code

```
/*
Source code of a client application.

Master thesis project:
"DISTRIBUTION OLTC CONTROL USING IEC 61850 CLIENT/SERVER ARCHITECTURE"
Author:
Andrius Maneikis

Communication stack API used:
SystemCORP Embedded Technology Pty Ltd PIS10 V2.06.28


*/

#include <stdio.h>
#include <stdlib.h>
#include <IEC61850API.h>
#include <gtk/gtk.h>
#include <windows.h> //to have multiple threads

#define SUPPORTED_PROTOCOL IEC61850_SYSTEMCOROP_GENERIC

#define measurementNumber 12 // number of current and voltage measurements
#define serverPollingTime 200 //in miliseconds

IEC61850 myClient;

/* *********************************Variables for server
data********************************** */
Integer8 tap_positionC;

Float32 voltage_HVa,  voltage_HVb,  voltage_HVc;
Float32 voltage_LVa,  voltage_LVb,  voltage_LVc;
Float32 current_HVa,  current_HVb,  current_HVc;
Float32 current_LVa,  current_LVb,  current_LVc;

Unsigned8 position_CBC; // circuit breaker position
Boolean position_CBctl; // circuit breaker control value

Float32 setpointC;
Boolean tripButton;

//Data to pass to callback functions
//Store data attributes and data attributes IDs
struct IEC61850_DataAttributeData measurementdataObjects[measurementNumber];
struct IEC61850_DataAttributeID_Generic measurementdataObjectsID[measurementNumber];


/*Function prototypes */
/* ************************************** */

enum eCommandAddCause OperativeTestCallbackHandler(void * ptUserData, struct
IEC61850_DataAttributeID* ptControlID, struct IEC61850_CommandParameters*
ptOperativeTestParameters);

void printValueToLabel (Float32 value, GtkWidget *label, Float32* newValue);

static gboolean changeCBpositionlabel(GtkWidget *label);
static gboolean setSwitchActive (gpointer switchdata);
static gboolean updateLabelTap (gpointer user_datalabel);
static gboolean updateLabelMeasurement (gpointer data);

void readDataFromServer(void* ptdata);

typedef struct
{

    GtkWidget *label1, *labelCB_position;

    GtkWidget *label_HV_a, *label_HV_b, *label_HV_c;
    GtkWidget *label_IHV_a, *label_IHV_b, *label_IHV_c;
    GtkWidget *label_LV_a, *label_LV_b, *label_LV_c;
    GtkWidget *label_ILV_a, *label_ILV_b, *label_ILV_c;

    GtkWidget *CB_position_switch;
```

```
    struct IEC61850_DataAttributeData dataObject1, dataObject;
    struct IEC61850_DataAttributeID_Generic dataObjectID1, dataObjectID;

} myClientUserdata;


typedef struct
{
    Float32 measureValue;
    GtkWidget *label;

} labelUserdata;


G_MODULE_EXPORT void Read_CBPosition_activate_cb (GtkWidget* widget, gpointer user_data)
{
    printf("reading....\n");

    myClientUserdata mydata;
    memcpy(&mydata, user_data, sizeof(myClientUserdata));

    GtkWidget *CBpositionlabel;
    CBpositionlabel = mydata.labelCB_position;

    enum IEC61850_ErrorCodes error = IEC61850_ERROR_NONE;
    struct IEC61850_DataAttributeData circuitbreaker_positionClient;
    circuitbreaker_positionClient.ucType = IEC61850_DATATYPE_DBPOS;
    circuitbreaker_positionClient.uiBitLength = sizeof(position_CBC)*8;
    circuitbreaker_positionClient.iArrayIndex = -1;
    circuitbreaker_positionClient.pvData = &position_CBC;

    struct IEC61850_DataAttributeID_Generic  circuitbreaker_positionClientID;
    circuitbreaker_positionClientID.Generic_type = IEC61850_DAID_GENERIC;
    circuitbreaker_positionClientID.uiField1  = 3;
    circuitbreaker_positionClientID.uiField2 = 0;
    circuitbreaker_positionClientID.uiField3 = 0;
    circuitbreaker_positionClientID.uiField4 = 0;
    circuitbreaker_positionClientID.uiField5 = 0;

    error = IEC61850_Read(myClient, (struct IEC61850_DataAttributeID *)&
circuitbreaker_positionClientID, &circuitbreaker_positionClient);
    if(error != IEC61850_ERROR_NONE)
    {
            printf("Read failed: %s (%u)\n", IEC61850_ErrorString(error), error);
    }
    else
    {
            printf("CB position = %d.\n", position_CBC);
    }

}

G_MODULE_EXPORT void trip_relay_signal (GtkWidget* widget, gpointer user_data)
{
    enum IEC61850_ErrorCodes error = IEC61850_ERROR_NONE;

    myClientUserdata mydata;
    memcpy(&mydata, user_data, sizeof(myClientUserdata));

    struct IEC61850_DataAttributeData dataObject;
    struct IEC61850_DataAttributeID_Generic dataObjectID;

    dataObject = mydata.dataObject1;
    dataObjectID = mydata.dataObjectID1;

    struct IEC61850_ControlParameters ControlParameters = {0};

    printf("Trip signal \n");
    position_CBctl = 1;
    tripButton = 1;

    printf("Data to trip: %d\n", *(Boolean*)dataObject.pvData );

    error = IEC61850_ControlOperate ( myClient, (struct IEC61850_DataAttributeID *) &
```

```
dataObjectID, &dataObject, ControlParameters);
    printf("Generic fields: %d:%d:%d:%d:%d \n", dataObjectID.uiField1, dataObjectID.
uiField2, dataObjectID.uiField3, dataObjectID.uiField4, dataObjectID.uiField5);

    if(error != IEC61850_ERROR_NONE)
    {
       printf("Operate failed: %s (%u)\n", IEC61850_ErrorString(error), error);
    }

}

G_MODULE_EXPORT void CB_switch_signal (GtkSwitch *switchbutton, GParamSpec *pspec,
gpointer user_data)
{
    enum IEC61850_ErrorCodes error = IEC61850_ERROR_NONE;

    myClientUserdata mydata;
    memcpy(&mydata, user_data, sizeof(myClientUserdata));

    struct IEC61850_DataAttributeData dataObject;
    struct IEC61850_DataAttributeID_Generic dataObjectID;

    dataObject = mydata.dataObject1;
    dataObjectID = mydata.dataObjectID1;

    struct IEC61850_ControlParameters ControlParameters = {0};

    if (gtk_switch_get_active (switchbutton))
    {
    printf("Switch is on \n");
    position_CBctl = 0;

    printf("Data to operate: %d\n", *(Boolean*)dataObject.pvData );

    error = IEC61850_ControlOperate ( myClient, (struct IEC61850_DataAttributeID *) &
dataObjectID, &dataObject, ControlParameters);
    printf("Generic fields: %d:%d:%d:%d:%d \n", dataObjectID.uiField1, dataObjectID.
uiField2, dataObjectID.uiField3, dataObjectID.uiField4, dataObjectID.uiField5);

    if(error != IEC61850_ERROR_NONE)
    {
       printf("Operate failed: %s (%u)\n", IEC61850_ErrorString(error), error);
    }

    printf ("Circuit breaker control position %d\n", position_CBctl );
    }
    else
    {
    printf("Switch is off \n");
    position_CBctl = 1;
    printf("Data to operate: %d\n", *(Boolean*)dataObject.pvData );
    error = IEC61850_ControlOperate ( myClient, (struct IEC61850_DataAttributeID *) &
dataObjectID, &dataObject, ControlParameters);
    printf("Generic fields: %d:%d:%d:%d:%d \n", dataObjectID.uiField1, dataObjectID.
uiField2, dataObjectID.uiField3, dataObjectID.uiField4, dataObjectID.uiField5);

    if(error != IEC61850_ERROR_NONE)
    {
       printf("Operate failed: %s (%u)\n", IEC61850_ErrorString(error), error);
    }

    printf ("Circuit breaker control position %d\n", position_CBctl );
    }
}


G_MODULE_EXPORT void grab_double_from_spinbutton (GtkWidget* spin_button, gpointer
user_data)
{
    enum IEC61850_ErrorCodes error = IEC61850_ERROR_NONE;

    setpointC = (Float32) gtk_spin_button_get_value (GTK_SPIN_BUTTON(spin_button));

    myClientUserdata mydata;
    memcpy(&mydata, user_data, sizeof(myClientUserdata));
```

```
    struct IEC61850_DataAttributeData dataObject;
    struct IEC61850_DataAttributeID_Generic dataObjectID;

    dataObject = mydata.dataObject;
    dataObjectID = mydata.dataObjectID;

    struct IEC61850_ControlParameters ControlParameters = {0};

    printf("Data to operate: %f\n", *(Float32*)dataObject.pvData );
    printf("dataObject.pvData: %p\n", dataObject.pvData );

    error = IEC61850_ControlOperate ( myClient, (struct IEC61850_DataAttributeID *) &
dataObjectID, &dataObject, ControlParameters);
    printf("Generic fields: %d:%d:%d:%d:%d \n", dataObjectID.uiField1, dataObjectID.
uiField2, dataObjectID.uiField3, dataObjectID.uiField4, dataObjectID.uiField5);

    if(error != IEC61850_ERROR_NONE)
    {
       printf("Operate failed: %s (%u)\n", IEC61850_ErrorString(error), error);
    }

    printf ("%f\n", setpointC );
}

G_MODULE_EXPORT void quit_application (GtkWidget* widget, gpointer user_data)
{
    enum IEC61850_ErrorCodes error = IEC61850_ERROR_NONE;

    //Shutting down the client
    printf("Closing client...\n");
    error = IEC61850_Stop (myClient);
    if(error != IEC61850_ERROR_NONE)
    {
        printf("Failed : %i,", error);

    }
    printf("\nClient is free\n");
    IEC61850_Free(myClient);

    gtk_main_quit ();
}

G_MODULE_EXPORT void on_about1_activate (GtkWidget* widget, gpointer window){

    GtkWidget *label, *dialog, *content_area;
    GtkDialogFlags flags;

    dialog = gtk_dialog_new_with_buttons("Crediantials", GTK_WINDOW(window),
GTK_DIALOG_MODAL, GTK_STOCK_OK, GTK_RESPONSE_OK, NULL);
    content_area = gtk_dialog_get_content_area (GTK_DIALOG (dialog));
    label = gtk_label_new("\tMaster Thesis project @ KTH:\t\t\n \tWritten by Andrius
Maneikis\t\n\t\tmaneikis@kth.se");

    g_signal_connect_swapped (dialog,"response", G_CALLBACK (gtk_widget_destroy), dialog);

    gtk_container_add (GTK_CONTAINER (content_area), label);
    gtk_widget_show_all (dialog);

}

int main(int argc, char *argv[])
{
    GtkBuilder *builder;
    GtkWidget *window, *spin_button, *CB_position_switch;
    GtkWidget *labelTap, *labelCB_position;

    GtkWidget *label_HV_a, *label_HV_b, *label_HV_c;
    GtkWidget *label_IHV_a, *label_IHV_b, *label_IHV_c;
    GtkWidget *label_LV_a, *label_LV_b, *label_LV_c;
    GtkWidget *label_ILV_a, *label_ILV_b, *label_ILV_c;

    GtkAdjustment  *adj;

    enum IEC61850_ErrorCodes error = IEC61850_ERROR_NONE;
```

```
//Circuit breaker position
// position_CBC = 128;
struct IEC61850_DataAttributeData circuitbreaker_positionClient;
circuitbreaker_positionClient.ucType = IEC61850_DATATYPE_DBPOS;
circuitbreaker_positionClient.uiBitLength = sizeof(position_CBC)*8;
circuitbreaker_positionClient.iArrayIndex = -1;
circuitbreaker_positionClient.pvData = &position_CBC;

struct IEC61850_DataAttributeID_Generic  circuitbreaker_positionClientID;
circuitbreaker_positionClientID.Generic_type = IEC61850_DAID_GENERIC;
circuitbreaker_positionClientID.uiField1  = 3;
circuitbreaker_positionClientID.uiField2 = 0;
circuitbreaker_positionClientID.uiField3 = 0;
circuitbreaker_positionClientID.uiField4 = 0;
circuitbreaker_positionClientID.uiField5 = 0;

struct IEC61850_DataAttributeData circuitbreaker_positionClientControl;
circuitbreaker_positionClientControl.ucType = IEC61850_DATATYPE_BOOLEAN;
circuitbreaker_positionClientControl.uiBitLength = sizeof(position_CBctl)*8;
circuitbreaker_positionClientControl.iArrayIndex = -1;
circuitbreaker_positionClientControl.pvData = &position_CBctl;

struct IEC61850_DataAttributeID_Generic  circuitbreaker_positionClientControlID;
circuitbreaker_positionClientControlID.Generic_type = IEC61850_DAID_GENERIC;
circuitbreaker_positionClientControlID.uiField1  = 3;
circuitbreaker_positionClientControlID.uiField2 = 0;
circuitbreaker_positionClientControlID.uiField3 = 0;
circuitbreaker_positionClientControlID.uiField4 = 0;
circuitbreaker_positionClientControlID.uiField5 = 1;

//Voltage setpoint

struct IEC61850_DataAttributeData voltage_setpointClient;
voltage_setpointClient.ucType = IEC61850_DATATYPE_FLOAT32;
voltage_setpointClient.uiBitLength = sizeof(setpointC)*8;
voltage_setpointClient.iArrayIndex = -1;
voltage_setpointClient.pvData = &setpointC;

printf("voltage_setpointClient.pvData: %p\n", voltage_setpointClient.pvData );

struct IEC61850_DataAttributeID_Generic  voltage_setpointClientID;
voltage_setpointClientID.Generic_type = IEC61850_DAID_GENERIC;
voltage_setpointClientID.uiField1  = 5;
voltage_setpointClientID.uiField2 = 0;
voltage_setpointClientID.uiField3 = 0;
voltage_setpointClientID.uiField4 = 0;
voltage_setpointClientID.uiField5 = 1;

gtk_init (&argc, &argv);

const char *builder_file = "GUI\\client 0.2.glade";

builder = gtk_builder_new ();

gtk_builder_add_from_file (builder, builder_file, NULL);

window = GTK_WIDGET (gtk_builder_get_object (builder, "main_window"));
spin_button = GTK_WIDGET (gtk_builder_get_object (builder, "setpoint_spinbutton1"));

labelTap = GTK_WIDGET (gtk_builder_get_object(builder, "tap_position_value_label"));
labelCB_position = GTK_WIDGET (gtk_builder_get_object(builder, "CB_position"));

label_HV_a = GTK_WIDGET (gtk_builder_get_object(builder, "label_HV_a"));
label_HV_b = GTK_WIDGET (gtk_builder_get_object(builder, "label_HV_b"));
label_HV_c = GTK_WIDGET (gtk_builder_get_object(builder, "label_HV_c"));
label_IHV_a = GTK_WIDGET (gtk_builder_get_object(builder, "label_IHV_a"));
label_IHV_b = GTK_WIDGET (gtk_builder_get_object(builder, "label_IHV_b"));
label_IHV_c = GTK_WIDGET (gtk_builder_get_object(builder, "label_IHV_c"));
label_LV_a = GTK_WIDGET (gtk_builder_get_object(builder, "label_LV_a"));
label_LV_b = GTK_WIDGET (gtk_builder_get_object(builder, "label_LV_b"));
label_LV_c = GTK_WIDGET (gtk_builder_get_object(builder, "label_LV_c"));
label_ILV_a = GTK_WIDGET (gtk_builder_get_object(builder, "label_ILV_a"));
label_ILV_b = GTK_WIDGET (gtk_builder_get_object(builder, "label_ILV_b"));
label_ILV_c = GTK_WIDGET (gtk_builder_get_object(builder, "label_ILV_c"));
```

```
    CB_position_switch =  GTK_WIDGET (gtk_builder_get_object(builder, "CB_position_switch"
));

    adj=gtk_adjustment_new (1, 0, 2, 0.01, 0, 0); //(value, lower, upper, step_increment,
page_increment, page_size)
    gtk_spin_button_set_adjustment ( GTK_SPIN_BUTTON(spin_button), adj );

    //Client user data

    myClientUserdata userdata;

    userdata.label1 = labelTap;
    userdata.labelCB_position = labelCB_position;

    userdata.label_HV_a = label_HV_a;
    userdata.label_HV_b  = label_HV_b;
    userdata.label_HV_c  = label_HV_c;
    userdata.label_IHV_a  = label_IHV_a;
    userdata.label_IHV_b  = label_IHV_b;
    userdata.label_IHV_c  = label_IHV_c;
    userdata.label_LV_a = label_LV_a;
    userdata.label_LV_b = label_LV_b;
    userdata.label_LV_c = label_LV_c;
    userdata.label_ILV_a = label_ILV_a;
    userdata.label_ILV_b = label_ILV_b;
    userdata.label_ILV_c = label_ILV_c;

    userdata.CB_position_switch = CB_position_switch;
    userdata.dataObject = voltage_setpointClient;
    userdata.dataObjectID = voltage_setpointClientID;
    userdata.dataObject1 = circuitbreaker_positionClientControl;
    userdata.dataObjectID1 = circuitbreaker_positionClientControlID;

    gtk_builder_connect_signals (builder, &userdata);


    /* ****************************************************************************
*/

    char filepathClient[]="files\\client.cid";

    printf("File path for the CLIENT ICD file: \n%s\n\n", filepathClient);


    /* Client
*********************************************************************************************
****************** */
    //Create Client

    struct IEC61850_Parameters tCientParam = {0};
    memset(&tCientParam, 0, sizeof(struct IEC61850_Parameters));

    do
    {
        tCientParam.ClientServerFlag = IEC61850_CLIENT;
        tCientParam.Ed1_Ed2_Flag=IEC61850_Edition2;
        tCientParam.ptReadCallback = NULL;
        tCientParam.ptWriteCallback = NULL;
        tCientParam.ptUpdateCallback = NULL;
        tCientParam.ptOprTestCallback = OperativeTestCallbackHandler;

        myClient = IEC61850_Create(&tCientParam, &error);
        if (myClient == NULL)
        {
            printf(" Failed : %i, \t%s", error, IEC61850_ErrorString(error));
            break;
        }

        //Load SCL configuration file of the client
        printf("\r\n Client load SCL");
        error = IEC61850_LoadSCLFile(myClient, filepathClient); //Load ICD file
        if (error != IEC61850_ERROR_NONE)
        {
            printf(" Failed : %i, \t%s", error, IEC61850_ErrorString(error));
```

```
            break;
        }

        //Starting client
        printf("\r\n Client Start\n");
        error = IEC61850_Start(myClient);
        if(error != IEC61850_ERROR_NONE)
        {
            printf( "Failed : %i/t%s", error, IEC61850_ErrorString(error));
            break;
        }
    }
    while(0);

    printf("\r\n Client started.\n");

    /* ********************************************************************** */
    //Get initial circuit breaker position

    error = IEC61850_Read(myClient, (struct IEC61850_DataAttributeID *)&
circuitbreaker_positionClientID, &circuitbreaker_positionClient);
    if(error != IEC61850_ERROR_NONE)
    {
              printf("Read failed: %s (%u)\n", IEC61850_ErrorString(error), error);
              printf("Generic fields: %d:%d:%d:%d:%d \n", circuitbreaker_positionClientID.
uiField1,
                      circuitbreaker_positionClientID.uiField2,
circuitbreaker_positionClientID.uiField3,
                      circuitbreaker_positionClientID.uiField4,
circuitbreaker_positionClientID.uiField5);
    }
    else
    {
              printf("CB position = %d.\n", *(INT8*)circuitbreaker_positionClient.pvData);
    }

    error = IEC61850_SetUserData(myClient, &userdata); //provides user data for the
callback functions

    HANDLE thread = CreateThread(NULL, 0, readDataFromServer, &userdata, 0, NULL);

    /* Destroy builder, since we don't need it anymore */
    g_object_unref( G_OBJECT( builder ) );

    gtk_widget_show (window);
    gtk_main ();

    return 0;
}

enum eCommandAddCause OperativeTestCallbackHandler(void * ptUserData, struct
IEC61850_DataAttributeID* ptControlID, struct IEC61850_CommandParameters*
ptOperativeTestParameters)
{
    enum eCommandAddCause eErrorCode = IEC61850_COMMAND_ERROR_NONE;
    /*Do Operative Test Code Here*/
    return eErrorCode;
}

void printValueToLabel (Float32 value, GtkWidget *label, Float32* newValue){

    char tap_label_text[50];

    memcpy(&value, newValue, sizeof(Float32));

    snprintf(tap_label_text ,50,"%f", value);

    gtk_label_set_text ( GTK_LABEL (label), tap_label_text);

}

static gboolean changeCBpositionlabel(GtkWidget *label){

    if (position_CBC == 64 || position_CBC == -64){
```

```
        gtk_label_set_text ( GTK_LABEL (label), "CB position is open");

    } else if (position_CBC == 128 || position_CBC == -128){

        gtk_label_set_text ( GTK_LABEL (label), "CB position is close");

    } else {

        gtk_label_set_text ( GTK_LABEL (label), "CB position is unknown");
    }
    return G_SOURCE_REMOVE;
}
void readDataFromServer (void* ptdata){

    printf("Reading in another thread\n");

    myClientUserdata mydata;
    memcpy(&mydata, ptdata, sizeof(myClientUserdata));

    GtkWidget *label1, *labelCB_position;

    GtkWidget *label_HV_a, *label_HV_b, *label_HV_c;
    GtkWidget *label_IHV_a, *label_IHV_b, *label_IHV_c;
    GtkWidget *label_LV_a, *label_LV_b, *label_LV_c;
    GtkWidget *label_ILV_a, *label_ILV_b, *label_ILV_c;

    GtkSwitch *switchPos;

    switchPos = mydata.CB_position_switch;

    label1 = mydata.label1;
    labelCB_position = mydata.labelCB_position;

    label_HV_a = mydata.label_HV_a;
    label_HV_b = mydata.label_HV_b;
    label_HV_c = mydata.label_HV_c;
    label_IHV_a = mydata.label_IHV_a;
    label_IHV_b = mydata.label_IHV_b;
    label_IHV_c = mydata.label_IHV_c;
    label_LV_a = mydata.label_LV_a;
    label_LV_b = mydata.label_LV_b;
    label_LV_c = mydata.label_LV_c;
    label_ILV_a = mydata.label_ILV_a;
    label_ILV_b = mydata.label_ILV_b;
    label_ILV_c = mydata.label_ILV_c;

    GtkWidget* measurementLabel[] = {label_HV_a, label_HV_b, label_HV_c, label_IHV_a,
label_IHV_b, label_IHV_c,
    label_LV_a, label_LV_b, label_LV_c, label_ILV_a, label_ILV_b, label_ILV_c};

    Float32 newValue;
    labelUserdata labelUpdate;

    enum IEC61850_ErrorCodes error = IEC61850_ERROR_NONE;

    struct IEC61850_DataAttributeData tap;
    tap.ucType = IEC61850_DATATYPE_INT8;
    tap.uiBitLength = sizeof(tap_positionC)*8;
    tap.iArrayIndex = -1;
    tap.pvData = &tap_positionC;



    struct IEC61850_DataAttributeID_Generic  tapID;
    tapID.Generic_type = IEC61850_DAID_GENERIC;
    tapID.uiField1  = 4;
    tapID.uiField2 = 2;
    tapID.uiField3 = 0;
    tapID.uiField4 = 0;
    tapID.uiField5 = 0;

    struct IEC61850_DataAttributeData circuitbreaker_positionClient;
    circuitbreaker_positionClient.ucType = IEC61850_DATATYPE_DBPOS;
    circuitbreaker_positionClient.uiBitLength = sizeof(position_CBC)*8;
    circuitbreaker_positionClient.iArrayIndex = -1;
```

```
circuitbreaker_positionClient.pvData = &position_CBC;

struct IEC61850_DataAttributeID_Generic  circuitbreaker_positionClientID;
circuitbreaker_positionClientID.Generic_type = IEC61850_DAID_GENERIC;
circuitbreaker_positionClientID.uiField1  = 3;
circuitbreaker_positionClientID.uiField2 = 0;
circuitbreaker_positionClientID.uiField3 = 0;
circuitbreaker_positionClientID.uiField4 = 0;
circuitbreaker_positionClientID.uiField5 = 0;

//MMXU0 Voltage measurements
//Voltage measurement of HV side phase A
struct IEC61850_DataAttributeData voltage_HVaData;
voltage_HVaData.ucType = IEC61850_DATATYPE_FLOAT32;
voltage_HVaData.uiBitLength = sizeof(voltage_HVa)*8;
voltage_HVaData.iArrayIndex = -1;
voltage_HVaData.pvData = &voltage_HVa;

measurementdataObjects[0]=voltage_HVaData;

struct IEC61850_DataAttributeID_Generic  voltage_HVaDataID;
voltage_HVaDataID.Generic_type = IEC61850_DAID_GENERIC;
voltage_HVaDataID.uiField1  = 1;
voltage_HVaDataID.uiField2 = 1;
voltage_HVaDataID.uiField3 = 0;
voltage_HVaDataID.uiField4 = 0;
voltage_HVaDataID.uiField5 = 0;

measurementdataObjectsID[0]=voltage_HVaDataID;

//Voltage measurement of HV side phase B
struct IEC61850_DataAttributeData voltage_HVbData;
memcpy(&voltage_HVbData, &voltage_HVaData, sizeof(voltage_HVaData));
voltage_HVbData.pvData = &voltage_HVb;

measurementdataObjects[1]=voltage_HVbData;

struct IEC61850_DataAttributeID_Generic  voltage_HVbDataID;
memcpy(&voltage_HVbDataID, &voltage_HVaDataID, sizeof(voltage_HVaDataID));
voltage_HVbDataID.uiField5 = 1;

measurementdataObjectsID[1]=voltage_HVbDataID;

//Voltage measurement of HV side phase C
struct IEC61850_DataAttributeData voltage_HVcData;
memcpy(&voltage_HVcData, &voltage_HVaData, sizeof(voltage_HVaData));
voltage_HVcData.pvData = &voltage_HVc;

measurementdataObjects[2]=voltage_HVcData;

struct IEC61850_DataAttributeID_Generic  voltage_HVcDataID;
memcpy(&voltage_HVcDataID, &voltage_HVaDataID, sizeof(voltage_HVaDataID));
voltage_HVcDataID.uiField5 = 2;

measurementdataObjectsID[2]=voltage_HVcDataID;

//MMXU0 Current measurements
//Current measurement of HV side phase A
struct IEC61850_DataAttributeData current_HVaData;
memcpy(&current_HVaData, &voltage_HVaData, sizeof(voltage_HVaData));
current_HVaData.pvData = &current_HVa;

measurementdataObjects[3]=current_HVaData;

struct IEC61850_DataAttributeID_Generic  current_HVaDataID;
memcpy(&current_HVaDataID, &voltage_HVaDataID, sizeof(voltage_HVaDataID));
current_HVaDataID.uiField2 = 2;


measurementdataObjectsID[3]=current_HVaDataID;

//Current measurement of HV side phase B
struct IEC61850_DataAttributeData current_HVbData;
memcpy(&current_HVbData, &voltage_HVaData, sizeof(voltage_HVaData));
current_HVbData.pvData = &current_HVb;
```

```
measurementdataObjects[4]=current_HVbData;

struct IEC61850_DataAttributeID_Generic  current_HVbDataID;
memcpy(&current_HVbDataID, &voltage_HVbDataID, sizeof(voltage_HVbDataID));
current_HVbDataID.uiField2 = 2;


measurementdataObjectsID[4]=current_HVbDataID;

//Current measurement of HV side phase C
struct IEC61850_DataAttributeData current_HVcData;
memcpy(&current_HVcData, &voltage_HVaData, sizeof(voltage_HVaData));
current_HVcData.pvData = &current_HVc;

measurementdataObjects[5]=current_HVcData;

struct IEC61850_DataAttributeID_Generic  current_HVcDataID;
memcpy(&current_HVcDataID, &voltage_HVcDataID, sizeof(voltage_HVcDataID));
current_HVcDataID.uiField2 = 2;

measurementdataObjectsID[5]=current_HVcDataID;

//MMXU1 Voltage measurements
//Voltage measurement of LV side phase A
struct IEC61850_DataAttributeData voltage_LVaData;
memcpy(&voltage_LVaData, &voltage_HVaData, sizeof(voltage_HVaData));
voltage_LVaData.pvData = &voltage_LVa;

measurementdataObjects[6]=voltage_LVaData;

struct IEC61850_DataAttributeID_Generic  voltage_LVaDataID;
memcpy(&voltage_LVaDataID, &voltage_HVaDataID, sizeof(voltage_HVaDataID));
voltage_LVaDataID.uiField1 = 2;

measurementdataObjectsID[6]=voltage_LVaDataID;

//Voltage measurement of LV side phase B
struct IEC61850_DataAttributeData voltage_LVbData;
memcpy(&voltage_LVbData, &voltage_HVaData, sizeof(voltage_HVaData));
voltage_LVbData.pvData = &voltage_LVb;

measurementdataObjects[7]=voltage_LVbData;

struct IEC61850_DataAttributeID_Generic  voltage_LVbDataID;
memcpy(&voltage_LVbDataID, &voltage_HVbDataID, sizeof(voltage_HVbDataID));
voltage_LVbDataID.uiField1 = 2;

measurementdataObjectsID[7]=voltage_LVbDataID;

//Voltage measurement of LV side phase C
struct IEC61850_DataAttributeData voltage_LVcData;
memcpy(&voltage_LVcData, &voltage_HVaData, sizeof(voltage_HVaData));
voltage_LVcData.pvData = &voltage_LVc;

measurementdataObjects[8]=voltage_LVcData;

struct IEC61850_DataAttributeID_Generic  voltage_LVcDataID;
memcpy(&voltage_LVcDataID, &voltage_HVcDataID, sizeof(voltage_HVcDataID));
voltage_LVcDataID.uiField1 = 2;

measurementdataObjectsID[8]=voltage_LVcDataID;

//MMXU1 Current measurements
//Current measurement of LV side phase A
struct IEC61850_DataAttributeData current_LVaData;
memcpy(&current_LVaData, &voltage_HVaData, sizeof(voltage_HVaData));
current_LVaData.pvData = &current_LVa;

measurementdataObjects[9]=current_LVaData;

struct IEC61850_DataAttributeID_Generic  current_LVaDataID;
memcpy(&current_LVaDataID, &current_HVaDataID, sizeof(current_HVaDataID));
current_LVaDataID.uiField1  = 2;
```

```
    measurementdataObjectsID[9]=current_LVaDataID;

    //Current measurement of LV side phase B
    struct IEC61850_DataAttributeData current_LVbData;
    memcpy(&current_LVbData, &voltage_HVaData, sizeof(voltage_HVaData));
    current_LVbData.pvData = &current_LVb;

    measurementdataObjects[10]=current_LVbData;

    struct IEC61850_DataAttributeID_Generic  current_LVbDataID;
    memcpy(&current_LVbDataID, &current_HVbDataID, sizeof(current_HVbDataID));
    current_LVbDataID.uiField1  = 2;


    measurementdataObjectsID[10]=current_LVbDataID;

    //Current measurement of LV side phase C
    struct IEC61850_DataAttributeData current_LVcData;
    memcpy(&current_LVcData, &voltage_HVaData, sizeof(voltage_HVaData));
    current_LVcData.pvData = &current_LVc;

    measurementdataObjects[11]=current_LVcData;

    struct IEC61850_DataAttributeID_Generic  current_LVcDataID;
    memcpy(&current_LVcDataID, &current_HVcDataID, sizeof(current_HVcDataID));
    current_LVcDataID.uiField1  = 2;

    measurementdataObjectsID[11]=current_LVcDataID;

    while (1){

    Sleep(serverPollingTime);

    error = IEC61850_Read(myClient, (struct IEC61850_DataAttributeID *)&tapID, &tap);
    if(error != IEC61850_ERROR_NONE)
    {
                printf("Tap read failed: %s (%u)\n", IEC61850_ErrorString(error), error);
    }
        gdk_threads_add_idle    (updateLabelTap, label1);


    error = IEC61850_Read(myClient, (struct IEC61850_DataAttributeID *)&
circuitbreaker_positionClientID, &circuitbreaker_positionClient);
    if(error != IEC61850_ERROR_NONE)
    {
                printf("Circuit breaker position read failed: %s (%u)\n",
IEC61850_ErrorString(error), error);
    }

    if (position_CBC == 64 && !gtk_switch_get_active(switchPos) && tripButton == 1){

        gdk_threads_add_idle(setSwitchActive, switchPos);
        tripButton = 0;

        }

    gdk_threads_add_idle   (changeCBpositionlabel, labelCB_position);

    int i;


        for (i = 0; i < measurementNumber; i++ ){
    error = IEC61850_Read(myClient, (struct IEC61850_DataAttributeID *)&
measurementdataObjectsID[i], &measurementdataObjects[i]);
        if(error != IEC61850_ERROR_NONE)
    {
                printf("Measurement read failed: %s (%u)\n", IEC61850_ErrorString(error),
error);
    }

    newValue = *(Float32*)measurementdataObjects[i].pvData;

    labelUpdate.measureValue =newValue;
    labelUpdate.label = measurementLabel[i];
```

```
    gdk_threads_add_idle    ( updateLabelMeasurement, &labelUpdate);

    }

    }//While(1) ends
}

static gboolean updateLabelTap (gpointer user_datalabel){

    char tap_label_text[50];

    snprintf(tap_label_text ,50, "%d", tap_positionC);

    gtk_label_set_text ( GTK_LABEL (user_datalabel), tap_label_text);
    return G_SOURCE_REMOVE;

}
static gboolean updateLabelMeasurement (gpointer data){

    labelUserdata mydata;

    memcpy(&mydata, data, sizeof(labelUserdata));

    char tap_label_text[50];

    snprintf(tap_label_text ,50, "%f", mydata.measureValue);

    gtk_label_set_text ( GTK_LABEL (mydata.label), tap_label_text);
    return G_SOURCE_REMOVE;

}
static gboolean setSwitchActive (gpointer switchdata){
        gtk_switch_set_active(switchdata, TRUE);
        return G_SOURCE_REMOVE;
}
```

9.3 Server CID file

```
<?xml version="1.0" encoding="UTF-8"?>
<SCL xmlns="http://www.iec.ch/61850/2003/SCL" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" revision="A" version="2007"
xsi:schemaLocation="http://www.iec.ch/61850/2003/SCL SCL.xsd">
   <Header id="0" version="0"/>
   <Communication>
      <SubNetwork name="SubNetworkName">
         <ConnectedAP apName="SubstationRing1" iedName="NewIED">
            <Address>
               <P type="OSI-AP-Title">1,1,9999,1</P>
               <P type="OSI-AE-Qualifier">12</P>
               <P type="OSI-PSEL">00000001</P>
               <P type="OSI-SSEL">0001</P>
               <P type="OSI-TSEL">0001</P>
               <P type="IP">130.229.188.193</P>
               <P type="IP-SUBNET">255.255.192.0</P>
               <P type="IP-GATEWAY">130.229.128.1</P>
               <P type="MAC-Address">5C-51-4F-12-61-23</P>
            </Address>
            <GSE cbName="GSE_CB" ldInst="LDevice1">
               <Address>
                  <P type="MAC-Address">01-0C-CD-01-00-01</P>
                  <P type="APPID">0000</P>
                  <P type="VLAN-PRIORITY">4</P>
                  <P type="VLAN-ID">000</P>
               </Address>
               <MinTime>20</MinTime>
               <MaxTime>4000</MaxTime>
            </GSE>
         </ConnectedAP>
      </SubNetwork>
   </Communication>
   <IED desc="NewIED" manufacturer="SystemCORP Pty Ltd" name="NewIED" type="RTUType">
      <Services>
         <GOOSE max="1"/>
      </Services>
      <AccessPoint name="SubstationRing1" router="false">
         <Server timeout="30">
            <Authentication/>
            <LDevice inst="LDevice1">
               <LN0 desc="Logical node zero" inst="" lnClass="LLN0" lnType="LLN0_0">
                  <DataSet name="GooseCBDataset">
                     <FCDA daName="stVal" doName="Pos" fc="ST" ldInst="LDevice1" lnClass="XCBR"
                     lnInst="0"/>
                     <FCDA daName="Oper.ctlVal" doName="Mod" fc="CO" ldInst="LDevice1" lnClass=
                     "XCBR" lnInst="0"/>
                  </DataSet>
                  <DOI desc="Controllable enumerated status" name="Mod">
                     <DAI name="stVal">
                        <Val>on</Val>
                     </DAI>
                  </DOI>
                  <DOI desc="Enumerated status" name="Beh">
                     <DAI name="stVal">
```

```
            <Val>on</Val>
         </DAI>
      </DOI>
      <DOI desc="Enumerated status" name="Health">
         <DAI name="stVal">
            <Val>Ok</Val>
         </DAI>
      </DOI>
      <GSEControl appID="GSE_ControlBlock" confRev="1" datSet="GooseCBDataset" name=
      "GSE_CB" type="GOOSE"/>
   </LN0>
   <LN desc="Physical device information" inst="0" lnClass="LPHD" lnType="LPHD_0"
   prefix=""/>
   <LN desc="Voltage control" inst="0" lnClass="AVCO" lnType="AVCO_0" prefix="">
      <DataSet name="VoltageSetPointDataset">
         <FCDA doName="SptVol" fc="MX" ldInst="LDevice1" lnClass="AVCO" lnInst="0"/>
         <FCDA doName="SptVol" fc="CF" ldInst="LDevice1" lnClass="AVCO" lnInst="0"/>
      </DataSet>
      <ReportControl confRev="1" datSet="VoltageSetPointDataset" intgPd="300000"
      name="Setpoint" rptID="AVCOID">
         <TrgOps dchg="true" dupd="true" qchg="true"/>
         <OptFields seqNum="true" timeStamp="true"/>
         <RptEnabled max="3"/>
      </ReportControl>
      <DOI desc="Enumerated status" name="Beh">
         <DAI name="stVal">
            <Val>on</Val>
         </DAI>
      </DOI>
      <DOI desc="Controllable analogue set point information" name="SptVol">
         <DAI name="q">
            <Private type="SystemCorp_Generic">
               <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
               "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="5" Field2=
               "0" Field3="1" Field4="0" Field5="0"/>
            </Private>
         </DAI>
         <DAI name="t">
            <Private type="SystemCorp_Generic">
               <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
               "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="5" Field2=
               "0" Field3="2" Field4="0" Field5="0"/>
            </Private>
         </DAI>
         <DAI name="ctlModel">
            <Val>direct-with-normal-security</Val>
         </DAI>
         <SDI name="Oper">
            <DAI name="ctlVal">
               <Private type="SystemCorp_Generic">
                  <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
                  "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="5" Field2
                  ="0" Field3="0" Field4="0" Field5="1"/>
               </Private>
```

```
            </DAI>
         </SDI>
         <SDI name="mxVal">
            <DAI name="f">
               <Private type="SystemCorp_Generic">
                  <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
                  "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="5" Field2
                  ="0" Field3="0" Field4="0" Field5="0"/>
               </Private>
            </DAI>
         </SDI>
      </DOI>
   </LN>
   <LN desc="Tap changer" inst="0" lnClass="YLTC" lnType="YLTC_0" prefix="">
      <DataSet name="TapPositionDataset">
         <FCDA doName="TapPos" fc="ST" ldInst="LDevice1" lnClass="YLTC" lnInst="0"/>
         <FCDA doName="TapPos" fc="CF" ldInst="LDevice1" lnClass="YLTC" lnInst="0"/>
      </DataSet>
      <ReportControl confRev="0" datSet="TapPositionDataset" intgPd="300000" name=
      "TapPosition" rptID="YLTCOID">
         <TrgOps dchg="true" dupd="true"/>
         <OptFields dataSet="true" seqNum="true" timeStamp="true"/>
         <RptEnabled max="3"/>
      </ReportControl>
      <DOI desc="Integer controlled step position information" name="TapPos">
         <DAI name="q">
            <Private type="SystemCorp_Generic">
               <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
               "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="4" Field2=
               "2" Field3="1" Field4="0" Field5="0"/>
            </Private>
         </DAI>
         <DAI name="t">
            <Private type="SystemCorp_Generic">
               <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
               "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="4" Field2=
               "2" Field3="2" Field4="0" Field5="0"/>
            </Private>
         </DAI>
         <DAI name="ctlModel">
            <Val>status-only</Val>
         </DAI>
         <SDI name="valWTr">
            <DAI name="posVal">
               <Private type="SystemCorp_Generic">
                  <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
                  "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="4" Field2
                  ="2" Field3="0" Field4="0" Field5="0"/>
               </Private>
            </DAI>
         </SDI>
      </DOI>
      <DOI desc="Single point status" name="EndPosR">
         <DAI name="stVal">
```

```
            <Private type="SystemCorp_Generic">
               <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
               "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="4" Field2=
               "0" Field3="0" Field4="0" Field5="0"/>
            </Private>
         </DAI>
         <DAI name="q">
            <Private type="SystemCorp_Generic">
               <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
               "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="4" Field2=
               "0" Field3="1" Field4="0" Field5="0"/>
            </Private>
         </DAI>
         <DAI name="t">
            <Private type="SystemCorp_Generic">
               <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
               "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="4" Field2=
               "0" Field3="2" Field4="0" Field5="0"/>
            </Private>
         </DAI>
      </DOI>
      <DOI desc="Single point status" name="EndPosL">
         <DAI name="stVal">
            <Private type="SystemCorp_Generic">
               <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
               "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="4" Field2=
               "1" Field3="0" Field4="0" Field5="0"/>
            </Private>
         </DAI>
         <DAI name="q">
            <Private type="SystemCorp_Generic">
               <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
               "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="4" Field2=
               "1" Field3="1" Field4="0" Field5="0"/>
            </Private>
         </DAI>
         <DAI name="t">
            <Private type="SystemCorp_Generic">
               <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
               "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="4" Field2=
               "1" Field3="2" Field4="0" Field5="0"/>
            </Private>
         </DAI>
      </DOI>
   </LN>
   <LN desc="Circuit breaker" inst="0" lnClass="XCBR" lnType="XCBR_0" prefix="">
      <DataSet name="CBDataset">
         <FCDA doName="Mod" fc="ST" ldInst="LDevice1" lnClass="XCBR" lnInst="0"/>
         <FCDA doName="Pos" fc="ST" ldInst="LDevice1" lnClass="XCBR" lnInst="0"/>
         <FCDA doName="Mod" fc="CO" ldInst="LDevice1" lnClass="XCBR" lnInst="0"/>
         <FCDA doName="Mod" fc="CF" ldInst="LDevice1" lnClass="XCBR" lnInst="0"/>
         <FCDA doName="Pos" fc="CF" ldInst="LDevice1" lnClass="XCBR" lnInst="0"/>
      </DataSet>
      <DataSet name="DataSet01">
```

```
      <FCDA daName="stVal" doName="Pos" fc="ST" ldInst="LDevice1" lnClass="XCBR"
      lnInst="0"/>
   </DataSet>
   <ReportControl confRev="0" datSet="CBDataset" intgPd="30000" name=
   "CircuitBreaker" rptID="XCBRID">
      <TrgOps dchg="true" dupd="true"/>
      <OptFields dataRef="true" dataSet="true" reasonCode="true" seqNum="true"
      timeStamp="true"/>
      <RptEnabled max="3"/>
   </ReportControl>
   <DOI desc="Controllable enumerated status" name="Mod">
      <DAI name="stVal">
         <Private type="SystemCorp_Generic">
            <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
            "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="3" Field2=
            "1" Field3="0" Field4="0" Field5="0"/>
         </Private>
      </DAI>
      <DAI name="ctlModel">
         <Val>direct-with-normal-security</Val>
      </DAI>
      <SDI name="Oper">
         <DAI name="ctlVal">
            <Private type="SystemCorp_Generic">
               <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
               "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="3" Field2
               ="0" Field3="0" Field4="0" Field5="1"/>
            </Private>
         </DAI>
      </SDI>
   </DOI>
   <DOI desc="Controllable double point" name="Pos">
      <DAI name="stVal">
         <Private type="SystemCorp_Generic">
            <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
            "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="3" Field2=
            "0" Field3="0" Field4="0" Field5="0"/>
         </Private>
      </DAI>
      <DAI name="q">
         <Private type="SystemCorp_Generic">
            <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
            "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="3" Field2=
            "0" Field3="1" Field4="0" Field5="0"/>
         </Private>
      </DAI>
      <DAI name="t">
         <Private type="SystemCorp_Generic">
            <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
            "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="3" Field2=
            "0" Field3="2" Field4="0" Field5="0"/>
         </Private>
      </DAI>
      <DAI name="ctlModel">
```

```
            <Val>direct-with-normal-security</Val>
         </DAI>
      </DOI>
   </LN>
   <LN desc="Measurement" inst="0" lnClass="MMXU" lnType="MMXU_0" prefix="">
      <DataSet name="CurrentDataset">
         <FCDA doName="A" fc="MX" ldInst="LDevice1" lnClass="MMXU" lnInst="0"/>
      </DataSet>
      <DataSet name="VoltageDataset">
         <FCDA doName="PhV" fc="MX" ldInst="LDevice1" lnClass="MMXU" lnInst="0"/>
      </DataSet>
      <ReportControl confRev="0" datSet="VoltageDataset" intgPd="300000" name=
      "Voltage" rptID="MMXUVID">
         <TrgOps dchg="true" dupd="true"/>
         <OptFields dataSet="true" seqNum="true" timeStamp="true"/>
         <RptEnabled max="3"/>
      </ReportControl>
      <ReportControl confRev="0" datSet="CurrentDataset" intgPd="30000" name=
      "Current" rptID="MMXUIID">
         <TrgOps dchg="true" dupd="true"/>
         <OptFields dataSet="true" reasonCode="true" seqNum="true" timeStamp="true"/>
         <RptEnabled max="3"/>
      </ReportControl>
      <DOI desc="Enumerated status" name="Beh">
         <DAI name="stVal">
            <Val>on</Val>
         </DAI>
      </DOI>
      <DOI desc="Phase to ground related measurement values of a three phase Star"
      name="PhV">
         <SDI name="phsA">
            <SDI name="cVal">
               <SDI name="mag">
                  <DAI name="f">
                     <Private type="SystemCorp_Generic">
                        <SystemCorp_Generic:GenericPrivateObject
                        xmlns:SystemCorp_Generic=
                        "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="1"
                        Field2="1" Field3="0" Field4="0" Field5="0"/>
                     </Private>
                  </DAI>
               </SDI>
            </SDI>
            <DAI name="q">
               <Private type="SystemCorp_Generic">
                  <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
                  "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="1" Field2
                  ="1" Field3="1" Field4="0" Field5="0"/>
               </Private>
            </DAI>
            <DAI name="t">
               <Private type="SystemCorp_Generic">
                  <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
                  "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="1" Field2
```

```
            ="1" Field3="2" Field4="0" Field5="0"/>
         </Private>
      </DAI>
   </SDI>
   <SDI name="phsB">
      <SDI name="cVal">
         <SDI name="mag">
            <DAI name="f">
               <Private type="SystemCorp_Generic">
                  <SystemCorp_Generic:GenericPrivateObject
                  xmlns:SystemCorp_Generic=
                  "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="1"
                  Field2="1" Field3="0" Field4="0" Field5="1"/>
               </Private>
            </DAI>
         </SDI>
      </SDI>
      <DAI name="q">
         <Private type="SystemCorp_Generic">
            <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
            "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="1" Field2
            ="1" Field3="1" Field4="0" Field5="1"/>
         </Private>
      </DAI>
      <DAI name="t">
         <Private type="SystemCorp_Generic">
            <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
            "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="1" Field2
            ="1" Field3="2" Field4="0" Field5="1"/>
         </Private>
      </DAI>
   </SDI>
   <SDI name="phsC">
      <SDI name="cVal">
         <SDI name="mag">
            <DAI name="f">
               <Private type="SystemCorp_Generic">
                  <SystemCorp_Generic:GenericPrivateObject
                  xmlns:SystemCorp_Generic=
                  "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="1"
                  Field2="1" Field3="0" Field4="0" Field5="2"/>
               </Private>
            </DAI>
         </SDI>
      </SDI>
      <DAI name="q">
         <Private type="SystemCorp_Generic">
            <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
            "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="1" Field2
            ="1" Field3="1" Field4="0" Field5="2"/>
         </Private>
      </DAI>
      <DAI name="t">
         <Private type="SystemCorp_Generic">
```

```
                    <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
                    "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="1" Field2
                    ="1" Field3="2" Field4="0" Field5="2"/>
                 </Private>
              </DAI>
           </SDI>
        </DOI>
        <DOI desc="Phase to ground related measurement values of a three phase Star"
        name="A">
           <SDI name="phsA">
              <SDI name="cVal">
                 <SDI name="mag">
                    <DAI name="f">
                       <Private type="SystemCorp_Generic">
                          <SystemCorp_Generic:GenericPrivateObject
                          xmlns:SystemCorp_Generic=
                          "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="1"
                          Field2="2" Field3="0" Field4="0" Field5="0"/>
                       </Private>
                    </DAI>
                 </SDI>
              </SDI>
              <DAI name="q">
                 <Private type="SystemCorp_Generic">
                    <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
                    "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="1" Field2
                    ="2" Field3="1" Field4="0" Field5="0"/>
                 </Private>
              </DAI>
              <DAI name="t">
                 <Private type="SystemCorp_Generic">
                    <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
                    "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="1" Field2
                    ="2" Field3="2" Field4="0" Field5="0"/>
                 </Private>
              </DAI>
           </SDI>
           <SDI name="phsB">
              <SDI name="cVal">
                 <SDI name="mag">
                    <DAI name="f">
                       <Private type="SystemCorp_Generic">
                          <SystemCorp_Generic:GenericPrivateObject
                          xmlns:SystemCorp_Generic=
                          "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="1"
                          Field2="2" Field3="0" Field4="0" Field5="1"/>
                       </Private>
                    </DAI>
                 </SDI>
              </SDI>
              <DAI name="q">
                 <Private type="SystemCorp_Generic">
                    <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
                    "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="1" Field2
```

```
                    ="2" Field3="1" Field4="0" Field5="1"/>
                </Private>
            </DAI>
            <DAI name="t">
                <Private type="SystemCorp_Generic">
                    <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
                    "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="1" Field2
                    ="2" Field3="2" Field4="0" Field5="1"/>
                </Private>
            </DAI>
        </SDI>
        <SDI name="phsC">
            <SDI name="cVal">
                <SDI name="mag">
                    <DAI name="f">
                        <Private type="SystemCorp_Generic">
                            <SystemCorp_Generic:GenericPrivateObject
                            xmlns:SystemCorp_Generic=
                            "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="1"
                            Field2="2" Field3="0" Field4="0" Field5="2"/>
                        </Private>
                    </DAI>
                </SDI>
            </SDI>
            <DAI name="q">
                <Private type="SystemCorp_Generic">
                    <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
                    "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="1" Field2
                    ="2" Field3="1" Field4="0" Field5="2"/>
                </Private>
            </DAI>
            <DAI name="t">
                <Private type="SystemCorp_Generic">
                    <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
                    "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="1" Field2
                    ="2" Field3="2" Field4="0" Field5="2"/>
                </Private>
            </DAI>
        </SDI>
    </DOI>
</LN>
<LN desc="Measurement" inst="1" lnClass="MMXU" lnType="MMXU_0" prefix="">
    <DataSet name="VoltageDataset1">
        <FCDA doName="PhV" fc="MX" ldInst="LDevice1" lnClass="MMXU" lnInst="1"/>
    </DataSet>
    <DataSet name="CurrentDataSet1">
        <FCDA doName="A" fc="MX" ldInst="LDevice1" lnClass="MMXU" lnInst="1"/>
    </DataSet>
    <ReportControl confRev="0" datSet="VoltageDataset1" intgPd="300000" name=
    "Voltage1" rptID="MMXU1VID">
        <TrgOps dchg="true" dupd="true"/>
        <OptFields dataSet="true" seqNum="true" timeStamp="true"/>
        <RptEnabled max="3"/>
    </ReportControl>
```

```
<ReportControl confRev="0" datSet="CurrentDataSet1" intgPd="30000" name=
"Current1" rptID="MMXU1IID">
   <TrgOps dchg="true" dupd="true"/>
   <OptFields dataSet="true" reasonCode="true" seqNum="true" timeStamp="true"/>
   <RptEnabled max="3"/>
</ReportControl>
<DOI desc="Enumerated status" name="Beh">
   <DAI name="stVal">
      <Val>on</Val>
   </DAI>
</DOI>
<DOI desc="Phase to ground related measurement values of a three phase Star"
name="PhV">
   <SDI name="phsA">
      <SDI name="cVal">
         <SDI name="mag">
            <DAI name="f">
               <Private type="SystemCorp_Generic">
                  <SystemCorp_Generic:GenericPrivateObject
                  xmlns:SystemCorp_Generic=
                  "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="2"
                  Field2="1" Field3="0" Field4="0" Field5="0"/>
               </Private>
            </DAI>
         </SDI>
      </SDI>
      <DAI name="q">
         <Private type="SystemCorp_Generic">
            <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
            "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="2" Field2
            ="1" Field3="1" Field4="0" Field5="0"/>
         </Private>
      </DAI>
      <DAI name="t">
         <Private type="SystemCorp_Generic">
            <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
            "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="2" Field2
            ="1" Field3="2" Field4="0" Field5="0"/>
         </Private>
      </DAI>
   </SDI>
   <SDI name="phsB">
      <SDI name="cVal">
         <SDI name="mag">
            <DAI name="f">
               <Private type="SystemCorp_Generic">
                  <SystemCorp_Generic:GenericPrivateObject
                  xmlns:SystemCorp_Generic=
                  "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="2"
                  Field2="1" Field3="0" Field4="0" Field5="1"/>
               </Private>
            </DAI>
         </SDI>
      </SDI>
```

```
            <DAI name="q">
               <Private type="SystemCorp_Generic">
                  <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
                  "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="2" Field2
                  ="1" Field3="1" Field4="0" Field5="1"/>
               </Private>
            </DAI>
            <DAI name="t">
               <Private type="SystemCorp_Generic">
                  <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
                  "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="2" Field2
                  ="1" Field3="2" Field4="0" Field5="1"/>
               </Private>
            </DAI>
         </SDI>
         <SDI name="phsC">
            <SDI name="cVal">
               <SDI name="mag">
                  <DAI name="f">
                     <Private type="SystemCorp_Generic">
                        <SystemCorp_Generic:GenericPrivateObject
                        xmlns:SystemCorp_Generic=
                        "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="2"
                        Field2="1" Field3="0" Field4="0" Field5="2"/>
                     </Private>
                  </DAI>
               </SDI>
            </SDI>
            <DAI name="q">
               <Private type="SystemCorp_Generic">
                  <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
                  "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="2" Field2
                  ="1" Field3="1" Field4="0" Field5="2"/>
               </Private>
            </DAI>
            <DAI name="t">
               <Private type="SystemCorp_Generic">
                  <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
                  "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="2" Field2
                  ="1" Field3="2" Field4="0" Field5="2"/>
               </Private>
            </DAI>
         </SDI>
      </DOI>
      <DOI desc="Phase to ground related measurement values of a three phase Star"
      name="A">
         <SDI name="phsA">
            <SDI name="cVal">
               <SDI name="mag">
                  <DAI name="f">
                     <Private type="SystemCorp_Generic">
                        <SystemCorp_Generic:GenericPrivateObject
                        xmlns:SystemCorp_Generic=
                        "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="2"
```

```
                Field2="2" Field3="0" Field4="0" Field5="0"/>
            </Private>
         </DAI>
      </SDI>
   </SDI>
   <DAI name="q">
      <Private type="SystemCorp_Generic">
         <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
         "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="2" Field2
         ="2" Field3="1" Field4="0" Field5="0"/>
      </Private>
   </DAI>
   <DAI name="t">
      <Private type="SystemCorp_Generic">
         <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
         "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="2" Field2
         ="2" Field3="2" Field4="0" Field5="0"/>
      </Private>
   </DAI>
</SDI>
<SDI name="phsB">
   <SDI name="cVal">
      <SDI name="mag">
         <DAI name="f">
            <Private type="SystemCorp_Generic">
               <SystemCorp_Generic:GenericPrivateObject
               xmlns:SystemCorp_Generic=
               "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="2"
               Field2="2" Field3="0" Field4="0" Field5="1"/>
            </Private>
         </DAI>
      </SDI>
   </SDI>
   <DAI name="q">
      <Private type="SystemCorp_Generic">
         <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
         "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="2" Field2
         ="2" Field3="1" Field4="0" Field5="1"/>
      </Private>
   </DAI>
   <DAI name="t">
      <Private type="SystemCorp_Generic">
         <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
         "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="2" Field2
         ="2" Field3="2" Field4="0" Field5="1"/>
      </Private>
   </DAI>
</SDI>
<SDI name="phsC">
   <SDI name="cVal">
      <SDI name="mag">
         <DAI name="f">
            <Private type="SystemCorp_Generic">
               <SystemCorp_Generic:GenericPrivateObject
```

```
                                    xmlns:SystemCorp_Generic=
                                    "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="2"
                                    Field2="2" Field3="0" Field4="0" Field5="2"/>
                                </Private>
                            </DAI>
                        </SDI>
                    </SDI>
                    <DAI name="q">
                        <Private type="SystemCorp_Generic">
                            <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
                            "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="2" Field2
                            ="2" Field3="1" Field4="0" Field5="2"/>
                        </Private>
                    </DAI>
                    <DAI name="t">
                        <Private type="SystemCorp_Generic">
                            <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
                            "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="2" Field2
                            ="2" Field3="2" Field4="0" Field5="2"/>
                        </Private>
                    </DAI>
                </SDI>
            </DOI>
        </LN>
    </LDevice>
</Server>
</AccessPoint>
</IED>
<DataTypeTemplates>
<LNodeType id="MMXU_0" lnClass="MMXU">
<DO desc="Enumerated status" name="Beh" type="ENS_0"/>
<DO desc="Phase to ground related measurement values of a three phase Star" name="PhV"
type="WYE_0"/>
<DO desc="Phase to ground related measurement values of a three phase Star" name="A"
type="WYE_0"/>
</LNodeType>
<LNodeType id="XCBR_0" lnClass="XCBR">
<DO desc="Controllable enumerated status" name="Mod" type="ENC_0"/>
<DO desc="Enumerated status" name="Beh" type="ENS_0"/>
<DO desc="Single point status" name="Loc" type="SPS_0"/>
<DO desc="Integer status" name="OpCnt" type="INS_0"/>
<DO desc="Controllable double point" name="Pos" type="DPC_0"/>
<DO desc="Controllable single point" name="BlkOpn" type="SPC_0"/>
<DO desc="Controllable single point" name="BlkCls" type="SPC_0"/>
</LNodeType>
<LNodeType id="YLTC_0" lnClass="YLTC">
<DO desc="Enumerated status" name="Beh" type="ENS_0"/>
<DO desc="Integer controlled step position information" name="TapPos" type="ISC_0"/>
<DO desc="Single point status" name="EndPosR" type="SPS_0"/>
<DO desc="Single point status" name="EndPosL" type="SPS_0"/>
</LNodeType>
<LNodeType id="AVCO_0" lnClass="AVCO">
<DO desc="Enumerated status" name="Beh" type="ENS_0"/>
<DO desc="Single point status" name="Loc" type="SPS_0"/>
```

```
    <DO desc="Binary controlled step position information" name="TapChg" type="BSC_0"/>
    <DO desc="Controllable analogue set point information" name="SptVol" type="APC_0"/>
</LNodeType>
<LNodeType id="LPHD_0" lnClass="LPHD">
    <DO desc="Enumerated status" name="Beh" type="ENS_0"/>
    <DO desc="Device name plate" name="PhyNam" type="DPL_0"/>
    <DO desc="Enumerated status" name="PhyHealth" type="ENS_0"/>
    <DO desc="Single point status" name="Proxy" type="SPS_0"/>
</LNodeType>
<LNodeType id="LLN0_0" lnClass="LLN0">
    <DO desc="Controllable enumerated status" name="Mod" type="ENC_1"/>
    <DO desc="Enumerated status" name="Beh" type="ENS_2"/>
    <DO desc="Enumerated status" name="Health" type="ENS_1"/>
    <DO desc="Logical Node name plate" name="NamPlt" type="LPL_0"/>
</LNodeType>
<DOType cdc="CMV" desc="Complex measured value" id="phsC_0">
    <DA bType="Struct" dchg="true" dupd="true" fc="MX" name="cVal" type="cVal_0"/>
    <DA bType="Quality" fc="MX" name="q" qchg="true"/>
    <DA bType="Timestamp" fc="MX" name="t"/>
</DOType>
<DOType cdc="CMV" desc="Complex measured value" id="phsB_0">
    <DA bType="Struct" dchg="true" dupd="true" fc="MX" name="cVal" type="cVal_0"/>
    <DA bType="Quality" fc="MX" name="q" qchg="true"/>
    <DA bType="Timestamp" fc="MX" name="t"/>
</DOType>
<DOType cdc="CMV" desc="Complex measured value" id="phsA_0">
    <DA bType="Struct" dchg="true" dupd="true" fc="MX" name="cVal" type="cVal_0"/>
    <DA bType="Quality" fc="MX" name="q" qchg="true"/>
    <DA bType="Timestamp" fc="MX" name="t"/>
</DOType>
<DOType cdc="WYE" desc="Phase to ground related measurement values of a three phase Star"
id="WYE_0">
    <SDO desc="Complex measured value" name="phsA" type="phsA_0"/>
    <SDO desc="Complex measured value" name="phsB" type="phsB_0"/>
    <SDO desc="Complex measured value" name="phsC" type="phsC_0"/>
</DOType>
<DOType cdc="SPC" desc="Controllable single point" id="SPC_0">
    <DA bType="Enum" dchg="true" fc="CF" name="ctlModel" type="ctlModel"/>
</DOType>
<DOType cdc="DPC" desc="Controllable double point" id="DPC_0">
    <DA bType="Dbpos" dchg="true" fc="ST" name="stVal" type="Dbpos"/>
    <DA bType="Quality" fc="ST" name="q" qchg="true"/>
    <DA bType="Timestamp" fc="ST" name="t"/>
    <DA bType="Enum" dchg="true" fc="CF" name="ctlModel" type="ctlModel"/>
</DOType>
<DOType cdc="INS" desc="Integer status" id="INS_0">
    <DA bType="INT32" dchg="true" dupd="true" fc="ST" name="stVal"/>
    <DA bType="Quality" fc="ST" name="q" qchg="true"/>
    <DA bType="Timestamp" fc="ST" name="t"/>
</DOType>
<DOType cdc="SPS" desc="Single point status" id="SPS_0">
    <DA bType="BOOLEAN" dchg="true" fc="ST" name="stVal"/>
    <DA bType="Quality" fc="ST" name="q" qchg="true"/>
    <DA bType="Timestamp" fc="ST" name="t"/>
```

```
</DOType>
<DOType cdc="ENS" desc="Enumerated status" id="ENS_0">
   <DA bType="Enum" dchg="true" dupd="true" fc="ST" name="stVal" type="Mod"/>
   <DA bType="Quality" fc="ST" name="q" qchg="true"/>
   <DA bType="Timestamp" fc="ST" name="t"/>
</DOType>
<DOType cdc="ENC" desc="Controllable enumerated status" id="ENC_0">
   <DA bType="Struct" fc="CO" name="Oper" type="Oper_0"/>
   <DA bType="Enum" dchg="true" fc="ST" name="stVal" type="Mod"/>
   <DA bType="Quality" fc="ST" name="q" qchg="true"/>
   <DA bType="Timestamp" fc="ST" name="t"/>
   <DA bType="Enum" fc="CF" name="ctlModel" type="ctlModel"/>
</DOType>
<DOType cdc="ISC" desc="Integer controlled step position information" id="ISC_0">
   <DA bType="Struct" dchg="true" fc="ST" name="valWTr" type="valWTr_0"/>
   <DA bType="Quality" fc="ST" name="q" qchg="true"/>
   <DA bType="Timestamp" fc="ST" name="t"/>
   <DA bType="Enum" dchg="true" fc="CF" name="ctlModel" type="ctlModel"/>
</DOType>
<DOType cdc="APC" desc="Controllable analogue set point information" id="APC_0">
   <DA bType="Struct" fc="CO" name="Oper" type="Oper_1"/>
   <DA bType="Struct" dchg="true" fc="MX" name="mxVal" type="mxVal_0"/>
   <DA bType="Quality" fc="MX" name="q" qchg="true"/>
   <DA bType="Timestamp" fc="MX" name="t"/>
   <DA bType="Enum" fc="CF" name="ctlModel" type="ctlModel"/>
</DOType>
<DOType cdc="BSC" desc="Binary controlled step position information" id="BSC_0">
   <DA bType="Struct" dchg="true" fc="ST" name="valWTr" type="valWTr_0"/>
   <DA bType="Quality" fc="ST" name="q" qchg="true"/>
   <DA bType="Timestamp" fc="ST" name="t"/>
   <DA bType="BOOLEAN" dchg="true" fc="CF" name="persistent"/>
   <DA bType="Enum" dchg="true" fc="CF" name="ctlModel" type="ctlModel"/>
</DOType>
<DOType cdc="DPL" desc="Device name plate" id="DPL_0">
   <DA bType="VisString255" fc="DC" name="vendor"/>
</DOType>
<DOType cdc="LPL" desc="Logical Node name plate" id="LPL_0">
   <DA bType="VisString255" fc="DC" name="vendor"/>
   <DA bType="VisString255" fc="DC" name="swRev"/>
   <DA bType="VisString255" fc="DC" name="d"/>
</DOType>
<DOType cdc="ENS" desc="Enumerated status" id="ENS_1">
   <DA bType="Enum" dchg="true" fc="ST" name="stVal" type="Health"/>
   <DA bType="Quality" fc="ST" name="q" qchg="true"/>
   <DA bType="Timestamp" fc="ST" name="t"/>
</DOType>
<DOType cdc="ENS" desc="Enumerated status" id="ENS_2">
   <DA bType="Enum" dchg="true" fc="ST" name="stVal" type="Mod"/>
   <DA bType="Quality" fc="ST" name="q" qchg="true"/>
   <DA bType="Timestamp" fc="ST" name="t"/>
</DOType>
<DOType cdc="ENC" desc="Controllable enumerated status" id="ENC_1">
   <DA bType="Enum" dchg="true" fc="ST" name="stVal" type="Mod"/>
   <DA bType="Quality" fc="ST" name="q" qchg="true"/>
```

```
    <DA bType="Timestamp" fc="ST" name="t"/>
    <DA bType="Enum" fc="CF" name="ctlModel" type="ctlModel"/>
</DOType>
<DAType id="origin_0">
    <BDA bType="Enum" name="orCat" type="orCat"/>
    <BDA bType="Octet64" name="orIdent"/>
</DAType>
<DAType id="Oper_0">
    <BDA bType="BOOLEAN" name="ctlVal" type="Dbpos"/>
    <BDA bType="Struct" name="origin" type="origin_0"/>
    <BDA bType="INT8U" name="ctlNum"/>
    <BDA bType="Timestamp" name="T"/>
    <BDA bType="BOOLEAN" name="Test"/>
    <BDA bType="Check" name="Check"/>
</DAType>
<DAType id="valWTr_0">
    <BDA bType="INT8" name="posVal"/>
</DAType>
<DAType id="mxVal_0">
    <BDA bType="INT32" name="i"/>
    <BDA bType="FLOAT32" name="f"/>
</DAType>
<DAType id="Oper_1">
    <BDA bType="FLOAT32" name="ctlVal"/>
    <BDA bType="Struct" name="origin" type="origin_0"/>
    <BDA bType="INT8U" name="ctlNum"/>
    <BDA bType="Timestamp" name="T"/>
    <BDA bType="BOOLEAN" name="Test"/>
    <BDA bType="Check" name="Check"/>
</DAType>
<DAType id="mag_0">
    <BDA bType="FLOAT32" name="f"/>
</DAType>
<DAType id="cVal_0">
    <BDA bType="Struct" name="mag" type="mag_0"/>
</DAType>
<EnumType id="ctlModel">
    <EnumVal ord="0">status-only</EnumVal>
    <EnumVal ord="1">direct-with-normal-security</EnumVal>
    <EnumVal ord="2">sbo-with-normal-security</EnumVal>
    <EnumVal ord="3">direct-with-enhanced-security</EnumVal>
    <EnumVal ord="4">sbo-with-enhanced-security</EnumVal>
</EnumType>
<EnumType id="Dbpos">
    <EnumVal ord="0">intermediate</EnumVal>
    <EnumVal ord="1">off</EnumVal>
    <EnumVal ord="2">on</EnumVal>
    <EnumVal ord="3">bad</EnumVal>
</EnumType>
<EnumType id="Mod">
    <EnumVal ord="1">on</EnumVal>
    <EnumVal ord="2">on-blocked</EnumVal>
    <EnumVal ord="3">test</EnumVal>
    <EnumVal ord="4">test/blocked</EnumVal>
```

```
         <EnumVal ord="5">off</EnumVal>
      </EnumType>
      <EnumType id="orCat">
         <EnumVal ord="0">not-supported</EnumVal>
         <EnumVal ord="1">bay-control</EnumVal>
         <EnumVal ord="2">station-control</EnumVal>
         <EnumVal ord="3">remote-control</EnumVal>
         <EnumVal ord="4">automatic-bay</EnumVal>
         <EnumVal ord="5">automatic-station</EnumVal>
         <EnumVal ord="6">automatic-remote</EnumVal>
         <EnumVal ord="7">maintenance</EnumVal>
         <EnumVal ord="8">process</EnumVal>
      </EnumType>
      <EnumType id="Health">
         <EnumVal ord="1">Ok</EnumVal>
         <EnumVal ord="2">Warning</EnumVal>
         <EnumVal ord="3">Alarm</EnumVal>
      </EnumType>
   </DataTypeTemplates>
</SCL>
```

9.4 Client CID file

```
<?xml version="1.0" encoding="UTF-8"?>
<SCL xmlns="http://www.iec.ch/61850/2003/SCL" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" revision="A" version="2007"
xsi:schemaLocation="http://www.iec.ch/61850/2003/SCL SCL.xsd">
   <Header id="0" version="0"/>
   <Communication>
      <SubNetwork name="SubNetworkName">
         <ConnectedAP apName="SubstationRing1" iedName="NewIED">
            <Address>
               <P type="OSI-AP-Title">1,1,9999,1</P>
               <P type="OSI-AE-Qualifier">12</P>
               <P type="OSI-PSEL">00000001</P>
               <P type="OSI-SSEL">0001</P>
               <P type="OSI-TSEL">0001</P>
               <P type="IP">130.237.44.155</P>
               <P type="IP-SUBNET">255.255.255.0</P>
               <P type="IP-GATEWAY">130.237.44.1</P>
               <P type="MAC-Address">08-00-27-A0-D8-5F</P>
            </Address>
            <GSE cbName="GSE_CB" ldInst="LDevice1">
               <Address>
                  <P type="MAC-Address">01-0C-CD-01-00-01</P>
                  <P type="APPID">0000</P>
                  <P type="VLAN-PRIORITY">4</P>
                  <P type="VLAN-ID">000</P>
               </Address>
               <MinTime>20</MinTime>
               <MaxTime>4000</MaxTime>
            </GSE>
         </ConnectedAP>
         <ConnectedAP apName="SubstationRing2" iedName="NewClientIED">
            <Address>
               <P type="OSI-AP-Title">1,1,9999,1</P>
               <P type="OSI-AE-Qualifier">12</P>
               <P type="OSI-PSEL">00000001</P>
               <P type="OSI-SSEL">0001</P>
               <P type="OSI-TSEL">0001</P>
               <P type="IP">130.237.53.154</P>
               <P type="IP-SUBNET">255.255.255.0</P>
               <P type="IP-GATEWAY">130.237.53.1</P>
               <P type="MAC-Address">9C-EB-E8-12-5E-3F</P>
            </Address>
         </ConnectedAP>
      </SubNetwork>
   </Communication>
   <IED name="NewClientIED">
      <AccessPoint name="SubstationRing2">
         <LN inst="0" lnClass="IHMI" lnType="IHMI_0" prefix=""/>
      </AccessPoint>
   </IED>
   <IED desc="NewIED" manufacturer="SystemCORP Pty Ltd" name="NewIED" type="RTUType">
      <Services>
         <GOOSE max="1"/>
      </Services>
```

```
<AccessPoint name="SubstationRing1" router="false">
   <Server timeout="30">
      <Authentication/>
      <LDevice inst="LDevice1">
         <LN0 desc="Logical node zero" inst="" lnClass="LLN0" lnType="LLN0_0">
            <DataSet name="GooseCBDataset">
               <FCDA daName="stVal" doName="Pos" fc="ST" ldInst="LDevice1" lnClass="XCBR"
               lnInst="0"/>
               <FCDA daName="Oper.ctlVal" doName="Mod" fc="CO" ldInst="LDevice1" lnClass=
               "XCBR" lnInst="0"/>
            </DataSet>
            <DOI desc="Controllable enumerated status" name="Mod">
               <DAI name="stVal">
                  <Val>on</Val>
               </DAI>
            </DOI>
            <DOI desc="Enumerated status" name="Beh">
               <DAI name="stVal">
                  <Val>on</Val>
               </DAI>
            </DOI>
            <DOI desc="Enumerated status" name="Health">
               <DAI name="stVal">
                  <Val>Ok</Val>
               </DAI>
            </DOI>
            <GSEControl appID="GSE_ControlBlock" confRev="1" datSet="GooseCBDataset" name=
            "GSE_CB" type="GOOSE"/>
         </LN0>
         <LN desc="Physical device information" inst="0" lnClass="LPHD" lnType="LPHD_0"
         prefix=""/>
         <LN desc="Voltage control" inst="0" lnClass="AVCO" lnType="AVCO_0" prefix="">
            <DataSet name="VoltageSetPointDataset">
               <FCDA doName="SptVol" fc="MX" ldInst="LDevice1" lnClass="AVCO" lnInst="0"/>
               <FCDA doName="SptVol" fc="CF" ldInst="LDevice1" lnClass="AVCO" lnInst="0"/>
            </DataSet>
            <ReportControl confRev="1" datSet="VoltageSetPointDataset" intgPd="300000"
            name="Setpoint" rptID="AVCOID">
               <TrgOps dchg="true" dupd="true" qchg="true"/>
               <OptFields seqNum="true" timeStamp="true"/>
               <RptEnabled max="3">
                  <ClientLN iedName="NewClientIED" ldInst="none" lnClass="IHMI" lnInst="0"
                  />
               </RptEnabled>
            </ReportControl>
            <DOI desc="Enumerated status" name="Beh">
               <DAI name="stVal">
                  <Val>on</Val>
               </DAI>
            </DOI>
            <DOI desc="Controllable analogue set point information" name="SptVol">
               <DAI name="q">
                  <Private type="SystemCorp_Generic">
                     <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
```

```
                    "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="5" Field2=
                    "0" Field3="1" Field4="0" Field5="0"/>
                </Private>
            </DAI>
            <DAI name="t">
                <Private type="SystemCorp_Generic">
                    <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
                    "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="5" Field2=
                    "0" Field3="2" Field4="0" Field5="0"/>
                </Private>
            </DAI>
            <DAI name="ctlModel">
                <Val>direct-with-normal-security</Val>
            </DAI>
            <SDI name="Oper">
                <DAI name="ctlVal">
                    <Private type="SystemCorp_Generic">
                        <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
                        "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="5" Field2
                        ="0" Field3="0" Field4="0" Field5="1"/>
                    </Private>
                </DAI>
            </SDI>
            <SDI name="mxVal">
                <DAI name="f">
                    <Private type="SystemCorp_Generic">
                        <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
                        "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="5" Field2
                        ="0" Field3="0" Field4="0" Field5="0"/>
                    </Private>
                </DAI>
            </SDI>
        </DOI>
    </LN>
    <LN desc="Tap changer" inst="0" lnClass="YLTC" lnType="YLTC_0" prefix="">
        <DataSet name="TapPositionDataset">
            <FCDA doName="TapPos" fc="ST" ldInst="LDevice1" lnClass="YLTC" lnInst="0"/>
            <FCDA doName="TapPos" fc="CF" ldInst="LDevice1" lnClass="YLTC" lnInst="0"/>
        </DataSet>
        <ReportControl confRev="0" datSet="TapPositionDataset" intgPd="300000" name=
        "TapPosition" rptID="YLTCOID">
            <TrgOps dchg="true" dupd="true"/>
            <OptFields dataSet="true" seqNum="true" timeStamp="true"/>
            <RptEnabled max="3">
                <ClientLN iedName="NewClientIED" ldInst="none" lnClass="IHMI" lnInst="0"
                />
            </RptEnabled>
        </ReportControl>
        <DOI desc="Integer controlled step position information" name="TapPos">
            <DAI name="q">
                <Private type="SystemCorp_Generic">
                    <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
                    "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="4" Field2=
                    "2" Field3="1" Field4="0" Field5="0"/>
```

```
            </Private>
         </DAI>
         <DAI name="t">
            <Private type="SystemCorp_Generic">
               <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
               "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="4" Field2=
               "2" Field3="2" Field4="0" Field5="0"/>
            </Private>
         </DAI>
         <DAI name="ctlModel">
            <Val>status-only</Val>
         </DAI>
         <SDI name="valWTr">
            <DAI name="posVal">
               <Private type="SystemCorp_Generic">
                  <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
                  "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="4" Field2
                  ="2" Field3="0" Field4="0" Field5="0"/>
               </Private>
            </DAI>
         </SDI>
      </DOI>
      <DOI desc="Single point status" name="EndPosR">
         <DAI name="stVal">
            <Private type="SystemCorp_Generic">
               <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
               "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="4" Field2=
               "0" Field3="0" Field4="0" Field5="0"/>
            </Private>
         </DAI>
         <DAI name="q">
            <Private type="SystemCorp_Generic">
               <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
               "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="4" Field2=
               "0" Field3="1" Field4="0" Field5="0"/>
            </Private>
         </DAI>
         <DAI name="t">
            <Private type="SystemCorp_Generic">
               <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
               "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="4" Field2=
               "0" Field3="2" Field4="0" Field5="0"/>
            </Private>
         </DAI>
      </DOI>
      <DOI desc="Single point status" name="EndPosL">
         <DAI name="stVal">
            <Private type="SystemCorp_Generic">
               <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
               "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="4" Field2=
               "1" Field3="0" Field4="0" Field5="0"/>
            </Private>
         </DAI>
         <DAI name="q">
```

```
            <Private type="SystemCorp_Generic">
               <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
               "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="4" Field2=
               "1" Field3="1" Field4="0" Field5="0"/>
            </Private>
         </DAI>
         <DAI name="t">
            <Private type="SystemCorp_Generic">
               <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
               "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="4" Field2=
               "1" Field3="2" Field4="0" Field5="0"/>
            </Private>
         </DAI>
      </DOI>
   </LN>
   <LN desc="Circuit breaker" inst="0" lnClass="XCBR" lnType="XCBR_0" prefix="">
      <DataSet name="CBDataset">
         <FCDA doName="Mod" fc="ST" ldInst="LDevice1" lnClass="XCBR" lnInst="0"/>
         <FCDA doName="Pos" fc="ST" ldInst="LDevice1" lnClass="XCBR" lnInst="0"/>
         <FCDA doName="Mod" fc="CO" ldInst="LDevice1" lnClass="XCBR" lnInst="0"/>
         <FCDA doName="Mod" fc="CF" ldInst="LDevice1" lnClass="XCBR" lnInst="0"/>
         <FCDA doName="Pos" fc="CF" ldInst="LDevice1" lnClass="XCBR" lnInst="0"/>
      </DataSet>
      <DataSet name="DataSet01">
         <FCDA daName="stVal" doName="Pos" fc="ST" ldInst="LDevice1" lnClass="XCBR"
         lnInst="0"/>
      </DataSet>
      <ReportControl confRev="0" datSet="CBDataset" intgPd="30000" name=
      "CircuitBreaker" rptID="XCBRID">
         <TrgOps dchg="true" dupd="true"/>
         <OptFields dataRef="true" dataSet="true" reasonCode="true" seqNum="true"
         timeStamp="true"/>
         <RptEnabled max="3">
            <ClientLN iedName="NewClientIED" ldInst="none" lnClass="IHMI" lnInst="0"
            />
         </RptEnabled>
      </ReportControl>
      <DOI desc="Controllable enumerated status" name="Mod">
         <DAI name="stVal">
            <Private type="SystemCorp_Generic">
               <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
               "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="3" Field2=
               "1" Field3="0" Field4="0" Field5="0"/>
            </Private>
         </DAI>
         <DAI name="ctlModel">
            <Val>direct-with-normal-security</Val>
         </DAI>
         <SDI name="Oper">
            <DAI name="ctlVal">
               <Private type="SystemCorp_Generic">
                  <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
                  "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="3" Field2
                  ="0" Field3="0" Field4="0" Field5="1"/>
```

```
                    </Private>
                </DAI>
            </SDI>
        </DOI>
        <DOI desc="Controllable double point" name="Pos">
            <DAI name="stVal">
                <Private type="SystemCorp_Generic">
                    <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
                    "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="3" Field2=
                    "0" Field3="0" Field4="0" Field5="0"/>
                </Private>
            </DAI>
            <DAI name="q">
                <Private type="SystemCorp_Generic">
                    <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
                    "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="3" Field2=
                    "0" Field3="1" Field4="0" Field5="0"/>
                </Private>
            </DAI>
            <DAI name="t">
                <Private type="SystemCorp_Generic">
                    <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
                    "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="3" Field2=
                    "0" Field3="2" Field4="0" Field5="0"/>
                </Private>
            </DAI>
            <DAI name="ctlModel">
                <Val>direct-with-normal-security</Val>
            </DAI>
        </DOI>
    </LN>
    <LN desc="Measurement" inst="0" lnClass="MMXU" lnType="MMXU_0" prefix="">
        <DataSet name="CurrentDataset">
            <FCDA doName="A" fc="MX" ldInst="LDevice1" lnClass="MMXU" lnInst="0"/>
        </DataSet>
        <DataSet name="VoltageDataset">
            <FCDA doName="PhV" fc="MX" ldInst="LDevice1" lnClass="MMXU" lnInst="0"/>
        </DataSet>
        <ReportControl confRev="0" datSet="VoltageDataset" intgPd="300000" name=
        "Voltage" rptID="MMXUVID">
            <TrgOps dchg="true" dupd="true"/>
            <OptFields dataSet="true" seqNum="true" timeStamp="true"/>
            <RptEnabled max="3">
                <ClientLN iedName="NewClientIED" ldInst="none" lnClass="IHMI" lnInst="0"
                />
            </RptEnabled>
        </ReportControl>
        <ReportControl confRev="0" datSet="CurrentDataset" intgPd="30000" name=
        "Current" rptID="MMXUIID">
            <TrgOps dchg="true" dupd="true"/>
            <OptFields dataSet="true" reasonCode="true" seqNum="true" timeStamp="true"/>
            <RptEnabled max="3">
                <ClientLN iedName="NewClientIED" ldInst="none" lnClass="IHMI" lnInst="0"
                />
```

```
      </RptEnabled>
   </ReportControl>
   <DOI desc="Enumerated status" name="Beh">
      <DAI name="stVal">
         <Val>on</Val>
      </DAI>
   </DOI>
   <DOI desc="Phase to ground related measurement values of a three phase Star"
   name="PhV">
      <SDI name="phsA">
         <SDI name="cVal">
            <SDI name="mag">
               <DAI name="f">
                  <Private type="SystemCorp_Generic">
                     <SystemCorp_Generic:GenericPrivateObject
                     xmlns:SystemCorp_Generic=
                     "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="1"
                     Field2="1" Field3="0" Field4="0" Field5="0"/>
                  </Private>
               </DAI>
            </SDI>
         </SDI>
         <DAI name="q">
            <Private type="SystemCorp_Generic">
               <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
               "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="1" Field2
               ="1" Field3="1" Field4="0" Field5="0"/>
            </Private>
         </DAI>
         <DAI name="t">
            <Private type="SystemCorp_Generic">
               <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
               "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="1" Field2
               ="1" Field3="2" Field4="0" Field5="0"/>
            </Private>
         </DAI>
      </SDI>
      <SDI name="phsB">
         <SDI name="cVal">
            <SDI name="mag">
               <DAI name="f">
                  <Private type="SystemCorp_Generic">
                     <SystemCorp_Generic:GenericPrivateObject
                     xmlns:SystemCorp_Generic=
                     "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="1"
                     Field2="1" Field3="0" Field4="0" Field5="1"/>
                  </Private>
               </DAI>
            </SDI>
         </SDI>
         <DAI name="q">
            <Private type="SystemCorp_Generic">
               <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
               "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="1" Field2
```

```
                    ="1" Field3="1" Field4="0" Field5="1"/>
                </Private>
            </DAI>
            <DAI name="t">
                <Private type="SystemCorp_Generic">
                    <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
                    "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="1" Field2
                    ="1" Field3="2" Field4="0" Field5="1"/>
                </Private>
            </DAI>
        </SDI>
        <SDI name="phsC">
            <SDI name="cVal">
                <SDI name="mag">
                    <DAI name="f">
                        <Private type="SystemCorp_Generic">
                            <SystemCorp_Generic:GenericPrivateObject
                            xmlns:SystemCorp_Generic=
                            "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="1"
                            Field2="1" Field3="0" Field4="0" Field5="2"/>
                        </Private>
                    </DAI>
                </SDI>
            </SDI>
            <DAI name="q">
                <Private type="SystemCorp_Generic">
                    <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
                    "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="1" Field2
                    ="1" Field3="1" Field4="0" Field5="2"/>
                </Private>
            </DAI>
            <DAI name="t">
                <Private type="SystemCorp_Generic">
                    <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
                    "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="1" Field2
                    ="1" Field3="2" Field4="0" Field5="2"/>
                </Private>
            </DAI>
        </SDI>
    </DOI>
    <DOI desc="Phase to ground related measurement values of a three phase Star"
    name="A">
        <SDI name="phsA">
            <SDI name="cVal">
                <SDI name="mag">
                    <DAI name="f">
                        <Private type="SystemCorp_Generic">
                            <SystemCorp_Generic:GenericPrivateObject
                            xmlns:SystemCorp_Generic=
                            "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="1"
                            Field2="2" Field3="0" Field4="0" Field5="0"/>
                        </Private>
                    </DAI>
                </SDI>
```

```
      </SDI>
      <DAI name="q">
         <Private type="SystemCorp_Generic">
            <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
            "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="1" Field2
            ="2" Field3="1" Field4="0" Field5="0"/>
         </Private>
      </DAI>
      <DAI name="t">
         <Private type="SystemCorp_Generic">
            <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
            "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="1" Field2
            ="2" Field3="2" Field4="0" Field5="0"/>
         </Private>
      </DAI>
   </SDI>
   <SDI name="phsB">
      <SDI name="cVal">
         <SDI name="mag">
            <DAI name="f">
               <Private type="SystemCorp_Generic">
                  <SystemCorp_Generic:GenericPrivateObject
                  xmlns:SystemCorp_Generic=
                  "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="1"
                  Field2="2" Field3="0" Field4="0" Field5="1"/>
               </Private>
            </DAI>
         </SDI>
      </SDI>
      <DAI name="q">
         <Private type="SystemCorp_Generic">
            <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
            "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="1" Field2
            ="2" Field3="1" Field4="0" Field5="1"/>
         </Private>
      </DAI>
      <DAI name="t">
         <Private type="SystemCorp_Generic">
            <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
            "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="1" Field2
            ="2" Field3="2" Field4="0" Field5="1"/>
         </Private>
      </DAI>
   </SDI>
   <SDI name="phsC">
      <SDI name="cVal">
         <SDI name="mag">
            <DAI name="f">
               <Private type="SystemCorp_Generic">
                  <SystemCorp_Generic:GenericPrivateObject
                  xmlns:SystemCorp_Generic=
                  "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="1"
                  Field2="2" Field3="0" Field4="0" Field5="2"/>
               </Private>
```

```
                        </DAI>
                     </SDI>
                  </SDI>
                  <DAI name="q">
                     <Private type="SystemCorp_Generic">
                        <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
                        "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="1" Field2
                        ="2" Field3="1" Field4="0" Field5="2"/>
                     </Private>
                  </DAI>
                  <DAI name="t">
                     <Private type="SystemCorp_Generic">
                        <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
                        "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="1" Field2
                        ="2" Field3="2" Field4="0" Field5="2"/>
                     </Private>
                  </DAI>
               </SDI>
            </DOI>
         </LN>
         <LN desc="Measurement" inst="1" lnClass="MMXU" lnType="MMXU_0" prefix="">
            <DataSet name="VoltageDataset1">
               <FCDA doName="PhV" fc="MX" ldInst="LDevice1" lnClass="MMXU" lnInst="1"/>
            </DataSet>
            <DataSet name="CurrentDataSet1">
               <FCDA doName="A" fc="MX" ldInst="LDevice1" lnClass="MMXU" lnInst="1"/>
            </DataSet>
            <ReportControl confRev="0" datSet="VoltageDataset1" intgPd="300000" name=
            "Voltage1" rptID="MMXU1VID">
               <TrgOps dchg="true" dupd="true"/>
               <OptFields dataSet="true" seqNum="true" timeStamp="true"/>
               <RptEnabled max="3">
                  <ClientLN iedName="NewClientIED" ldInst="none" lnClass="IHMI" lnInst="0"
                  />
               </RptEnabled>
            </ReportControl>
            <ReportControl confRev="0" datSet="CurrentDataSet1" intgPd="30000" name=
            "Current1" rptID="MMXU1IID">
               <TrgOps dchg="true" dupd="true"/>
               <OptFields dataSet="true" reasonCode="true" seqNum="true" timeStamp="true"/>
               <RptEnabled max="3">
                  <ClientLN iedName="NewClientIED" ldInst="none" lnClass="IHMI" lnInst="0"
                  />
               </RptEnabled>
            </ReportControl>
            <DOI desc="Enumerated status" name="Beh">
               <DAI name="stVal">
                  <Val>on</Val>
               </DAI>
            </DOI>
            <DOI desc="Phase to ground related measurement values of a three phase Star"
            name="PhV">
               <SDI name="phsA">
                  <SDI name="cVal">
```

```
            <SDI name="mag">
               <DAI name="f">
                  <Private type="SystemCorp_Generic">
                     <SystemCorp_Generic:GenericPrivateObject
                     xmlns:SystemCorp_Generic=
                     "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="2"
                     Field2="1" Field3="0" Field4="0" Field5="0"/>
                  </Private>
               </DAI>
            </SDI>
         </SDI>
         <DAI name="q">
            <Private type="SystemCorp_Generic">
               <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
               "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="2" Field2
               ="1" Field3="1" Field4="0" Field5="0"/>
            </Private>
         </DAI>
         <DAI name="t">
            <Private type="SystemCorp_Generic">
               <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
               "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="2" Field2
               ="1" Field3="2" Field4="0" Field5="0"/>
            </Private>
         </DAI>
      </SDI>
      <SDI name="phsB">
         <SDI name="cVal">
            <SDI name="mag">
               <DAI name="f">
                  <Private type="SystemCorp_Generic">
                     <SystemCorp_Generic:GenericPrivateObject
                     xmlns:SystemCorp_Generic=
                     "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="2"
                     Field2="1" Field3="0" Field4="0" Field5="1"/>
                  </Private>
               </DAI>
            </SDI>
         </SDI>
         <DAI name="q">
            <Private type="SystemCorp_Generic">
               <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
               "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="2" Field2
               ="1" Field3="1" Field4="0" Field5="1"/>
            </Private>
         </DAI>
         <DAI name="t">
            <Private type="SystemCorp_Generic">
               <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
               "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="2" Field2
               ="1" Field3="2" Field4="0" Field5="1"/>
            </Private>
         </DAI>
      </SDI>
```

```
      <SDI name="phsC">
         <SDI name="cVal">
            <SDI name="mag">
               <DAI name="f">
                  <Private type="SystemCorp_Generic">
                     <SystemCorp_Generic:GenericPrivateObject
                     xmlns:SystemCorp_Generic=
                     "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="2"
                     Field2="1" Field3="0" Field4="0" Field5="2"/>
                  </Private>
               </DAI>
            </SDI>
         </SDI>
         <DAI name="q">
            <Private type="SystemCorp_Generic">
               <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
               "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="2" Field2
               ="1" Field3="1" Field4="0" Field5="2"/>
            </Private>
         </DAI>
         <DAI name="t">
            <Private type="SystemCorp_Generic">
               <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
               "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="2" Field2
               ="1" Field3="2" Field4="0" Field5="2"/>
            </Private>
         </DAI>
      </SDI>
   </DOI>
   <DOI desc="Phase to ground related measurement values of a three phase Star"
   name="A">
      <SDI name="phsA">
         <SDI name="cVal">
            <SDI name="mag">
               <DAI name="f">
                  <Private type="SystemCorp_Generic">
                     <SystemCorp_Generic:GenericPrivateObject
                     xmlns:SystemCorp_Generic=
                     "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="2"
                     Field2="2" Field3="0" Field4="0" Field5="0"/>
                  </Private>
               </DAI>
            </SDI>
         </SDI>
         <DAI name="q">
            <Private type="SystemCorp_Generic">
               <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
               "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="2" Field2
               ="2" Field3="1" Field4="0" Field5="0"/>
            </Private>
         </DAI>
         <DAI name="t">
            <Private type="SystemCorp_Generic">
               <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
```

```
            "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="2" Field2
            ="2" Field3="2" Field4="0" Field5="0"/>
         </Private>
      </DAI>
   </SDI>
   <SDI name="phsB">
      <SDI name="cVal">
         <SDI name="mag">
            <DAI name="f">
               <Private type="SystemCorp_Generic">
                  <SystemCorp_Generic:GenericPrivateObject
                  xmlns:SystemCorp_Generic=
                  "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="2"
                  Field2="2" Field3="0" Field4="0" Field5="1"/>
               </Private>
            </DAI>
         </SDI>
      </SDI>
      <DAI name="q">
         <Private type="SystemCorp_Generic">
            <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
            "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="2" Field2
            ="2" Field3="1" Field4="0" Field5="1"/>
         </Private>
      </DAI>
      <DAI name="t">
         <Private type="SystemCorp_Generic">
            <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
            "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="2" Field2
            ="2" Field3="2" Field4="0" Field5="1"/>
         </Private>
      </DAI>
   </SDI>
   <SDI name="phsC">
      <SDI name="cVal">
         <SDI name="mag">
            <DAI name="f">
               <Private type="SystemCorp_Generic">
                  <SystemCorp_Generic:GenericPrivateObject
                  xmlns:SystemCorp_Generic=
                  "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="2"
                  Field2="2" Field3="0" Field4="0" Field5="2"/>
               </Private>
            </DAI>
         </SDI>
      </SDI>
      <DAI name="q">
         <Private type="SystemCorp_Generic">
            <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
            "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="2" Field2
            ="2" Field3="1" Field4="0" Field5="2"/>
         </Private>
      </DAI>
      <DAI name="t">
```

```
                        <Private type="SystemCorp_Generic">
                           <SystemCorp_Generic:GenericPrivateObject xmlns:SystemCorp_Generic=
                           "http://www.systemcorp.com.au/61850/SCL/Generic" Field1="2" Field2
                           ="2" Field3="2" Field4="0" Field5="2"/>
                        </Private>
                     </DAI>
                  </SDI>
               </DOI>
            </LN>
         </LDevice>
      </Server>
   </AccessPoint>
</IED>
<DataTypeTemplates>
   <LNodeType id="MMXU_0" lnClass="MMXU">
      <DO desc="Enumerated status" name="Beh" type="ENS_0"/>
      <DO desc="Phase to ground related measurement values of a three phase Star" name="PhV"
      type="WYE_0"/>
      <DO desc="Phase to ground related measurement values of a three phase Star" name="A"
      type="WYE_0"/>
   </LNodeType>
   <LNodeType id="XCBR_0" lnClass="XCBR">
      <DO desc="Controllable enumerated status" name="Mod" type="ENC_0"/>
      <DO desc="Enumerated status" name="Beh" type="ENS_0"/>
      <DO desc="Single point status" name="Loc" type="SPS_0"/>
      <DO desc="Integer status" name="OpCnt" type="INS_0"/>
      <DO desc="Controllable double point" name="Pos" type="DPC_0"/>
      <DO desc="Controllable single point" name="BlkOpn" type="SPC_0"/>
      <DO desc="Controllable single point" name="BlkCls" type="SPC_0"/>
   </LNodeType>
   <LNodeType id="YLTC_0" lnClass="YLTC">
      <DO desc="Enumerated status" name="Beh" type="ENS_0"/>
      <DO desc="Integer controlled step position information" name="TapPos" type="ISC_0"/>
      <DO desc="Single point status" name="EndPosR" type="SPS_0"/>
      <DO desc="Single point status" name="EndPosL" type="SPS_0"/>
   </LNodeType>
   <LNodeType id="AVCO_0" lnClass="AVCO">
      <DO desc="Enumerated status" name="Beh" type="ENS_0"/>
      <DO desc="Single point status" name="Loc" type="SPS_0"/>
      <DO desc="Binary controlled step position information" name="TapChg" type="BSC_0"/>
      <DO desc="Controllable analogue set point information" name="SptVol" type="APC_0"/>
   </LNodeType>
   <LNodeType id="LPHD_0" lnClass="LPHD">
      <DO desc="Enumerated status" name="Beh" type="ENS_0"/>
      <DO desc="Device name plate" name="PhyNam" type="DPL_0"/>
      <DO desc="Enumerated status" name="PhyHealth" type="ENS_0"/>
      <DO desc="Single point status" name="Proxy" type="SPS_0"/>
   </LNodeType>
   <LNodeType id="LLN0_0" lnClass="LLN0">
      <DO desc="Controllable enumerated status" name="Mod" type="ENC_1"/>
      <DO desc="Enumerated status" name="Beh" type="ENS_2"/>
      <DO desc="Enumerated status" name="Health" type="ENS_1"/>
      <DO desc="Logical Node name plate" name="NamPlt" type="LPL_0"/>
   </LNodeType>
```

```
<LNodeType id="IHMI_0" lnClass="IHMI">
   <DO name="Mod" type="INC_0"/>
   <DO name="Beh" type="INS_2"/>
   <DO name="Health" type="INS_1"/>
   <DO name="NamPlt" type="LPL_1"/>
</LNodeType>
<DOType cdc="CMV" desc="Complex measured value" id="phsC_0">
   <DA bType="Struct" dchg="true" dupd="true" fc="MX" name="cVal" type="cVal_0"/>
   <DA bType="Quality" fc="MX" name="q" qchg="true"/>
   <DA bType="Timestamp" fc="MX" name="t"/>
</DOType>
<DOType cdc="CMV" desc="Complex measured value" id="phsB_0">
   <DA bType="Struct" dchg="true" dupd="true" fc="MX" name="cVal" type="cVal_0"/>
   <DA bType="Quality" fc="MX" name="q" qchg="true"/>
   <DA bType="Timestamp" fc="MX" name="t"/>
</DOType>
<DOType cdc="CMV" desc="Complex measured value" id="phsA_0">
   <DA bType="Struct" dchg="true" dupd="true" fc="MX" name="cVal" type="cVal_0"/>
   <DA bType="Quality" fc="MX" name="q" qchg="true"/>
   <DA bType="Timestamp" fc="MX" name="t"/>
</DOType>
<DOType cdc="WYE" desc="Phase to ground related measurement values of a three phase Star"
id="WYE_0">
   <SDO desc="Complex measured value" name="phsA" type="phsA_0"/>
   <SDO desc="Complex measured value" name="phsB" type="phsB_0"/>
   <SDO desc="Complex measured value" name="phsC" type="phsC_0"/>
</DOType>
<DOType cdc="SPC" desc="Controllable single point" id="SPC_0">
   <DA bType="Enum" dchg="true" fc="CF" name="ctlModel" type="ctlModel"/>
</DOType>
<DOType cdc="DPC" desc="Controllable double point" id="DPC_0">
   <DA bType="Dbpos" dchg="true" fc="ST" name="stVal" type="Dbpos"/>
   <DA bType="Quality" fc="ST" name="q" qchg="true"/>
   <DA bType="Timestamp" fc="ST" name="t"/>
   <DA bType="Enum" dchg="true" fc="CF" name="ctlModel" type="ctlModel"/>
</DOType>
<DOType cdc="INS" desc="Integer status" id="INS_0">
   <DA bType="INT32" dchg="true" dupd="true" fc="ST" name="stVal"/>
   <DA bType="Quality" fc="ST" name="q" qchg="true"/>
   <DA bType="Timestamp" fc="ST" name="t"/>
</DOType>
<DOType cdc="SPS" desc="Single point status" id="SPS_0">
   <DA bType="BOOLEAN" dchg="true" fc="ST" name="stVal"/>
   <DA bType="Quality" fc="ST" name="q" qchg="true"/>
   <DA bType="Timestamp" fc="ST" name="t"/>
</DOType>
<DOType cdc="ENS" desc="Enumerated status" id="ENS_0">
   <DA bType="Enum" dchg="true" dupd="true" fc="ST" name="stVal" type="Mod"/>
   <DA bType="Quality" fc="ST" name="q" qchg="true"/>
   <DA bType="Timestamp" fc="ST" name="t"/>
</DOType>
<DOType cdc="ENC" desc="Controllable enumerated status" id="ENC_0">
   <DA bType="Struct" fc="CO" name="Oper" type="Oper_0"/>
   <DA bType="Enum" dchg="true" fc="ST" name="stVal" type="Mod"/>
```

```
    <DA bType="Quality" fc="ST" name="q" qchg="true"/>
    <DA bType="Timestamp" fc="ST" name="t"/>
    <DA bType="Enum" fc="CF" name="ctlModel" type="ctlModel"/>
</DOType>
<DOType cdc="ISC" desc="Integer controlled step position information" id="ISC_0">
    <DA bType="Struct" dchg="true" fc="ST" name="valWTr" type="valWTr_0"/>
    <DA bType="Quality" fc="ST" name="q" qchg="true"/>
    <DA bType="Timestamp" fc="ST" name="t"/>
    <DA bType="Enum" dchg="true" fc="CF" name="ctlModel" type="ctlModel"/>
</DOType>
<DOType cdc="APC" desc="Controllable analogue set point information" id="APC_0">
    <DA bType="Struct" fc="CO" name="Oper" type="Oper_1"/>
    <DA bType="Struct" dchg="true" fc="MX" name="mxVal" type="mxVal_0"/>
    <DA bType="Quality" fc="MX" name="q" qchg="true"/>
    <DA bType="Timestamp" fc="MX" name="t"/>
    <DA bType="Enum" fc="CF" name="ctlModel" type="ctlModel"/>
</DOType>
<DOType cdc="BSC" desc="Binary controlled step position information" id="BSC_0">
    <DA bType="Struct" dchg="true" fc="ST" name="valWTr" type="valWTr_0"/>
    <DA bType="Quality" fc="ST" name="q" qchg="true"/>
    <DA bType="Timestamp" fc="ST" name="t"/>
    <DA bType="BOOLEAN" dchg="true" fc="CF" name="persistent"/>
    <DA bType="Enum" dchg="true" fc="CF" name="ctlModel" type="ctlModel"/>
</DOType>
<DOType cdc="DPL" desc="Device name plate" id="DPL_0">
    <DA bType="VisString255" fc="DC" name="vendor"/>
</DOType>
<DOType cdc="LPL" desc="Logical Node name plate" id="LPL_0">
    <DA bType="VisString255" fc="DC" name="vendor"/>
    <DA bType="VisString255" fc="DC" name="swRev"/>
    <DA bType="VisString255" fc="DC" name="d"/>
</DOType>
<DOType cdc="ENS" desc="Enumerated status" id="ENS_1">
    <DA bType="Enum" dchg="true" fc="ST" name="stVal" type="Health"/>
    <DA bType="Quality" fc="ST" name="q" qchg="true"/>
    <DA bType="Timestamp" fc="ST" name="t"/>
</DOType>
<DOType cdc="ENS" desc="Enumerated status" id="ENS_2">
    <DA bType="Enum" dchg="true" fc="ST" name="stVal" type="Mod"/>
    <DA bType="Quality" fc="ST" name="q" qchg="true"/>
    <DA bType="Timestamp" fc="ST" name="t"/>
</DOType>
<DOType cdc="ENC" desc="Controllable enumerated status" id="ENC_1">
    <DA bType="Enum" dchg="true" fc="ST" name="stVal" type="Mod"/>
    <DA bType="Quality" fc="ST" name="q" qchg="true"/>
    <DA bType="Timestamp" fc="ST" name="t"/>
    <DA bType="Enum" fc="CF" name="ctlModel" type="ctlModel"/>
</DOType>
<DOType cdc="LPL" id="LPL_1">
    <DA bType="VisString255" fc="DC" name="vendor"/>
    <DA bType="VisString255" fc="DC" name="swRev"/>
    <DA bType="VisString255" fc="DC" name="d"/>
</DOType>
<DOType cdc="INS" id="INS_1">
```

```
    <DA bType="Enum" dchg="true" fc="ST" name="stVal" type="Health"/>
    <DA bType="Quality" fc="ST" name="q" qchg="true"/>
    <DA bType="Timestamp" fc="ST" name="t"/>
</DOType>
<DOType cdc="INS" id="INS_2">
    <DA bType="Enum" dchg="true" fc="ST" name="stVal" type="Mod"/>
    <DA bType="Quality" fc="ST" name="q" qchg="true"/>
    <DA bType="Timestamp" fc="ST" name="t"/>
</DOType>
<DOType cdc="INC" id="INC_0">
    <DA bType="Enum" dchg="true" fc="ST" name="stVal" type="Mod"/>
    <DA bType="Quality" fc="ST" name="q" qchg="true"/>
    <DA bType="Timestamp" fc="ST" name="t"/>
    <DA bType="Enum" fc="CF" name="ctlModel" type="ctlModel"/>
</DOType>
<DAType id="origin_0">
    <BDA bType="Enum" name="orCat" type="orCat"/>
    <BDA bType="Octet64" name="orIdent"/>
</DAType>
<DAType id="Oper_0">
    <BDA bType="BOOLEAN" name="ctlVal" type="Dbpos"/>
    <BDA bType="Struct" name="origin" type="origin_0"/>
    <BDA bType="INT8U" name="ctlNum"/>
    <BDA bType="Timestamp" name="T"/>
    <BDA bType="BOOLEAN" name="Test"/>
    <BDA bType="Check" name="Check"/>
</DAType>
<DAType id="valWTr_0">
    <BDA bType="INT8" name="posVal"/>
</DAType>
<DAType id="mxVal_0">
    <BDA bType="INT32" name="i"/>
    <BDA bType="FLOAT32" name="f"/>
</DAType>
<DAType id="Oper_1">
    <BDA bType="FLOAT32" name="ctlVal"/>
    <BDA bType="Struct" name="origin" type="origin_0"/>
    <BDA bType="INT8U" name="ctlNum"/>
    <BDA bType="Timestamp" name="T"/>
    <BDA bType="BOOLEAN" name="Test"/>
    <BDA bType="Check" name="Check"/>
</DAType>
<DAType id="mag_0">
    <BDA bType="FLOAT32" name="f"/>
</DAType>
<DAType id="cVal_0">
    <BDA bType="Struct" name="mag" type="mag_0"/>
</DAType>
<EnumType id="ctlModel">
    <EnumVal ord="0">status-only</EnumVal>
    <EnumVal ord="1">direct-with-normal-security</EnumVal>
    <EnumVal ord="2">sbo-with-normal-security</EnumVal>
    <EnumVal ord="3">direct-with-enhanced-security</EnumVal>
    <EnumVal ord="4">sbo-with-enhanced-security</EnumVal>
```

```
      </EnumType>
      <EnumType id="Dbpos">
         <EnumVal ord="0">intermediate</EnumVal>
         <EnumVal ord="1">off</EnumVal>
         <EnumVal ord="2">on</EnumVal>
         <EnumVal ord="3">bad</EnumVal>
      </EnumType>
      <EnumType id="Mod">
         <EnumVal ord="1">on</EnumVal>
         <EnumVal ord="2">on-blocked</EnumVal>
         <EnumVal ord="3">test</EnumVal>
         <EnumVal ord="4">test/blocked</EnumVal>
         <EnumVal ord="5">off</EnumVal>
      </EnumType>
      <EnumType id="orCat">
         <EnumVal ord="0">not-supported</EnumVal>
         <EnumVal ord="1">bay-control</EnumVal>
         <EnumVal ord="2">station-control</EnumVal>
         <EnumVal ord="3">remote-control</EnumVal>
         <EnumVal ord="4">automatic-bay</EnumVal>
         <EnumVal ord="5">automatic-station</EnumVal>
         <EnumVal ord="6">automatic-remote</EnumVal>
         <EnumVal ord="7">maintenance</EnumVal>
         <EnumVal ord="8">process</EnumVal>
      </EnumType>
      <EnumType id="Health">
         <EnumVal ord="1">Ok</EnumVal>
         <EnumVal ord="2">Warning</EnumVal>
         <EnumVal ord="3">Alarm</EnumVal>
      </EnumType>
   </DataTypeTemplates>
</SCL>
```

TRITA TRITA-EE 2016:164
ISSN 1653-5146