

# **HMI Configuration of a Substation**

Robin Malmström

Degree Thesis for Bachelor of Engineering

Degree Programme in Electrical Engineering and Automation

Vaasa, 2021

## **BACHELOR'S THESIS**

Author: Robin Malmström  
Degree Programme: Electrical Engineering and Automation  
Specialization: Automation Technology  
Supervisor(s): Ronnie Sundsten, Ari Honkonen

Title: HMI Configuration of a Substation

---

Date: 11.5.2021

Number of pages: 33

---

### **Abstract**

This thesis was made in cooperation with VEO Oy. The purpose of this thesis was to test a program called Zenon by creating an HMI of a substation. The configuration process of the HMI is explained and described.

The theoretical part consists of general information about substation functionality and key components used. After the establishment of a general understanding of a substations functionality, the configuration process of the HMI is described. The testing of functionality and communication was done virtually with a program called SCL-Runner to simulate IEDs. Both IEC61850 and IEC61870 communication were tested.

The result of this thesis is a general guide on how to use the software and functionality used in a substation HMI. A more advanced configuration guide with tips and better visualization was provided to VEO.

---

Language: English

Key words: Substation, Substation Automation , HMI

---

## EXAMENSARBETE

Författare: Robin Malmström  
Utbildning och ort: El- och automationsteknik, Vasa  
Inriktningsalternativ: Automationsteknik  
Handledare: Ronnie Sundsten, Ari Honkonen

Titel: HMI-konfiguration av en elstation

---

Datum 11.5.2021

Sidantal: 33

---

### Abstrakt

Detta examensarbete gjordes i samarbete med VEO Oy. Syftet med detta examensarbetet var att testa ett program som heter Zenon genom att skapa en HMI för en elstation. Konfigurationsprocessen för HMI:n förklaras och beskrivs.

Den teoretiska delen består av allmän information om elstationsfunktionalitet och viktiga komponenter som används. Efter att en allmän förståelse av elstationsfunktionalitet är skapad beskrivs konfigurationsprocessen för HMI:n. Test av funktionalitet och kommunikation gjordes med ett program som heter SCL-Runner för att simulera IED:n. Både IEC61850- och IEC61870-kommunikation testades.

Resultatet av detta examensarbete blev en allmän guide till hur man använder programvaran för att skapa en HMI till en elstation. En mer avancerad konfigurationsguide med tips och bättre visualisering levererades till VEO.

---

Språk: engelska

Nyckelord: elstation, automation, HMI

---

## OPINNÄYTETYÖ

Tekijä: Robin Malmström  
Koulutus ja paikkakunta: Sähkö- ja automaatiotekniikka, Vaasa  
Suuntautumisvaihtoehto: Automaatiotekniikka  
Ohjaajat: Ronnie Sundsten, Ari Honkonen

Nimike: Sähköaseman HMI-konfigurointi

---

Päivämäärä: 11.5.2021

Sivumäärä: 33

---

### Tiivistelmä

Opinnäytetyö tehtiin yhteistyössä VEO Oy:n kanssa. Tämän työn tarkoituksena oli testata Zenon-nimistä ohjelmaa luomalla sähköaseman käyttöliittymä. Käyttöliittymän konfigurointiprosessi selitetään ja kuvataan.

Teoreettinen osa koostuu yleisestä tiedosta sähköaseman toiminnallisuudesta ja käytetyistä avainkomponenteista. Asemien toiminnallisuuden yleisen ymmärryksen luomisen jälkeen kuvataan käyttöliittymän konfigurointiprosessi. Toiminnallisuuden ja viestinnän testaus tehtiin SCL-Runner-nimisellä ohjelmalla, jolla simuloitiin IED:t. Sekä IEC61850- että IEC61870-tiedonsiirto testattiin.

Tämän opinnäytetyön tulos on yleinen opas sähköaseman käyttöliittymän ohjelmoimiseen ja ohjelman toiminnallisuuden kuvaus. VEO:lle toimitettiin edistyneempi opas konfigurointiprosessista, jossa on vinkkejä konfigurointiin ja parempi visualisointi käyttäen enemmän kuvia.

---

Kieli: englanti

Avainsanat: sähköasema, automaatio, HMI

---

# Innehållsförteckning

1	Introduction .....	1
1.1	Objective .....	1
1.2	VEO Oy .....	1
2	Theory.....	2
2.1	Power grids.....	2
2.2	Substations.....	3
2.2.1	Switchyard.....	4
2.2.2	Circuit breaker .....	5
2.2.3	Disconnecter.....	6
2.2.4	Medium-voltage Switchgear.....	7
2.2.5	Transformer.....	7
2.3	Substation Automation .....	8
2.3.1	Station controller.....	8
2.3.2	IED .....	9
2.3.3	Supervisory Control and Data Acquisition .....	9
2.3.4	HMI.....	9
2.3.5	RTU.....	10
2.3.6	Communication protocols.....	11
3	Zenon.....	12
3.1	Zenon editor .....	12
3.2	Zenon runtime .....	13
4	HMI configuration .....	13
4.1	Editor workbench .....	13
4.2	Creation of a workspace and project.....	14
4.3	Driver configuration.....	15
4.4	Importing variables .....	16
4.5	Frames .....	17
4.6	Screens.....	17
4.7	Combined elements.....	18
4.8	Symbols .....	19
4.9	Functions .....	21
4.10	Chronological event list.....	21
4.11	Alarm list .....	22
4.12	Automatic line coloring .....	23
4.13	Reaction matrices .....	25
4.14	Command processing .....	27
4.14.1	Command processing screen.....	27

4.14.2	Command processing configuration.....	28
4.15	Language files.....	29
4.16	Context menus.....	30
5	Discussion and result.....	31
6	References.....	33

# 1 Introduction

Substations are a vital part of a working society as it is hard to imagine a day without electricity in modern times. Today, most components use electricity as a power source and as the technology keeps expanding, more and more is demanded from substations.

Software to efficiently configure substation HMIs is a key part to save time and costs on a project. Using poor software to configure station communication and user interfaces can lead to unnecessary project costs.

## 1.1 Objective

The main objective for this thesis is to research and test the functionality of a program called Zenon. Zenon is a software used to create user interfaces for processes, among other things. In this thesis we are creating an HMI of a substation and showing how the configuration is done. This should help the company evaluate the software and provide an opportunity for alternative implementation of the HMI.

## 1.2 VEO Oy

VEO provides three main services Power generation, Power distribution, and Power utilization. Power generation unit offers solutions for a variety of power plants such as hydro, wind, and thermal plants to engine and hybrid power plants. Power utilization does industrial electrification and automation.

This thesis was made for the Power distribution business unit. This unit makes substations and switchgear for transmission and distribution networks. These substation buildings are done in the factory in Vaasa. In the factory, they test the equipment as far as it is possible in a FAT (Factory Acceptance Test). After that, the cubicles are delivered to sites where the SAT (Site Acceptance Test) is done.

Vaasa Engineering Oy is a big Nordic company that was founded in 1989, in 2012 they changed the name to VEO Oy. It was the first energy technology company that settled down in Runsor Vaasa which is the headquarters today. 2002-2009 was a great period for growth at VEO, in this timeframe they doubled their employees and the turnover went from 25 to

80 M€. VEO has offices in Norway, Sweden, United Kingdom, and several in Finland. Today VEO has 485 employees and a turnover of 124 M€ in 2019. (Holma, ei pvm)

## **2 Theory**

This chapter will give general information on substation functionality. What a substation is, what devices are involved, and equipment. Having a basic understanding of the substation is important when you configure an HMI.

### **2.1 Power grids**

In Finland, Fingrid is the main power distributor with 400 kV, 220 kV, and 110 kV grids. Substations are connected to these and electricity is then distributed to end-users from substations. These substations convert the high voltage into standardized and non-hazardous voltage levels. Most commonly it is converted into 36 kV or 20 kV.



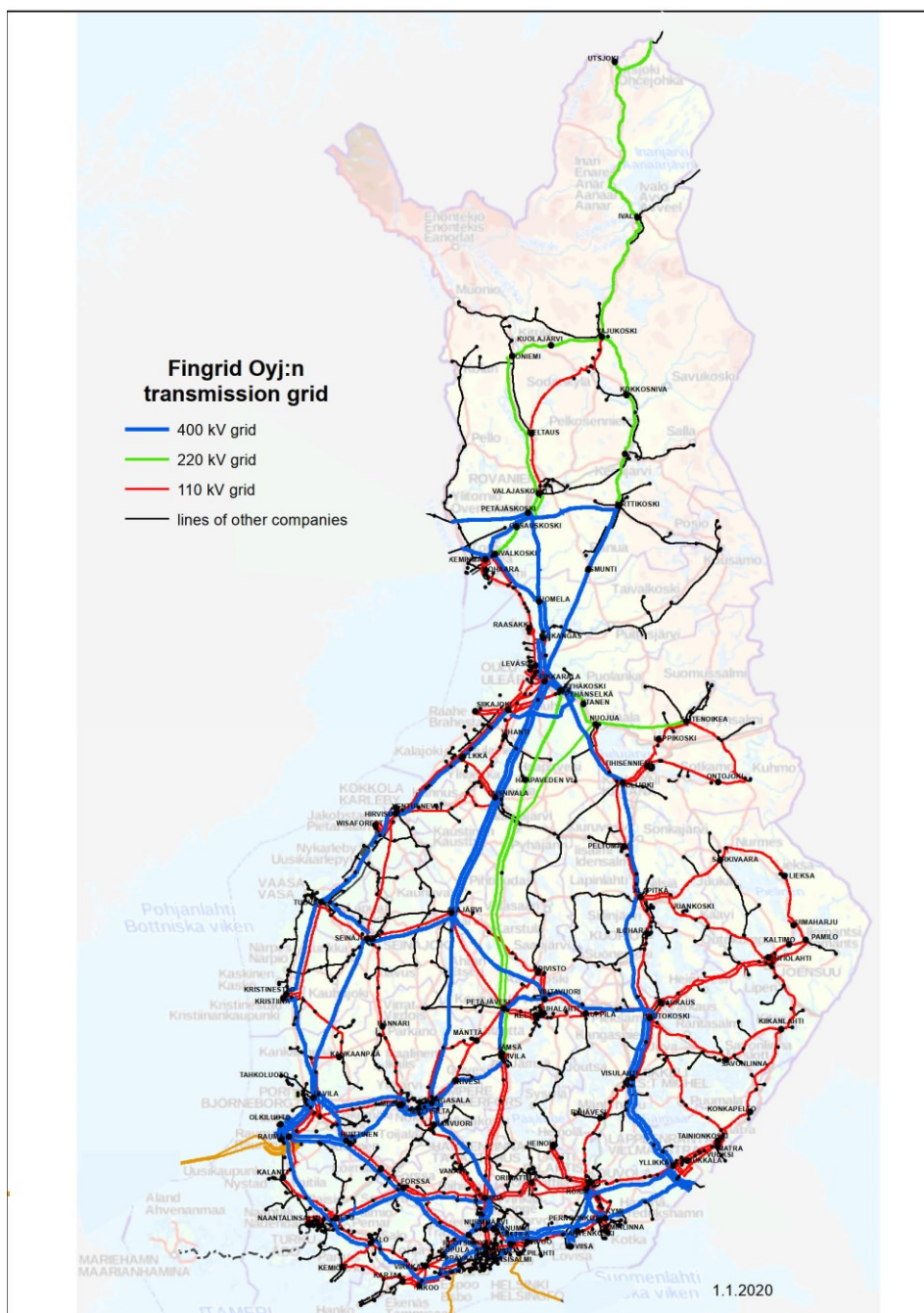


Figure 1 - Fingrids transmission grid (FINGRID, ei pvm)

## 2.2 Substations

All electric substations consist of high voltage apparatuses called primary equipment. The primary equipment consists of circuit breakers, disconnectors, and earthing switches in a switchyard. The main function of these is to maintain or interrupt the transmission of electricity.

To control and change the state of the primary equipment low voltage apparatuses are used. These apparatuses are called secondary equipment and are operated by operators. Secondary equipment monitors the primary equipment and when necessary protects it from malfunctions. Malfunctions that can occur for example are overcurrent, undercurrent, frequency drops, etc. In situations like this, the secondary equipment can control the primary equipment that results in a safe environment. These apparatuses that send commands to the secondary equipment are called IEDs and are used for protection.

Substations take care of all data acquisition processes, control, monitoring, and alarming functions of the primary equipment. The control and monitoring are shown to the operator by visual graphs, an alarm list, an event list and the primary circuit screen. The graphs and SLD must be user-friendly for ease of use. In Finland, a substation does not always have an HMI. The control is usually handled by SCADA, but in addition to the mimicry of the relays, a local HMI can be used for local operations. By clicking on the breakers/disconnector in the SLD screen, opens a command window where you can change the state of that exact breaker/disconnector. For operations like this two-step verification is required where you need to confirm the operation you are about to make. (Padilla, 2015)

### **2.2.1 Switchyard**

The switchyard is a fenced area around the substation where most of the primary equipment exists. Switchyard size is dependent on the substation extent and type of insulation media. The main two insulation media used are air or gas. Gas-insulated switchgear is a compact metal encapsulated switchgear that is mostly used when space is limited. Air-insulated takes more space but it is cheaper than the gas-insulated switchgear (GIS). (ABB, n.d.)

Switchgears can be operated manually on the yard by an operator or with electric motors. 110 VDC is supplied to these motors and other state indications and interlocking functions. These motors need to be operational even if a power failure happens in the substation. This is why the substation has a backup power source that makes it possible to operate the switchyard without the main power.

### **2.2.2 Circuit breaker**

The circuit breaker is the most vital component in a substation. Breaking the current when needed and safely is a top priority for a functional substation. The breaker consists of 3 identical poles for each phase. Each breaker pole has one or more sealed chambers where the current interruption process takes place. These breakers can disconnect the current in two circumstances:

- normal load
- short circuit conditions

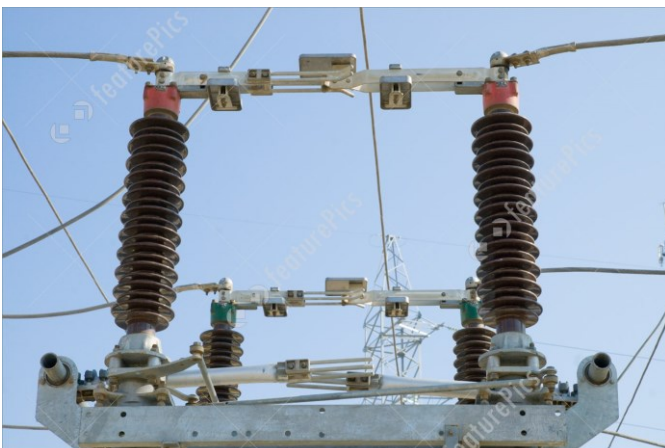
When interrupting the current an arc occurs. This must be extinguished quickly, safely, and completely to eliminate the current flow. Most breakers that are rated 52 kV or more use SF6 gas in the interrupting chambers when the pressure exceeds the limit. When the breaker opens which usually should be around 50 ms, the arc expands and gets exposed to the SF6 gas. This gas cools it down and extinguishes the arc. (Padilla, 2015)



**Figure 2 - Circuit breaker with SF6 gas chambers**

### **2.2.3 Disconnecter**

A disconnecter is a switch that provides visible air-gap isolation from equipment located on the switchyard. Disconnecters are used to disconnect line segments for maintenance and repair tasks. Disconnecters cannot be operated unless the associated circuit breaker is open, as it creates an arc that the disconnecter can't handle.



**Figure 3 - Disconnecter in a closed position**

#### 2.2.4 Medium-voltage Switchgear

Medium voltage switchgear is the distribution link between the substation and the consumer. It is equipped with a breaker, earthing switch, disconnecter, and an IED. Substations usually have more than 10 of these pieces of equipment depending on the size of the station and consumers.

VEO has its medium voltage switchgear called Vector. Vector is an air-insulated switchgear with a rated voltage of 12 kV and 24 kV. Vector is equipped with a vacuum circuit breaker and it is designed for indoor systems. Most substations that VEO implements are rated 110/20 kV so Vector is the perfect solution for these substations.



Figure 4 - VEO's medium voltage switchgear called Vector (Vector, ei pvm)

#### 2.2.5 Transformer

Transformers are located in the switchyard and their function is to transform the voltage from high to medium before it enters the distribution substation. This can be the other way around if it is a step-up transmission substation used to transport the voltage. High voltage is used to send the electricity long distances that allow minimal loss. And because of the high voltage, it must be downsized to a less hazardous level. This allows the use of

shorter safety distances and thinner insulation. Deeper knowledge is not required for this thesis. More can be read about working principles in the book made by Evelio Padilla.



Figure 5 - Transformer

## 2.3 Substation Automation

This chapter goes over the automation of a substation and components used. To automate a substation, devices are required to communicate with each other and to send commands to equipment that executes them. Without these intelligent devices, it would be impossible to monitor and control remotely. The focus is to give general information about the automation in a station as the main objective of this thesis is to show the configuration process.

### 2.3.1 Station controller

The station controller is an industrial computer running a network system for all the protection relays and other IEDs. A few other common names for a station controller are central substation processor, front end, or station computer. This controller communicates with IEDs, store parameter data, records events with adequate time resolution (e.g., less than 1 ms). These events are forwarded to the HMI where the event list and alarm list exist. (Padilla, 2015)

### 2.3.2 IED

IED which is an abbreviation from intelligent electronic device, is a modern device that can issue control commands, such as tripping circuit breakers if they sense voltage, current, or frequency anomalies. To put it simply, IEDs in a substation are the eyes, ears, and hands of the automation system. The main IED that substations are equipped with is the protective relay. Protective relays measure and monitor the circuit's current, voltage and frequency, etc. Usually, every medium voltage switchgear unit has one or several IEDs.



Figure 6 - IED

### 2.3.3 Supervisory Control and Data Acquisition

Supervisory Control and Data Acquisition aka. SCADA is a widely used control center in the industry. From SCADA you can remotely control and operate the substations. In today's age substations are most likely operated from SCADA. It is inconvenient going to these remote locations where substations usually are located, just to disconnect or connect a breaker, etc.

### 2.3.4 HMI

Human Machine Interface as the name tells you is an interface where operators monitor and control the system locally. Developing an HMI is crucial and must be user-friendly

otherwise controlling and monitoring will be hard. In a substation, the HMI consists of a single line diagram (SLD), Event list, and Alarm list.

SLD has all breakers and disconnectors that can be operated by clicking on them. Voltage, current, and frequency are measured and displayed in the same diagram. This gives a good monitoring place to monitor the station. IED's state of operations is also shown (Remote/local) which indicates if the IED is operatable locally or remotely.

The event list refers to a list where all events that have occurred in the system. This includes all changes that happen in the substation. Logging of events is a very important feature that helps to solve why there were some anomalies etc. In the event list and alarm list timestamps are the most crucial information that needs to be correct. Without the right timestamp, it can be very hard to pin down what caused for example an IED to trip a breaker.

The alarm list is where all the alarms are listed. These alarms are usually more important and might require immediate attention/acknowledgment. In the engineering stage of these lists, the events must be shown in chronological order. An SNTP server (Simple Network Time Protocol) is used for this to synchronize all the IEDs and other components connected to the network.

### **2.3.5 RTU**

A remote terminal unit is a unit that communicates with SCADA. RTU can receive messages from SCADA such as control messages that the RTU interprets and initiate action to the field devices. After the action is done RTU sends a completion message to SCADA. RTU can be described as a station controller as mentioned earlier. (Mini S. Thomas, 2015)

RTU is used to filter traffic to the master station as the IEDs have unnecessary traffic in the IEC61850 communication link. Only the most vital process points are forwarded to the master station when a change occurs. RTU can send information to one or several masters and information sent can be configured in the RTU. For example, some information on a station is sent to one master and other information to another master. All connections to IEDs are centralized to the RTU so only one link to the station is required. This makes it



easier for the SCADA system to have connections to several stations as only one link is required per station. Otherwise, you would need a communication link to every IED in the station. RTU can convert communication protocols for example from IEC61850 to IEC61870-5-104.



Figure 7 - RTU

### 2.3.6 Communication protocols

To control substations remotely the communication becomes the most important factor. Communication standards have been created to help with this. There are essentially two standards for the communication that is used in a substation. These standards must be followed when making the automation system.

- IEC 61850
- IEC 6070-5-104

IEC 61850 target is to improve the automation of the electrical substation. This is done by making sure you can integrate different manufacturer's equipment into the substation. Using technologies that reduce the cost in engineering and wiring. And as the last seeking for improvements for maintenance tasks and commissioning. (ENSOTEST, n.d.)

IEC 6070-5-104 uses the TCP/IP technology defined by IEC 6070-5-101 to accomplish remote-control tasks. (ENSOTEST, n.d.)

## 3 Zenon

Zenon is an industrial automation software developed by COPA-DATA. COPA-DATA was founded in 1987 by Thomas Punzerberg in Austria. Today COPA-DATA has 285 employees and 51 million € revenue in 2019. COPA-DATA is a global company with local sales & support in 51 countries. Most of these countries are in Europe. (COPA-DATA, n.d.)

The software was developed to make life easier for people in the industrial and infrastructure environment. There are two different versions of the software, standard and energy edition. One big focus in the energy edition is that a lot of settings are centralized. By having centralized settings makes it faster and more efficient to create projects/solutions. This is done by reusing symbols and changing the centralized settings. This way you don't have to make new symbols all the time. (Punzerberger, n.d.)

Zenon supports all common communication protocols in the energy industry. A list of the supported protocols is shown below:

- IEC 61850 Client/Server and GOOSE
- IEC 61850 Edition 2 (certified by TÜV Süd)
- IEC 61400-25
- IEC 60870-5 (101/103/104)
- DNP3
- IEC 62056-21
- OPC UA
- Modbus
- IEEE C37.118 (Synchrophasor)
- IEC 61850-90-5

### 3.1 Zenon editor

Zenon editor is the engineering tool. It is used to configure your projects, create user interfaces (HMI), configure communication protocols, etc. The editor is available in 32-bit and 64-bit versions.

## 3.2 Zenon runtime

Zenon runtime is where the projects and solutions are running. So essentially runtime is a user interface where operators can monitor and control the system. This runtime is located locally on a computer in the substation that is connected to the network with all IEDs.

## 4 HMI configuration

The following configuration steps describe the procedures used to accomplish the goal of this theses, to create an HMI of a substation.

### 4.1 Editor workbench

To understand the following chapter a picture of the editor workbench has to be shown and explained briefly. This makes it easier for the reader to grasp where the windows exist, etc.

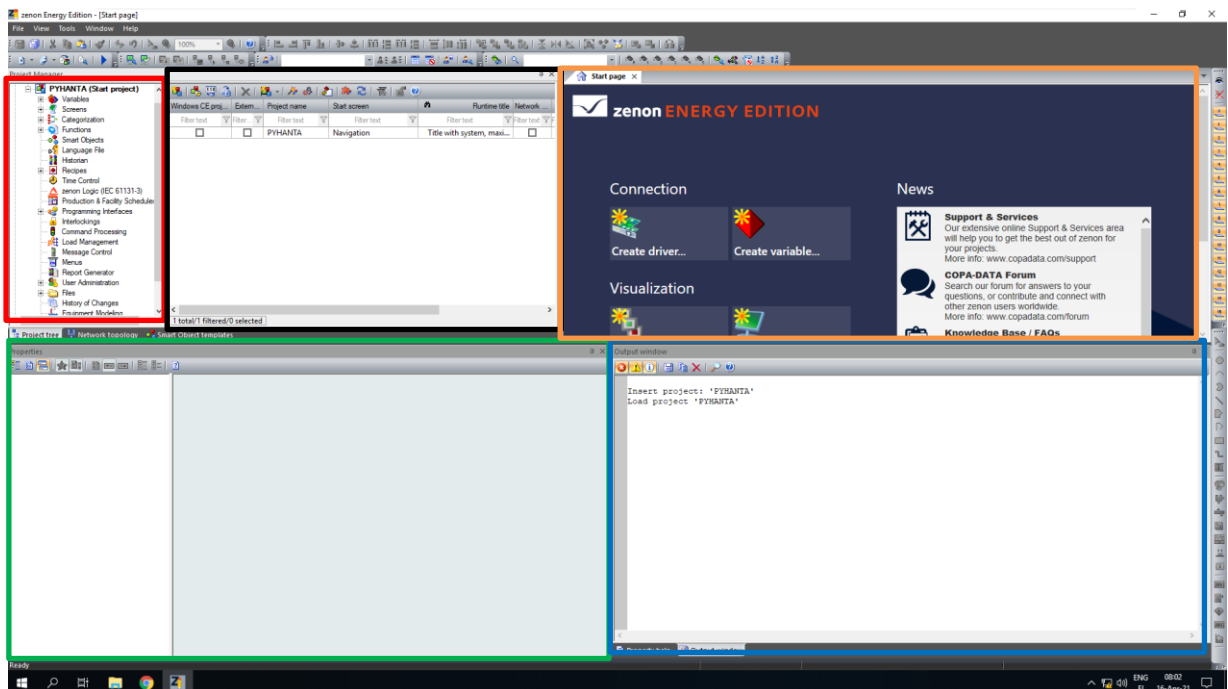


Figure 8 - Editor workbench

- Red box – Project tree with all categories and functions usable in Zenon.
- Green box – Properties window used for most configuration parameters on objects.

- Blue box – Output window where error messages and other notes on the project are shown.
- Black box – a better view of the category opened from the project tree. For example, opening variables show all variables made in the project.
- Orange box – This window is used to edit the screens, frames, and symbols. To open a screen you navigate from the project tree and open screens and then open the screen you want from the black window and edit the screen on the orange window.

## 4.2 Creation of a workspace and project

When opening Zenon editor for the first time a workspace needs to be created. Workspace is a container where all the projects are stored. In the workspace, all the components of a project are administrated. There are a few ways to create the workspace. By going to File in the upper left corner finding workspace which then asks to open or create a new workspace. The other way is to left-click with the mouse in the project manager window at the text “Workspace: ‘empty’(0)”, and then choosing a new workspace. This pops up a window where the workspace can be named and a destination path can be specified. Making the workspace is not necessarily required, because when creating a project it automatically creates a workspace for that project if no workspace exists.

Creating a project is similar to creating a workspace. By left-clicking on the workspace and choosing “a new project...”. There are two different types of projects that can be chosen, standard and global. The difference is that only one global project is allowed in a workspace. This global project can be used to store styles, colors, symbols, alarms, equipment groups, etc. When changing settings in the global project, this automatically changes in all the other standard projects that are in the workspace.

After the creation of a project, a project wizard opens which can be used to create a basic project with variables, screens, frames, etc. (Demo project). To get a clean project with nothing in it, press close. Editing/configuring of a project is only available in the active project. To change the active project left click on the project in the project tree and select “Activate project”.

### 4.3 Driver configuration

Drivers are the communication link between the software and the devices. In this case, the devices are IEDs located in the substation. To communicate with these IEDs, configure the driver so it can access the hardware.

By default, there are 3 drivers which are free of charge. Variables based on these drivers are not counted for licensed I/O:s :

- **Driver for internal variables** – These can be defined as failure-proof variables (hard disk data).
- **Driver for mathematics variables** – used for the calculation of mathematical functions or operating hours counters.
- **Driver for system variables** – Variables for monitoring and controlling the hardware, network, and other project-specific properties.

Drivers are located in the project tree under variables. To create a new driver, left-click on driver and select new driver straight from the project tree. An alternative way is to click on drivers in the project tree which will open a window for all existing drivers in the project. In the upper part of the window is a button for new driver.

After pressing new driver, a pop-up window opens where the driver is chosen. IEC61850 and IEC8750-5-104 drivers are used for communication in electrical substations. After choosing the driver a text field with a pre-defined name for the driver is created. Rename if it is necessary and press ok.

Now another pop-up for configuration of the driver created opens. No changes should be made in the general and basic settings. In “connections” configure a server where the driver communicates to. In this case, the servers refer to IED’s and a server for each device must be created here. Create a server with the button new, set server name, IP-address of the hardware (IED), and assign datasets. The driver is now configured, importing variables can now be done.

Figure 9 - Driver connection configuration

#### 4.4 Importing variables

Importing variables can be done in multiple ways. The most common is with XML files, these files are generated from other engineering tools used to configure the RTU. With the XML file, all variables and reaction matrices can be imported to Zenon. Another way of importing variables is with CID files (Configured IED Description). CID files are individual IED configuration files. The drawback of importing from these types of files is that all variables have to be selected that should be imported. The third option is to import directly from the hardware. This requires a connection to the IED. Selection of variables is also required with this method.

The XML import is done in the variable window, where the XML import button exists. The other two are done in the driver window. When importing from the driver window choose either PLC browsing or File browsing. PLC browsing is used to import from hardware, File browsing is used to import from CID files.

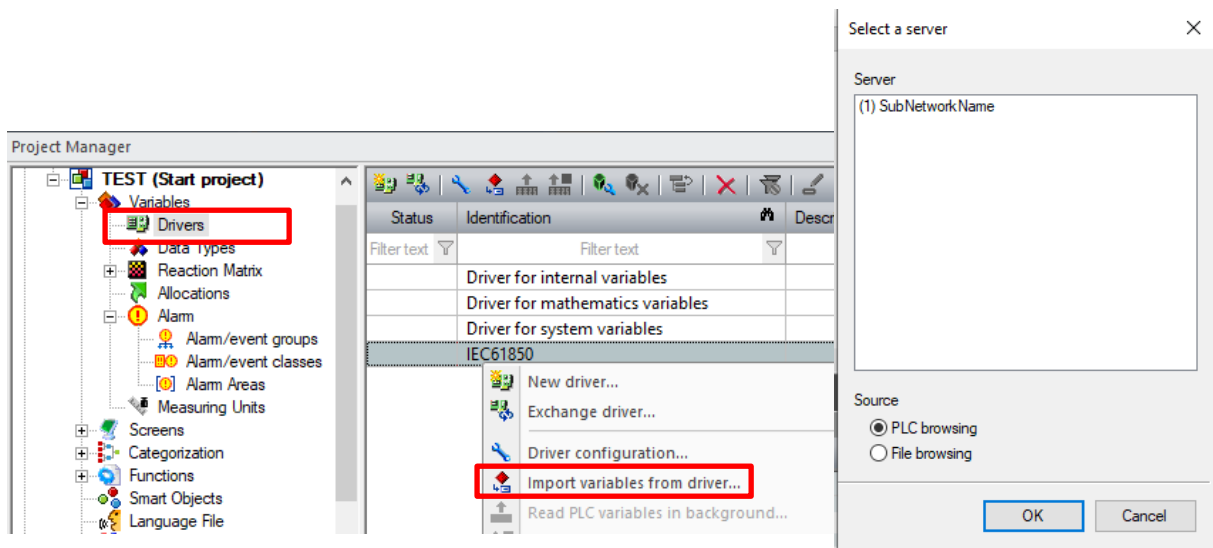


Figure 10 - Variable import with CID file or directly from IED

## 4.5 Frames

Frames determine the general window properties such as position, size, the appearance of a screen. Every screen has a frame assigned which determines the screen size. The screen category is found in the project tree and under screens there are frames. Left-click on frames and select “create new frame”. When created double click on the frame and a frame editor open where positioning of your frame is done.

In the properties of a frame, the border is set. Select border type “size fixed” and mark title and system menu for the command processing frame. This eliminates the need to make a button for closing the command screen. As there is implemented a system menu for the screens made with this frame.

## 4.6 Screens

Screens are a central element of a project. They display the configured equipment, inform, and provide data to the user interface. The screens consist of vector elements or dynamic elements. These are linked with variables or functions that show the data in variables etc.

Zenon offers many pre-defined screen types, such as Chronological Event list, Alarm list, Command processing, etc. The advantage of these screen types is that the desired

functionality is connected to the screen type. When creating a screen with a specific screen type, immediately a variety of pre-defined functions are linked to the associated screen type.

Creating screens is done by right-clicking on screens and then selecting “new screen”. A window pops up where the selection of screen type, frame, and naming of the screen is done. The screen editor opens when double-clicking the screen in the project manager window.

## 4.7 Combined elements

The combined element is dynamic. This means that it can be used universally and can adapt to the most varied graphic characteristics. Variables are linked to these elements and depending on the different values the variable can have, you make different graphics that are displayed with the different values. For substations, the breakers and disconnectors are made with these combined elements to show if it is on or off in the single line diagram.

Creating a combined element can be done in multiple ways. On the right side of the editor, a vertical small bar with different icons for all sorts of elements. A few central elements used when configuring an HMI for a substation are dynamic text, combined element, line, and static text. The difference between dynamic- and static text is that static text cannot be changed with variable values. This means that the text string configured cannot change by itself. The dynamic text has a variable assigned to it and has multiple different display texts that are defined in the properties. So, for example, if a dynamic text is created that shows the variable value it changes when the variable value changes.

The other way of creating is to click on the taskbar “elements” and choosing what element to add. The screen editor has to be open for the elements in the taskbar to be accessible. Once an element is clicked on in the taskbar drag a box in the editor to create the element. Depending on the element a pop-up might come where a variable for that element is assigned.

Creating graphics for different statuses in a combined element is done in the properties under “representation”. To open the configuration window click on the text that says “click



here - >” next to Configuration and test. The configuration window is used to define different statuses and choosing symbols that are displayed when that variable status is active. New conditions are created with the button new, and all conditions should have different symbols assigned, to visualize the status of the variable. There is a default value that cannot be deleted, and it should not be used when creating breakers or disconnectors. Putting a symbol to the default value is advisable to help position the element on the screen, without the symbol it is just an empty box in the screen editor.

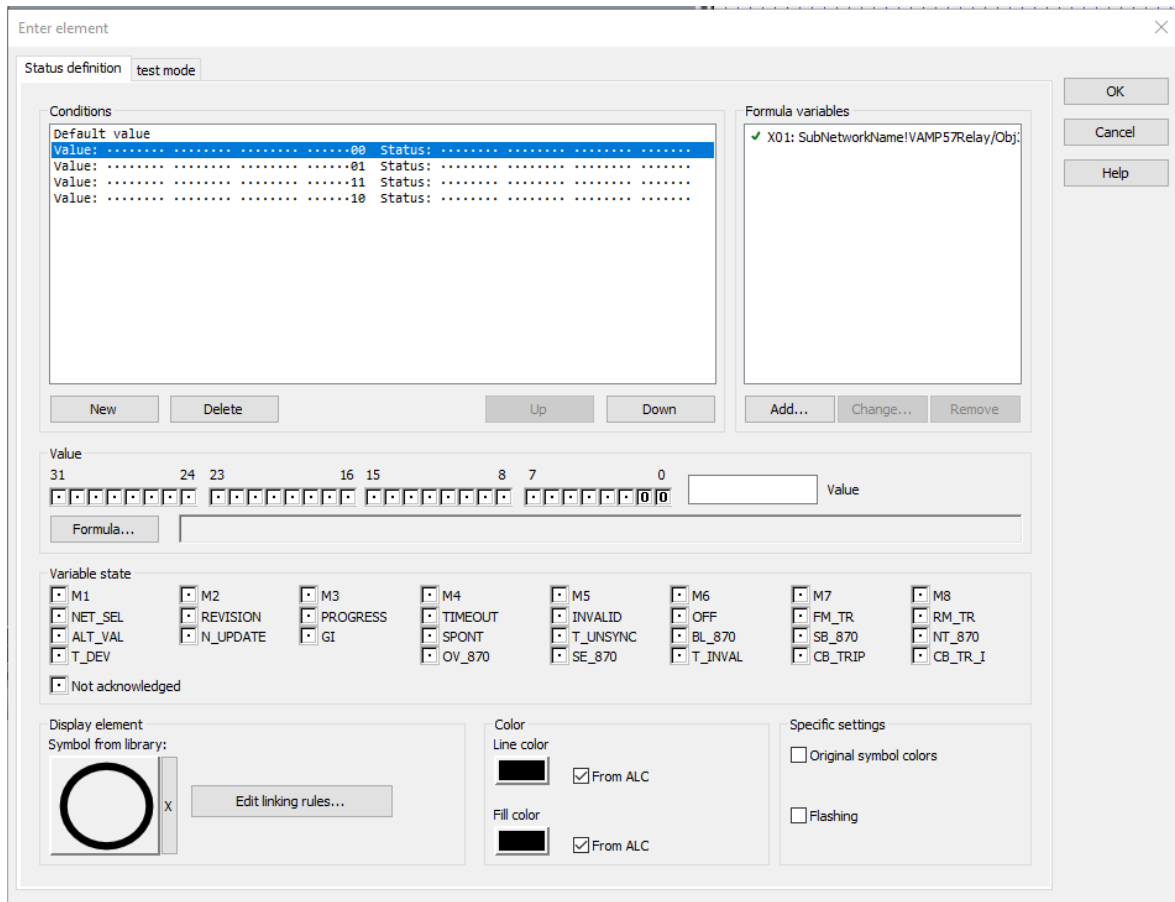


Figure 11 - Combined element configuration window. ( settings for a disconnector)

## 4.8 Symbols

Symbols are available in all projects. The general symbol library is located in the project tree as the last category. There are default symbols every project has access to. Creating symbols is done in the project tree under screens where the symbol library for the active project exists.

Creating a combined element for each breaker and disconnecter would be bad and time-consuming. One option is to copy and paste the combined element after creating one. By creating an element group out of a breaker allows it to be stored in the symbol library. And once it is stored as a symbol it can be dragged to the screen editor to make a new breaker. The new breaker has all the same settings as the original breaker configured. All that has to be done is the linking of a different variable. Afterward, changes made in the symbol are automatically forwarded to all existing symbols that were made using it, making it superior to the copy-paste method. Where the property to make changes to all the breakers doesn't exist because no link has been made between the elements with the copy-paste method. Marking all the breakers allows changes to be made to all of them, which should achieve the same thing, although not all properties can be changed with this method.

After testing and trying the different methods the most convenient method that was found out was the copy-paste. By creating a symbol out of a combined element requires a substitution of the variable that is defined in the symbol. This takes time and is much slower than the copy-paste and then choosing the correct variable in properties. The substitution window opens when dragging the symbol to the screen editor where the substitution is configured. The substitution is time-consuming and typos can happen easily. This might come down to personal preference but in my opinion, copy-paste is best.

By the same logic described earlier, storing a whole bay as a symbol is possible. The drawback of making a whole bay as an element is that changes can't be made to one specific bay afterward. For example, if you have 5 bays and modifying one of the 5 bays is needed, destruction of the link to the symbol library has to be done. This can be done by right-clicking on the symbol and choosing Symbol → Resolve. After resolving, making changes to the resolved bay will not make changes to the other four bays.

An element group is created by marking the objects, right-clicking, and selecting symbol → create element group. Once the element group is created, insert it into the symbol library. Right-click on the element group and selecting symbol → insert to library.

## 4.9 Functions

All functions used in a project are based on the existing system functions. These are pre-defined macros that are easy to use and parameterized by the user. Functions are needed for example to navigate to another screen with a screen switch function. Functions can be linked to buttons and when the button is pressed the function executes. Scripts can also be made that executes functions in a sequence.

## 4.10 Chronological event list

Chronological Event List (CEL) is used to collect information about the process events, system events that are displayed in a list. A screen has to be made and again there is a predefined CEL screen type that gives all the necessary objects. A button has to be created where linking of a function "screen switch" is used to access the CEL screen. When creating the function "screen switch" a window opens where the configuration of the CEL filters are defined. On the sheet Time, no time filter should be used. In the column sheet, determine what information you want in your list by marking them. The width can be edited, -1 means that the area is defined by the information length. By marking "Sort descending" the list shows the most recent event on top. Description text can be edited, and it is the label text showing in the list. For example, a column with Time received and the text to be in Finnish change the description text. This text is then the label for "time received" in the list.

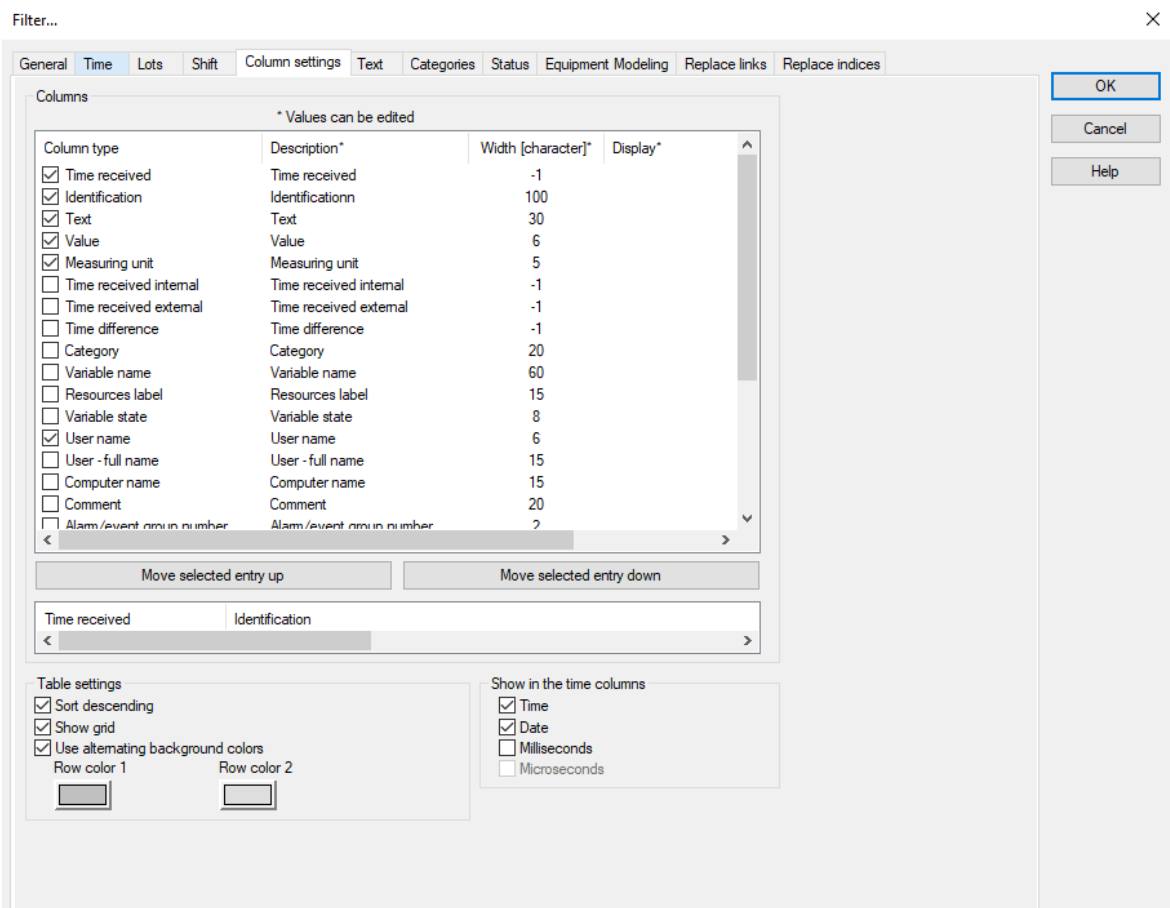


Figure 12 - CEL settings for what columns used in the list

## 4.11 Alarm list

The alarm list is where all the configured alarms pop up. There is a predefined Alarm Message List screen type that is usable. A function “screen switch” must be created to access the screen. In the function, configure the alarm list filters, similar to the CEL filters. Essential information on the alarm list is time received, time cleared, and time acknowledged.

The alarm sound is by default the normal windows sound for notifications. To use different sounds for alarms create a function called “Play audio file”. After the creation of the function link the function to an alarm/event class. And when an alarm or event that is in that event class occurs the sound plays once.

There is also an alarm status line that can be used. This line appears on any screen whenever an alarm occurs. This line shows the most recent alarm with a timestamp. To get

rid of the alarm line navigate to the alarm screen and acknowledge the alarm. To enable the alarm line, go into the general properties of a project, find the category Alarm Message List, and mark "Status line active". After it is activated an alarm status line frame is created that can be positioned on the screen.

## 4.12 Automatic line coloring

Automatic line coloring (ALC) makes it possible to color lines depending on process status. This allows a graphical view of where the current is transferred. Combined elements are used as process elements to show if the current is transferred through a breaker or not. A source is required to get the color. To define a source, click on the project in the project tree to open the general properties of the project. Under ALC properties select ALC configuration. Ten pre-defined system sources are in use and reserved for updates etc. Create a new source and select a color.

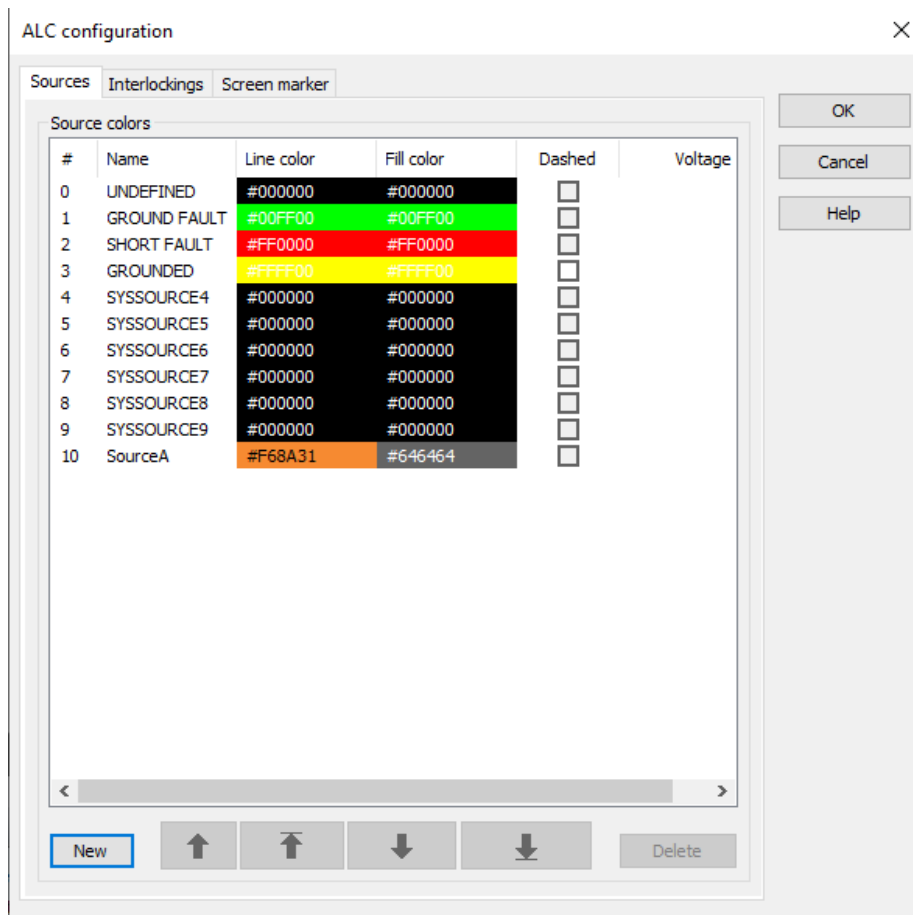


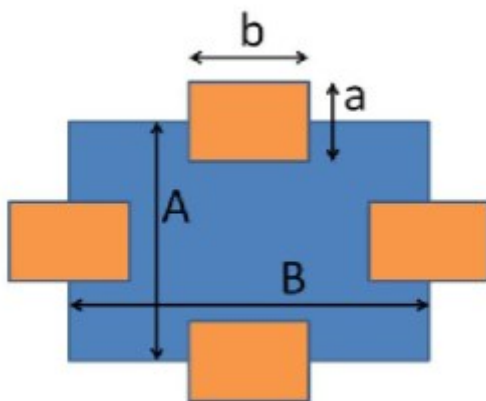
Figure 13 - Color configuration for ALC

Lines are used for forwarding the color to breakers and disconnectors. The lines must have a rotation angle of 0 for the ALC to work. To check if the line has no rotation angle press the line and in the properties position category. ALC color is only forwarded at the end and start of a line. To activate ALC on the lines created go to the properties of the line and under ALC settings you mark "Color from ALC".

Position	
Start point X [pixels]:	210
Start point Y [pixels]:	280
Width [pixels]:	0
Rotation angle [°]:	0

**Figure 14 - Rotation angle of 0 for lines!**

The connection point to a combined element for the ALC is in the middle of the object sides.



**Figure 15 - Connection points of a combined element**

In figure nine the connection points are colored orange of a combined element colored blue. The connection points have a maximum size of 20 by 20 pixels.

A: Height of Combined element,

B: Width of Combined element.

a: Height of connection area max 20 pixels or  $2/3$  of the A width.

b: Width of connection area max 20 pixels or  $2/3$  of the B length.

If the combined element is smaller than 30 by 30 pixels, the connection points may overlap.

Enabling magnetic points in the screen editor allows the objects to snap to these connection points for the ALC to work. To enable it, right-click in the editor and marking “use magnetic points”. Arrow keys can be used to position it more accurately also.

An internal variable has to be created for the color source. The data type of the internal variable is Boolean. This allows the color source to switch on/off. A combined element is created for the switching of the source. In the properties of the combined element under ALC, choose the function type to source and select the source color. In the “write set value” properties unmark write set value and mark the switch option. In the “variable/function” properties select the internal variable used for the element.

For the combined elements that mimic the disconnectors and breakers, a function type must be configured. Without this Zenon does not know what type of object the combined element is and because of it is unable to determine if the color should be forwarded or not. There are some default interlockings on these function types. One example is that disconnectors cannot be opened under load and if trying the interlocking becomes active and hinders the execution of the command. Unlocking of the interlocking can be done in the command processing screen.

The function configuration for the combined element is found in the properties under Automatic line coloring. Function type for breakers are switch and for disconnectors disconnector. In the configuration window for a combined element mark for every condition "get color from ALC". Check figure five.

#### **4.13 Reaction matrices**

Reaction matrices are used to get different text strings depending on the status of a variable. Reaction matrices are located under variables in the project tree. When creating a reaction matrix, it asks for a name and what type of matrix. There are five options binary, multi-binary, numeric, multi-numeric, and string. For substations, the majority of matrices used are binary and multi-binary.

When created a window opens for the configuration of the matrix. Create new conditions and select the value and write a limit value text. The limit value text configured here can

then be used in a dynamic element. For example, creating a matrix for remote/local. When the variable defining remote/local status is 0 it writes remote and when 1 it writes local to the dynamic element.

In this window, define if the condition should be displayed in the Chronological event list and alarm list. This happens when a state change occurs. The alarm/event class can also be configured here. The event classes are engineered in the project tree under variables and alarm. When these classes are configured to reaction matrices the events shown in the CEL and alarm list are colored as the event class color. And as mentioned earlier these classes can be used to create different sound effects on alarms. Every condition has its definition if it should be shown on the lists. Linking the matrices to variables is done in the properties of the variable under limit values.

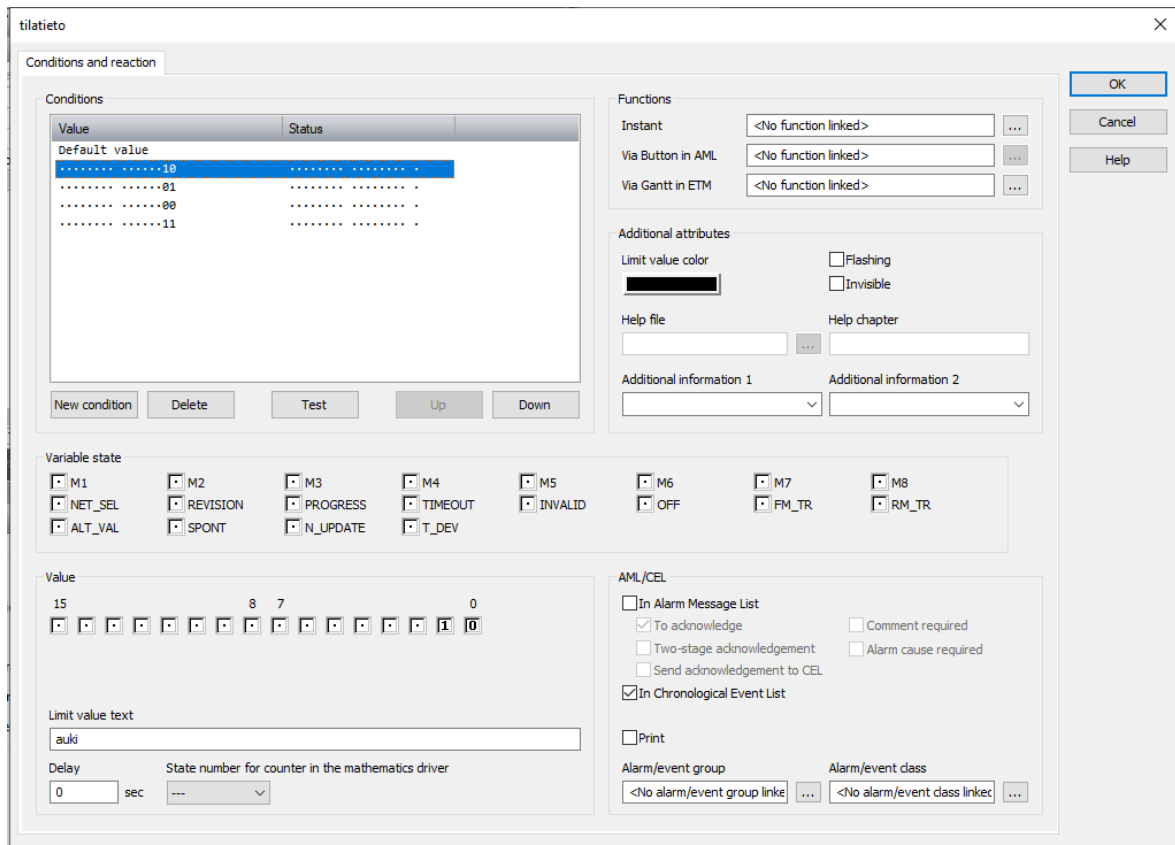


Figure 16 - Reaction matrix configuration window. This matrix is used for state indication.



## 4.14 Command processing

Command processing serves primarily for the secured switching of variables in energy technology. "Secured" means that there is a check whether the switching operation is allowed. A data point for the command processing always consist of two physical variables:

- *Response variable* - the response variable is the indication of whether the switch is open or closed. In energy technology, the logical node ends with stVal. This is an acronym for status value.
- *Command variable* – a variable that has two states 0 & 1 that writes the command to the hardware. In energy technology, the logical node ends with ctlVal (control value)

To simplify or to generalize the definition of the variable's wildcards can be used. Wildcards are only permitted as a prefix or suffix, i.e. \*xxx or xxx\*. This means that \*xxx\* is not allowed and it will not work. As a result of this flexible definition, the number of command groups that must be defined is reduced considerably.

### 4.14.1 Command processing screen

Command processing requires a screen where the actions are performed. Creating a command screen is simple with the predefined Command Processing screen type. When creating the screen with it you get all the necessary components to the screen. Meaning buttons for actions, confirmation of the action, interlockings, switching direction, and so on.

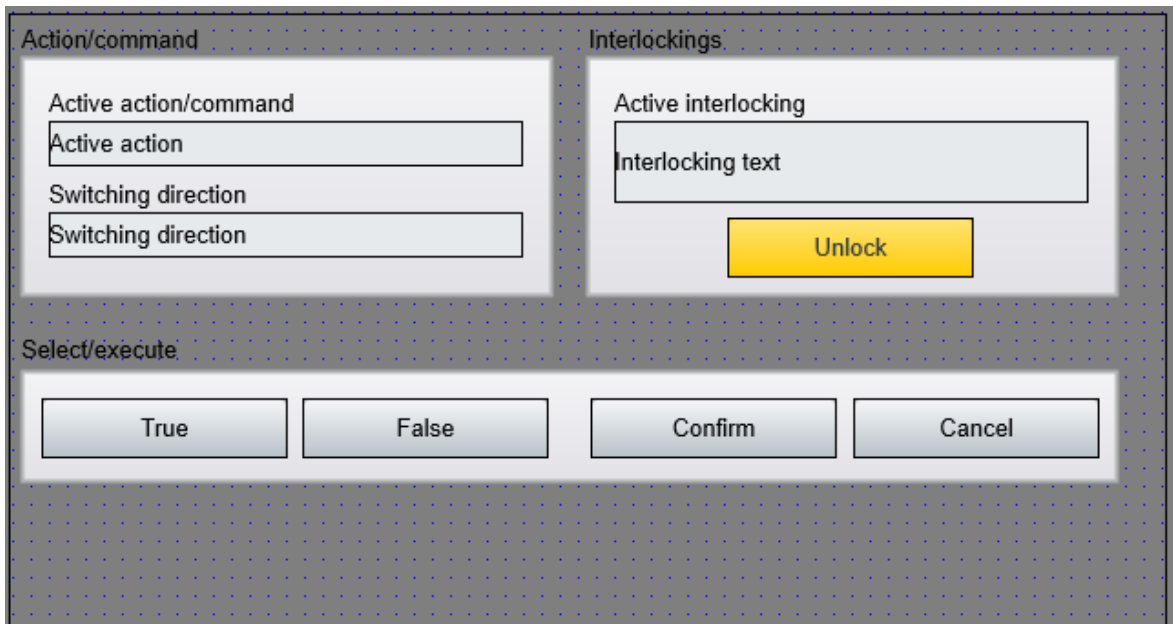


Figure 17 - Standard Command Processing screen

#### 4.14.2 Command processing configuration

From the project tree select command processing. Create and name the command group. In the properties of the command group, choose the command processing screen. Configure the response variable for the command group in the properties window. Wildcards should be used so the command group can be used multiple times. In the energy industry, it should look somewhat like this \*stVal. The start path of the variable name has to be the same until the wildcard otherwise it will not find the other command variable for the same breaker or disconnecter.

The command group tree has actions and variables in it. In the actions define your command variable and what button from the command screen is used for the command variable. By left-clicking on actions you can choose “Command new” to create an action.

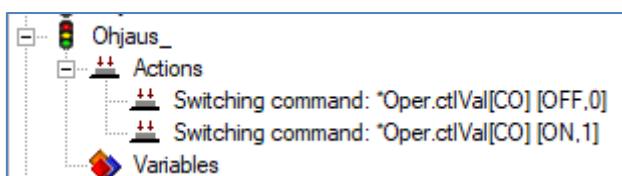


Figure 18 - Command processing tree

In the properties of the action, set the action variable which is the command variable with wildcards(\*ctlVal). In the action properties, the command processing screen isn't required

because it's already defined in the command group. Create two actions, one for switching on and one for switching off. In the action properties, the close automatically should be marked so the command screen closes when the command executes successfully.

The screenshot shows a configuration window with two main sections: 'Action settings' and 'Command Processing screen'.

**Action settings:**

- Action variable: \*Oper.ctVal[CO]
- Action type: Switching command
- Command: 0
- Set value: 0.000000
- Return state/switching direction: OFF
- Edge delay: 1000
- Modifiable states: (empty field)

**Command Processing screen:**

- Screen: <No screen>
- Replace in screen: (empty field)
- Action button: Action 1 / Button: True
- Nominal/current value comparison
- two-stage
- Close automatically
- Only executable if set <> actual
- Two-hand operation

**Figure 19 - Properties of an action. these settings are used for switching off.**

After that the command processing configuration is done, all that is left is to link the command group to variables and the combined elements that need commands. This is done in the variable properties under write set value. And for combined elements in Write set value → Write set value via: → Command processing.

## 4.15 Language files

As mentioned earlier VEO is a Nordic company that does substations. Because the substations can be in other countries than Finland, language needs to be corrected to the native language. This is where the language files become useful as they can be used to translate from let's say Finnish to Swedish. Not all properties can be changed with the language files, but most text strings, dialogs, limit value texts that are shown in the HMI can be changed. To change the language in the runtime a function has to be created that does the switching. A simple button can be engineered to the function for execution.

The @ character is an identifier for Zenon that tells that it is changeable with language file translations. So basically, wherever the text should be changeable with the language files add the @ character in front of the text string. There is always a text file called ZENONSTR.TXT this file refers to the Zenon setup language defined on the executing computer.

By clicking on the language file in the project tree, the configuration opens. In the keyword column, type the text string that is to be translated. And in the corresponding language files type the translation.

#### 4.16 Context menus

Context menus are another way to prompt the command processing. With the context menu, the switching direction can be configured. When a switching direction is selected from the menu the command processing screen opens. Because the switching direction was already chosen from the menu only confirmation is required from the command processing screen. To open the menu in runtime you right-click the object.

Open context menu from the project tree under menus. Once a new context menu is created a menu editor with the menu opens. Clicking on the empty menu opens properties. In the properties, the action type is selected to command processing. After the action type is selected Menu ID becomes configurable. Select ID\_CMD\_BDEF\_OFF for switching off. By clicking on the empty row in the menu creates another property for the menu. In this, you want to have the Menu ID as ID\_CMD\_BDEF\_ON for switching on.

Enabling this menu for the breakers and disconnectors is done in the properties of the combined element under runtime.

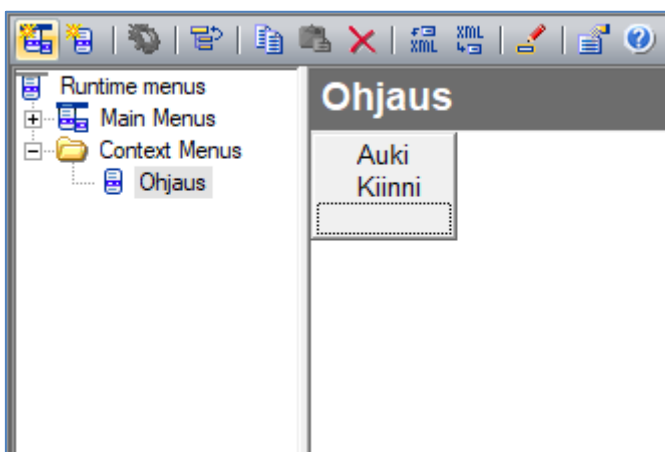


Figure 20 - Context menu editor

## 5 Discussion and result

The result of this thesis is a working HMI for a substation. Testing of the end product was done by simulating the IEDs with a software called SCL-runner. With it, you can simulate the IEDs and change statuses of logical nodes, etc. There were problems getting the XML import to work in the beginning stage of this thesis. The XML import was supposed to be used to import the variables and matrices to Zenon. The generated XML file format was bad as Zenon was not able to read it properly, and no time was available to fix it, which lead to the decision to import the variables from CID files instead.

Testing of different functions and learning about how Zenon works took time. As a beginner using this software was surprisingly easy. The User Interface is straight forward but at first, it might overwhelm you. There are a lot of windows and things happening at the same time in the editor but with time you get used to it. Zenon has well-made documentation of the software, from where it was very helpful to receive information about functions and their functionality. I got free training courses from COPA-DATA that was useful in the beginning to familiarize myself with the software. This thesis did only go through the vital functionalities for a substation but that is only a fraction of what Zenon has to offer.

There are some opportunities for improvement that can be done in the future. One is to investigate how to get the ALC color source on/off with a datapoint instead of a “dummy” button. This would lead to better visualization of the station as the color would not be forwarded if there is no current flowing in the system. In the current state of the HMI, the color is forwarded even though there is not any current in the system. This is because the “dummy” button is manually turned on when the runtime starts with a script.

Another improvement would be to create a standard symbol library with the most useful symbols that could be used to create the desired visuals. The symbol library could then be exported as an XML file and imported to different projects to save the implementer time by reusing symbols already created. And to make the symbols in a way that the connections would be identical. One instance of this is that sometimes customers want the breakers and disconnectors to have a shape of a rhombus instead of circles and squares. And if the implementer already created the HMI with squares and circles they have to be changed. Then the implementer would just swap the symbols and it would look good in the picture and no time would be spent to adjust the objects to make the visuals look okay. As the

connections points would be in the same places on all symbols so it would be easy to just swap.


I would say that this thesis was a success. The desired functionality that was tested worked well and as a result, the program is going to be tested further with a real project by VEO. When showing the configuration process to co-workers they were impressed by the software. It seems that Zenon made a good expression on them. This is a good sign that the program is useful and usable.

A lot was learned under this thesis. Most importantly the functionality of an HMI in a substation environment became clear and what it is supposed to achieve. It is hard to determine what was good and bad with the software as I don't have any experience in other similar software. But overall I would say that Zenon was good and efficient to create and configure multiple things at once. Updating the runtime to see changes takes one second which is great and allows for fast testing of the configurations made. One thing that can be bad when troubleshooting is that there are a lot of settings and it might be hard to pin down where the problem is.

## 6 References

ABB. (u.d.). Hämtat från <https://www.hitachiabb-powergrids.com/offering/product-and-system/high-voltage-switchgear-and-breakers/gas-insulated-switchgear>

COPA-DATA. (u.d.). Hämtat från <https://www.copadata.com/en/industries/energy-infrastructure/energy-solutions/zenon-energy-automated-substations/energy-infrastructure-2/>

ENSOTEST. (u.d.). Hämtat från INTRODUCTION TO THE IEC 61850 PROTOCOL | 2019  (ensotest.com)

ENSOTEST. (u.d.). Hämtat från Introduction to the IEC 60870-5-104 standard - ENSOTEST - 2019

FINGRID. (u.d.). Hämtat från <https://www.fingrid.fi/en/grid/power-transmission/power-transmission-grid-of-fingrid/>

Holma, M. (u.d.). Hämtat från <https://www.veo.fi/company/company-subpage/>

Mini S. Thomas, J. D. (2015). *Power System SCADA and Smart Grids*. CRC Press.

Padilla, E. (2015). *Substation Automation Systems - Design and implementation*. Wiley.

Punzerberger, T. (u.d.). Hämtat från <https://www.copadata.com/en/about-copadata/founding-principles/>

Vector. (u.d.). Hämtat från <https://www.veo.fi/solution/medium-voltage-switchgear/>