UNIVERSITY OF VAASA

SCHOOL OF TECHNOLOGY AND INNOVATIONS

ELECTRICAL ENGINEERING

Perttu Ollila

DEVELOPMENT OF TESTING TOOLS FOR SUBSTATION AUTOMATION AND SCADA SYSTEMS

Master's thesis

Vaasa 25.11.2019

Supervisor	Kimmo Kauhaniemi
Instructor	Marko Viitala
Evaluator	Timo Mantere

FOREWORD

I would like to thank all the people who have supported me during this thesis work and my studies.

Thanks to everybody at ABB Grid Automation for the support and expertise provided with this thesis work. Thanks to thesis instructor Marko Viitala for providing me with this thesis topic and the support during this work.

Thanks to everybody at the University of Vaasa for the education and expertise they have provided me during my studies and this thesis work. Thanks to thesis supervisor Kimmo Kauhaniemi for supporting me with this work and to thesis evaluator Timo Mantere for comments about the work.

Last but not least thanks to my friends and family who have supported me through the times of my studies and this thesis work.

Kiitos!

CONTENTS

SYMBOLS AND ABBREVIATIONS

f	Frequency
I	Current
L1	Phase L1
L2	Phase L2
N	Number of turns in transformer winding
P	Active power
$U12$	Phase-to-phase voltage on phases L1-L2
V	Voltage

ACP	Application Communication Protocol
APL	Application
DMS	Distribution Management System
FAT	Factory Acceptance Test
GDPR	General Data Protection Regulation
GOOSE	Generic Object Oriented Substation Event
HMI	Human Machine Interface
IEC	International Electrotechnical Commission
IED	Intelligent Electrical Device
ITT	Integrated Testing Tool
IO	Input/Output
NCC	Network Control Center
OPC	Open Platform Communications
OS	Object Status –attribute in SYS600

OV	Object Value –attribute in SYS600
PLC	Programmable Logic Controller
RTU	Remote Terminal Unit
SA	Substation Automation
SAT	Site Acceptance Test
SATEEN	Substation Automation Test Environment
SCADA	Supervisory Control and Data Acquisition
SCD	Substation Configuration Description
SCIL	Supervisory Control Implementation Language
SCS	Substation Control System
SS	Switch State –attribute in SYS600
TCP/IP	Transmission Control Protocol / Internet Protocol

UNIVERSITY OF VAASA
School of Technology and Innovation
Author: Perttu Ollila
Topic of the Thesis: Development of Testing Tools for Substation Automation and SCADA Systems
Supervisor: Kimmo Kauhaniemi
Instructor: Marko Viitala
Evaluator: Timo Mantere
Degree: Master of Science in Technology
Major: Electrical Engineering
Year of Entering the University: 2013
Year of Completing the Thesis: 2019 **Pages:** 92

ABSTRACT

This master's thesis describes the work in developing new testing tools for substation automation and SCADA systems. The targets of development are ABB's MicroSCADA Pro product family of network management products. Main focus is on the SYS600 Control System, which is used to monitor and control process automation applications in e.g. substations. The new testing tools will be used to test operational situations in test environments, with the tested situations being similar to the situations occurring in practical environments.

First parts of the work concentrated on collecting information related to existing testing processes and testing tools in the context where the new tools could be used. Information was collected from expert interviews, literature and prior development experiences of similar testing tools. The information was used to define requirements and features for the new tools. Initial development environment was set up based on the work and a system implementation proposal was written to describe the implementation of the new testing tools to existing products and processes.

The development of the new testing tools was based on the internal development tools of the MicroSCADA Pro SYS600 and the native programming languages SCIL and Visual SCIL. With these tools and languages the new testing tools could be developed with optimal compatibility to the products, similar to the several existing testing tools which were included in the development process. The development applied agile principles by following iterative and incremental development cycles, where demo presentations with feedback followed the development and testing stages cyclically.

The development succeeded with the result being a new operational situations testing tool with three main testing features: test case based testing, simulation run testing and communication gateway testing. Supporting features were included in the tool to set up tests with setup actions and generate test result data from the executed tests. Features from the existing testing tools were successfully combined with newly developed features, and the possibilities for future work related to the tool were considered in the end.

AVAINSANAT: Substation Automation, SCADA, Software Testing

VAASAN YLIOPISTO
Tekniikan ja innovaatiojohtamisen yksikkö
Tekijä: Perttu Ollila
Diplomityön nimi: Development of Testing Tools for Substation Automation and SCADA Systems
Valvoja: Kimmo Kauhaniemi
Ohjaaja: Marko Viitala
Tarkastaja: Timo Mantere
Tutkinto: Diplomi-insinööri
Oppiaine: Sähkötekniikka
Opintojen aloitusvuosi: 2013
Diplomityön valmistumisvuosi: 2019 **Sivumäärä:** 92

TIIVISTELMÄ

Diplomityön aiheena on uusien testaustyökalujen kehittäminen sähköasema-automaatio- ja SCADA -järjestelmille. Kehitystyön kohteena on ABB:n MicroSCADA Pro tuoteperhe, joka koostuu verkonhallinnan tuotteista. Työssä keskitytään SYS600 Control System -tuotteeseen, jota käytetään prosessiautomaatiosovellusten ohjaukseen ja valvontaan esimerkiksi sähköasemilla. Uusia testaustyökaluja tullaan käyttämään sovellusten käyttötilanteiden testaamiseen, jolloin käytännön tilanteita vastaavia testaustilanteita pyritään luomaan testausympäristöissä.

Työn ensimmäisissä osissa keskityttiin tiedonkeruuseen senhetkisistä testausprosesseista ja käytetyistä testaustyökaluista selvittäen uusien työkalujen käyttömahdollisuuksia. Tiedonkeruun lähteinä olivat asiantuntijahaastattelut, alan kirjallisuus ja aikaisempi kehitystieto samankaltaisista testaustyökaluista. Tiedon perusteella uusille työkaluille voitiin määrittää vaatimuksia ja toiminnallisuutta. Työlle valmisteltiin sopiva kehitysympäristö sekä implementaatioehdotus, joka selvittää uusien testaustyökalujen liittämistä olemassa oleviin tuotteisiin ja prosesseihin.

Kehitystyö perustui MicroSCADA Pro SYS600 -tuotteen sisäisiin kehitystyökaluihin ja tuotteen omiin SCIL- ja Visual SCIL -ohjelmointikieliin. Käyttämällä näitä työkaluja ja ohjelmointikieliä uudet testaustyökalut voitiin kehittää optimaalisella yhteensopivuudella tuotteisiin samaan tapaan kuin monet olemassa olevat työkalut, jotka olivat mukana kehitysprosessissa. Kehityksessä sovellettiin ketteriä menetelmiä käyttämällä iteratiivisia ja inkrementaalisia kehityssyklejä, joissa demoesitykset palautteineen seurasivat kehitys- ja testausvaiheita jaksollisesti.

Onnistuneen kehitystyön seurauksena saatiin aikaan uusi käyttötilanteiden testaustyökalu, joka sisältää kolme pääasiallista testaustoimintoa: yksittäiset testaustilanteet, laaja simulaatiotestaus ja kommunikaatioyhdyskäytävän testaus. Testauksen valmistelutoiminnot ja testaustulosten raportointi sisällytettiin työkaluun testausta tukevina toimintoina. Aiemmin kehitetyistä työkaluista sisällytettiin onnistuneesti ominaisuuksia uuteen työkaluun osana kehitystä, ja lopuksi voitiin arvioida työkalun tulevaisuuden kehitystä.

AVAINSANAT: Sähköasema-automaatio, SCADA, Ohjelmistotestaus

1 INTRODUCTION

Substation automation and SCADA systems are used for automated and remote control of electrical processes in substations and other parts of the power grid. Substations contain process devices used to manage the transmission and distribution in the grid, and the technology to monitor and control these devices. The components together form substation automation and SCADA systems which are controlled with system level control products. MicroSCADA Pro is a product family offered by ABB containing these system level products and is the target of the development in this thesis.

The main topic of this thesis is the development of new testing tools for the substation automation and SCADA systems. The testing tools are developed to test logical operational situations which should be comparable with how the products are used in practical environments. Testing tools are used to find unexpected features in system behavior at early development stages to ensure correct functionality in further development and customer environments.

Main objectives of the work are to collect information and requirements for the new testing tools to present the proposed implementation as a system implementation proposal, and then develop the new testing tools according to the proposal and requirements. The focus on finding information is on internal sources, such as expert interviews, related to the testing of MicroSCADA Pro systems, while theoretical sources are used to find supporting and complementary information related to software testing tool development for substation automation and SCADA systems.

The development environment is arranged based on the information found from the internal and external sources. The environment consists of the specified test environment, development tools and development practices that are applied in the process. Development is based on the use of internal development languages SCIL and Visual SCIL which are the native programming languages of MicroSCADA Pro control system SYS600. During the development process features from relevant existing testing tools

together with newly developed functionality are combined to implement a new operational situations testing tool.

The thesis contains 6 main chapters.

Chapter 2 is a theory chapter about substation automation and SCADA. The chapter explains substations and their automation and SCADA systems. Operational devices and operational situations are presented in the context of automation and SCADA.

Chapter 3 is a theory chapter about the MicroSCADA Pro. Here is the information about these substation automation and SCADA products with focus on the control system SYS600. The chapter covers also the programming language SCIL and the communication protocols relevant to this thesis.

Chapter 4 is a chapter about testing. It contains information from expert interviews and theory sources. It covers the topics of software testing theory, power system test specifications, MicroSCADA Pro project and product testing and internal testing tools.

Chapter 5 starts explaining the development. Testing system environment and components, test system configuration, new testing functionality and tool implementation are explained here.

Chapter 6 is the main development chapter. It contains the topics of development approach, development progression, tool features, tool operation and future work. This chapter describes the development work and the results of development.

Chapter 7 offers conclusions of the thesis work.

The references and attachments can be found in the end of the thesis.

2 SUBSTATION AUTOMATION AND SCADA

2.1 Substations

Substations provide centralized operation of several functions in power grids. Power is distributed and managed with the process components located in the substation, e.g. with switching devices and transformers. Voltage control, load distribution, protection schemes and network connections are among the most important tasks handled in substations. (Elovaara & Haarla 2011).

The stations can be designed for various operational purposes in the transmission and distribution grids, for example as voltage level transforming stations with switching plans or network switching state controlling stations. A voltage level transforming substation layout with digital components is shown in Figure 1. The incoming power lines are connected to the switchyard components and transformers. Operation and engineering functions are located in the service building.

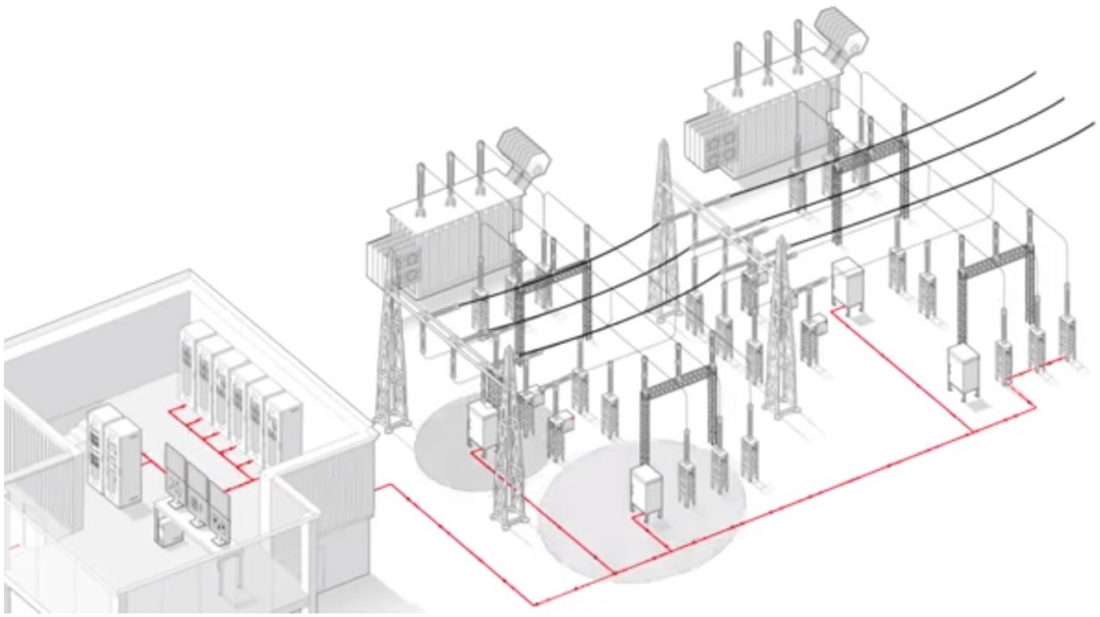

Figure 1. Transformer substation layout. (ABB 2018).

Substations can be operated locally or remotely. Depending on the situation and used devices the operators can access the system to perform control tasks on site or from remote locations. To have a clear picture of the network state at all times the substations and associated grid parts are monitored constantly by the use of monitoring components and devices. In the case of disturbances or faults the state of the substation and grid is restored to normal state by performing necessary control operations to minimize the power interruptions to consumers. (Elovaara & Haarla 2011).

Incoming high voltage power lines to substations and related switching devices are traditionally located in outdoors switchyards, as can be seen in Figure 2 substation area. This area contains powered components with high voltages and supporting structures insulated from the powered parts. Transformers and building functions of the substation can be seen in the background. Remote operation of the devices in these sorts of outdoors arrangements would be preferred for safety and efficiency.

Figure 2. Purola substation area in Vaasa.

The substation is usually a part of the regional electrical power grid and the design takes into account the operation of the substation as a supporting station in this grid. Operational situations that support grid stability and reliability can be coordinated between several substations in e.g. peak load and fault situations (Hiltunen 2016.)

Depending on the operational purpose the substation can include busbar systems, switching devices, transformers, measurement components, load balancing components and protection and control components. The components which are used in the primary power process can be pictured in substation circuit diagrams, which show the used devices and connections. Figure 3 shows a substation switching state single-line diagram with two busbars, a bus-coupler and two outgoing bays with bypass disconnectors. (Bayliss & Hardy 2011).

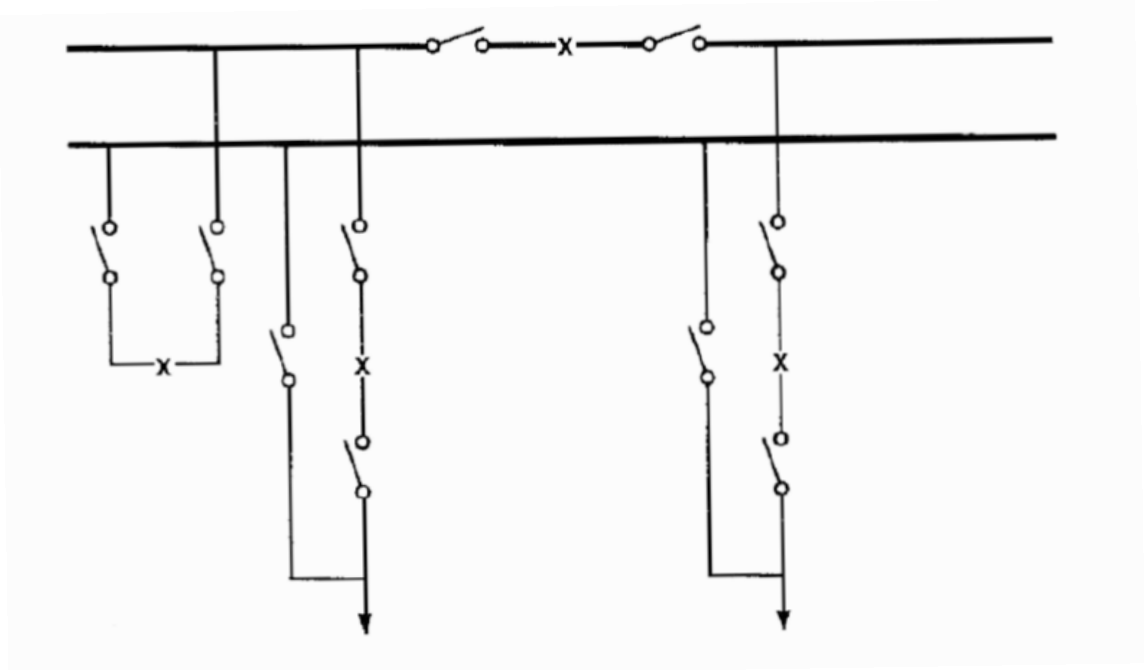

Figure 3. Double busbar substation single-line diagram. (Bayliss & Hardy 2011).

The arrangement of substation equipment is usually defined by the operational characteristics and economic costs of the system. Including several busbars, bypasses and devices in the circuit can provide reliability and operational flexibility, but the initial sys-

tem cost could be increased along with system complexity in operation and protection design. (Bayliss & Hardy 2011).

2.2 Automation and SCADA systems

The main electrical processes in substations contain high voltages and high currents, which makes it necessary to isolate the control and communication processes to use their own devices for practical operation of the substation. Operation of the devices in these processes is possible by various manual and automatic methods.

Manual control can be e.g. operating a disconnector from the control panel located close to the physical device or from a remote operations workstation with a control command. Automatic control in this case would be to operate the disconnector by an automatic operation triggered by logic programmed to the device controller or by an automatic control response signal to a received indication signal in the control system at a network control center. Both manual and automatic operation can be designed to be executed locally or remotely, if the system design allows this.

Benefits of adding automation to substations can include (Lehtosaari 2011):

- Improved safety and protection design
- Improved reliability
- Improved efficiency
- Reduced costs

Automation can even be an alternative to or postpone the addition new facilities to substations. By techniques like demand response and optimal network reconfiguration the capability of the grid can be extended with substation automation. On the other hand the automation systems can require more expertise in the design, engineering and operation

of the system as microprocessor based devices are added to the system and more data is being processed. The levels of automation in existing installed substation systems are varying and as the utilities are looking to modernize their substations, they are more likely to choose systems which have well-proven performance and which are being tested to a high extent. (Madrigal & Uluski 2014).

Substation automation systems can be seen as consisting of hierarchical levels. If the main process components are included in the levels, these levels are the following: plant level, process level, bay level and station level. This is demonstrated by the levels in Figure 4 with associated testing tools presented. (ABB 2017).

Plant level contains the process devices and their connections. From here the measurement values and device states are obtained and forwarded to next levels. Raw data and values are used to make operation decisions at upper levels.

Process level covers the components which handle the two-way signal communication between the plant and bay levels. Many signals can be combined to merging units which route the signals forward for monitoring and operations.

Bay level features the protection and control units for the bay. These units, e.g. PLCs (Programmable Logic Controller) and IEDs (Intelligent Electrical Device), can contain the protection and control logics used to operate the plant devices during normal operation and fault situations.

Station level is the level where all information from the substation components is combined for supervisory and control tasks. In this level workstations, e.g. HMIs (Human-Machine Interface) with process pictures and engineering machines are used to operate the substation. Connecting the substation to upper level systems is possible with communication links and devices, for example RTU’s (Remote Terminal Unit).

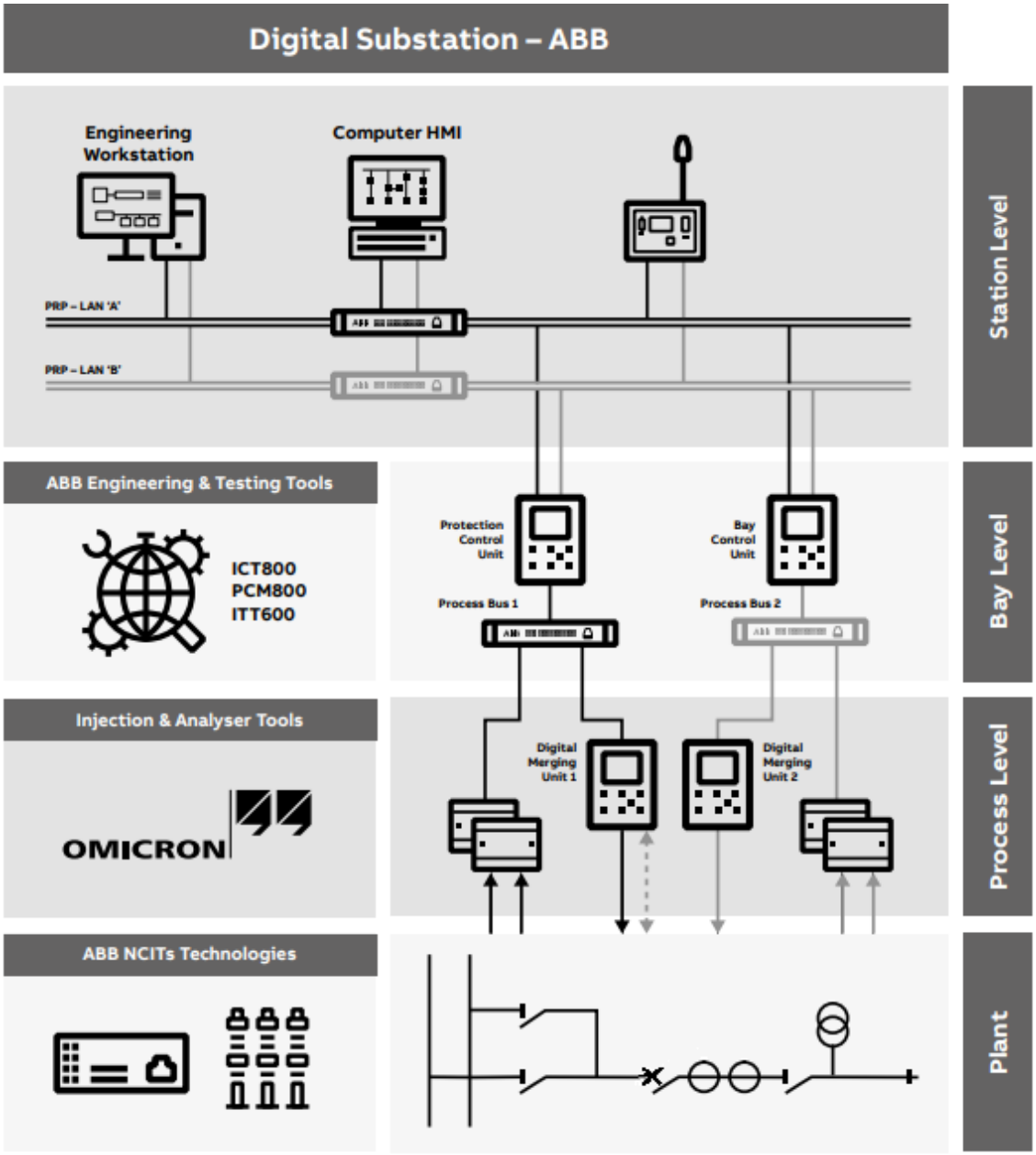

Figure 4. Substation system levels based on ABB (2017).

The testing and measuring technologies in Figure 4 can be used in engineering and commissioning substation automation systems. As the number of devices in the systems can become large, it is practical to have technologies which can test sections or levels of the system to e.g. simulate values and signals. The amount of programming and configuring in the system can be substantial and to ensure all components work well together extensive system wide tests can be designed.

SCADA (Supervisory Control and Data Acquisition) systems are used in substation applications to monitor and control the process. SCADA systems provide data for operators of the substation about the real-time state of the process and about the device states and measured values. Basic information presented in a power process SCADA HMI is positions of switching devices, powered sections of the circuits and presentation of measurement values. The main function of SCADA is the acquisition and presentation of this information for operations with process pictures and event and alarm lists, while complex analysis tasks are usually handled by other systems, such as a DMS (Distribution Management System). (Martikainen 2017).

Figure 5 presents typical SCADA components at substation level. Station computers are set up as redundant unit pairs to prepare for interruptions in operation, in that case the stand-by machine takes over the tasks of the interrupted machine. HMI machines contain the process diagrams, event and alarms lists and other operational information about the system. GPS clocks are used for time-synchronization. Printers are used to print operational information. Gateway machines connect to remote systems.

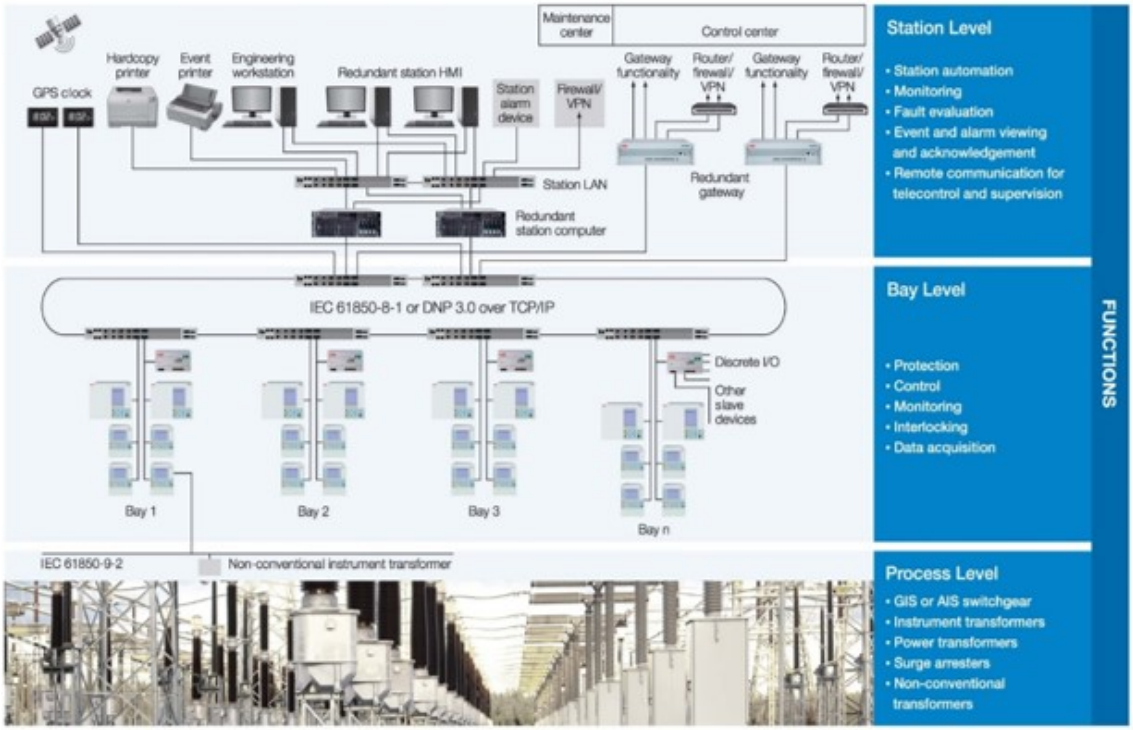

Figure 5. SCADA components at station level. (ABB 2015).

Essentially the SCADA system acquires data from the substation processes and presents the signal values in a real-time database (Bayliss & Hardy 2011.) This database can contain both real and computed signal values depending on the data types and functionality. These signals are transmitted between the process and the SCADA system by various devices, e.g. IEDs, PLCs and RTUs.

According to Bayliss & Hardy (2011) the main modes of input data communication to the SCADA system are:

- Scheduled polling of units on a regular basis for update of all input data
- Change of state capture where only changed input data is transferred

The representation of physical devices usually consists of several grouped signals which contain specific attributes of the device. In the case of e.g. a circuit breaker this can mean that the device is represented in the database by all input/output signals related to the device. These signals are then used to monitor and control the device. Same combining analogy can be applied to alarms in the context of computed signal values, so that alarms can be managed according to severity classes and related alarms. (Bayliss & Hardy 2011).

2.3 Operational devices

The main mechanical switching devices in substations are circuit breakers, disconnectors and earthing switches. Circuit breakers are capable of opening and closing during load and fault currents under specified conditions in the circuit. Disconnectors are designed to provide safe isolations in the circuit and can be operated when negligible current or voltage is present in their terminals. Earthing switches can connect parts of circuits to earth for e.g. maintenance operations and are capable of carrying specified currents during fault situations (Bayliss & Hardy 2011.) Instances of the switching devices and their drawing symbols are presented in Figure 6.

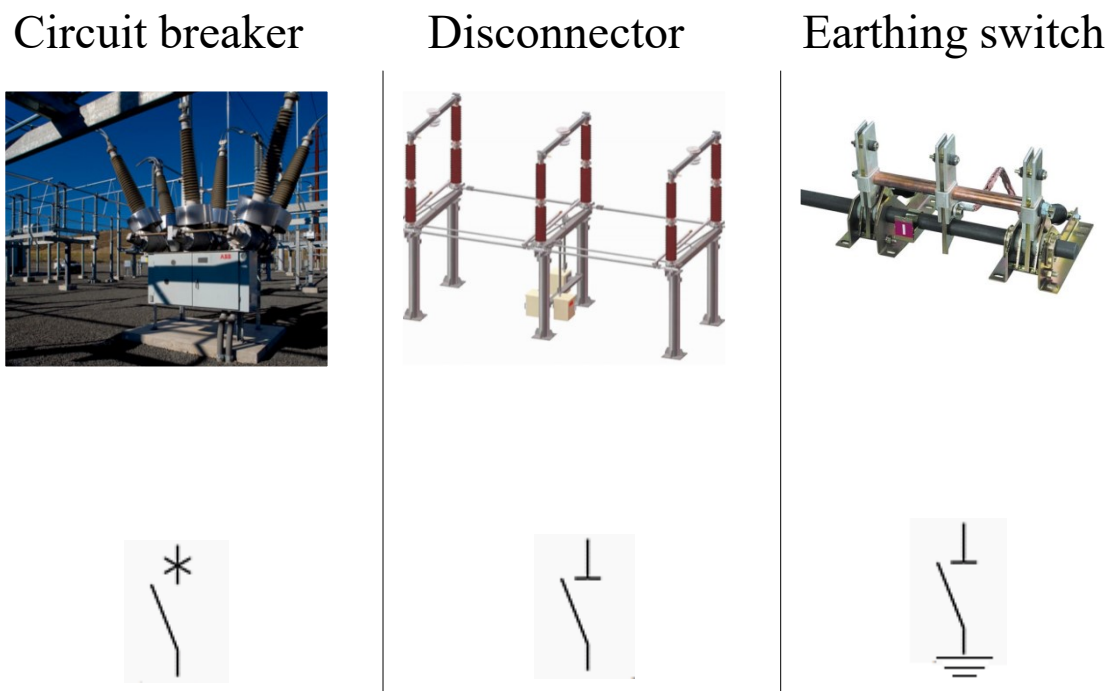

Figure 6. Switching devices. (ABB 2019a).

Switching devices in the substation change network topology state to control power flow in the grid, isolate faulty parts of the network to protect other parts and separate network sections from one another (Elovaara & Haarla 2011.) The devices can be operated automatically by e.g. protection relay tripping signals, or manually by operators. Regularly the switching plans include logical and sequential control schemes to operate many devices in series with interlocking conditions and safeguard functions.

SCADA presentations of the switching devices include various indication and command signals. The indication signals are received from the process and include information such as switch position, interlocking statuses and blocking statuses. The command signals can include open/close commands, cancel commands and possible selection commands depending on used protocols. In addition to the mentioned process signal types the device presentation in the SCADA database can contain internal signals configured for specific applications. Typical SCADA process objects for switching devices in ABB's MicroSCADA Pro SYS600 control system product are presented in Table 1. (ABB 2016b).

Table 1. Switching device process objects examples in SYS600. (ABB 2016b).

Index	Explanation	Purpose
10	Position indication	Used for position indication of the switching device state open/closed/intermediate/faulty
11	Open select or open execute	Depending on the defined control type it is used for sending Open select or Open execute to the control unit
12	Close select or close execute	Depending on the defined control type it is used for sending Close select or Close execute to the control unit
14	Cancel command or execute close command	Used for sending Cancel command or Execute close command to the control unit
15	External control blocking	Receives a control blocking signal from the control unit and prevents the control actions in the single line diagram
16	Open interlocked	Receives a control blocking signal for Open command from the control unit and prevents the Open command in the single line diagram
17	Close interlocked	Receives a control blocking signal for Close command from the control unit and prevents the Close command in the single line diagram

Transformer devices in substations are the power transformers and the instrument transformers. Power transformers are used to step-down or step-up voltage levels to connect e.g. high voltage transmission lines to distribution lines and power generation plants to higher voltage transmission lines. Instrument transformers are used to obtain measurement values from the network and scale the values to a suitable level for the systems

that use the measurement data (Elovaara & Haarla 2011.) Figure 7 represents instances of the power transformer, current instrument transformer and voltage instrument transformer with their corresponding drawing symbols.

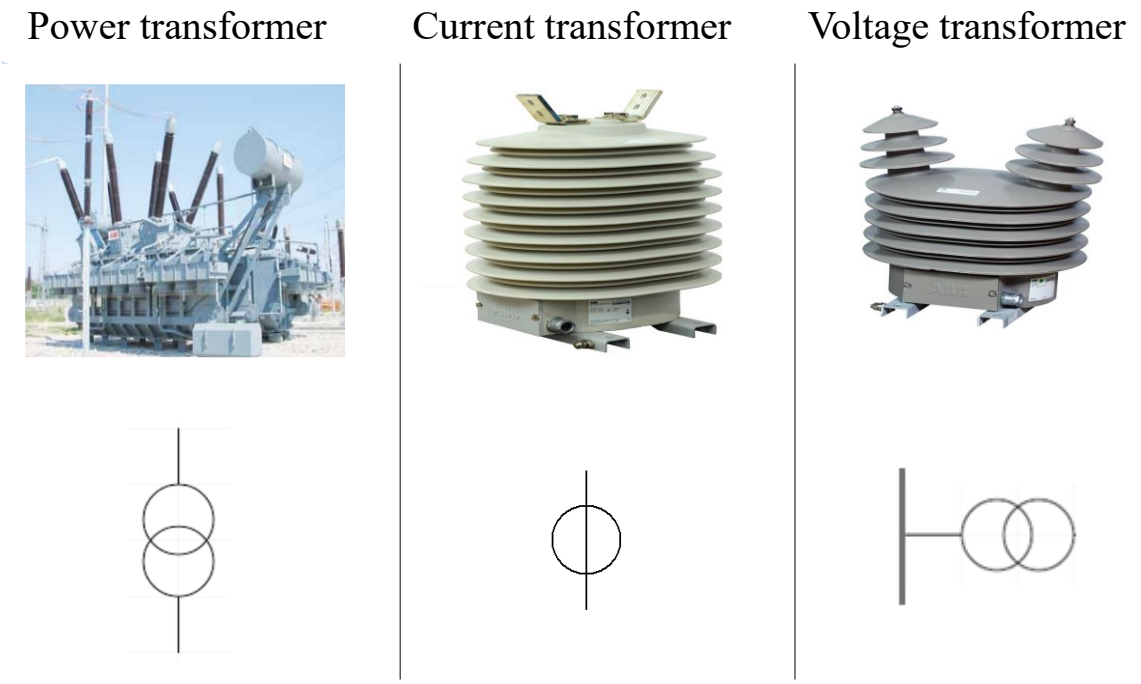

Figure 7. Transformer devices (ABB 2019b.)

Power transformer secondary voltage can be controlled with tap changers that select the number of turns in the secondary windings. Transformer voltage ratio is determined by the ratio of the turns in primary and secondary windings as presented in formula 1

$$\frac{V_1}{V_2} \sim \frac{N_1}{N_2}, \tag{1}$$

where V_1 and V_2 are the primary and secondary voltages and N_1 and N_2 the numbers of turns in primary and secondary windings, respectively. Therefore adding more turns to the secondary winding increases the output voltage. (Bayliss & Hardy 2011).

The SCADA presentation of transformer process control as signals can be implemented similar to the switching devices. Indication signals for power transformer include information such as tap changer position and voltage, and for instrument transformers measured values like phase currents and voltages. Command signals for the power transformer can include tap changer position commands and setting reference voltage values. Typical power transformer SCADA objects in MicroSCADA Pro SYS600 are presented in Table 2 and measurement device objects are presented in Table 3.

Table 2. Tap changer process object examples in SYS600. (ABB 2016b).

Index	Explanation	Purposes
10	Tap position indication	Used for indication of the tap changer position
24	Voltage	Used for indication of voltage
26	Reference voltage command	Used for sending the reference voltage to the control unit

Table 3. Measurement process object examples in SYS600. (ABB 2016b).

Index	Explanation	Purposes
10	Current L1	Current measurement on phase 1
16	Voltage *U*12	Phase-to-Phase voltage measurement on phases L1-L2
20	Active power *P*	Active power measurement
24	Frequency *f*	Frequency measurement

Typical automation and SCADA system devices at the substation and bay levels are RTU's, PLC's and IED's. The RTU serves as the communication link between the substation devices and upper level systems by providing functionality for I/O (Input/Output) connections, protocol communication and connections to SCADA systems (Elovaara & Haarla 2011.) The PLC is a controller device used at substations to control one or several process devices and enable interoperability of devices between e.g. different bays by providing logical signal processing and I/O functions (Bayliss & Hardy 2011.) The IED is a more general term for intelligent devices, but for substation applications a good example would be a microprocessor-based protection relay with protection and control logics.

Figure 8 shows a substation automation setup with connections. In this setup the RTU is the centerpiece which provides the bay level IED and PLC devices with connections to upper level systems and workstations.

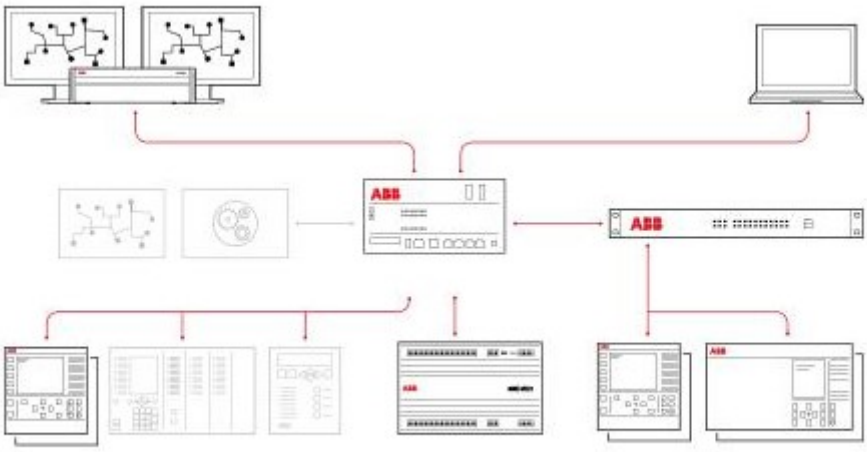

Figure 8. RTU-based substation automation setup. (ABB 2019c.)

In the SCADA system database all signals belonging to a specific device can be presented as e.g. entire bays with process objects defined as the bay devices. This way the bays can include objects specific to the automation device in addition to objects of the process devices. This enables including objects like trip circuit signals and auto-

reclosing controls to the SCADA system database under the bay controlled by the automation device. Typically the SCADA system includes data from several IEDs at different bays and different substations, and this enables the engineering of interoperability schemes and advanced control functionality for wide-ranging applications.

2.4 Operational situations

According to Warne (2005) the main operational responsibilities for transmission grid operation is to maintain constant frequencies and voltages in the system for the consumers and operate the system in a secure and efficient manner. To achieve this the system operators control and monitor the transmission components, mainly located in substations, and communicate with other system parts to co-ordinate balancing and compensating actions. The systems regularly have connected central and regional control centers with supervisory control facilities.

Warne (2005) also addresses the operational tasks at distribution grid level, where the handling of system alarms and co-ordination of repair and maintenance actions are seen as most important tasks. The distribution grid has far more substations, components and power lines to monitor and control than the transmission grid, leading to more repairs and faults needing operator actions. The large number of components also means that planning of maintenance and installation work is very important as the grid must stay functional while some of the components are out of use.

Substation operation can be roughly divided to two main situations: normal operating conditions and abnormal conditions. Under normal operating conditions the process is stable and most suitable switching plans are being used to enable effective power transmission and distribution. Minor changes in voltages and loads can be handled with transformer tap changers and compensation devices like capacitor banks. When the condition changes to abnormal, e.g. as the result of component fault or short circuit, the state of the system is kept stable with actions to operate switching devices and separating the faulty sections of the network from healthy parts. In worst cases the power sup-

ply to customers is temporarily lost until the faults have been repaired by field repair crews or alternative supply routes have been set up in the grid.

A simplified example from Bayliss & Hardy (2011) shows how the operating conditions are applied in practice to a single busbar system with two incoming power lines. The system contains three circuit breakers, A and B at the incoming lines and C at the busbar between the lines. Instrument transformers measuring voltages and currents are located on the incoming lines. The circuit is presented in Figure 9. In this case the devices are controlled by a PLC, enabling automatic control at local or remote states.

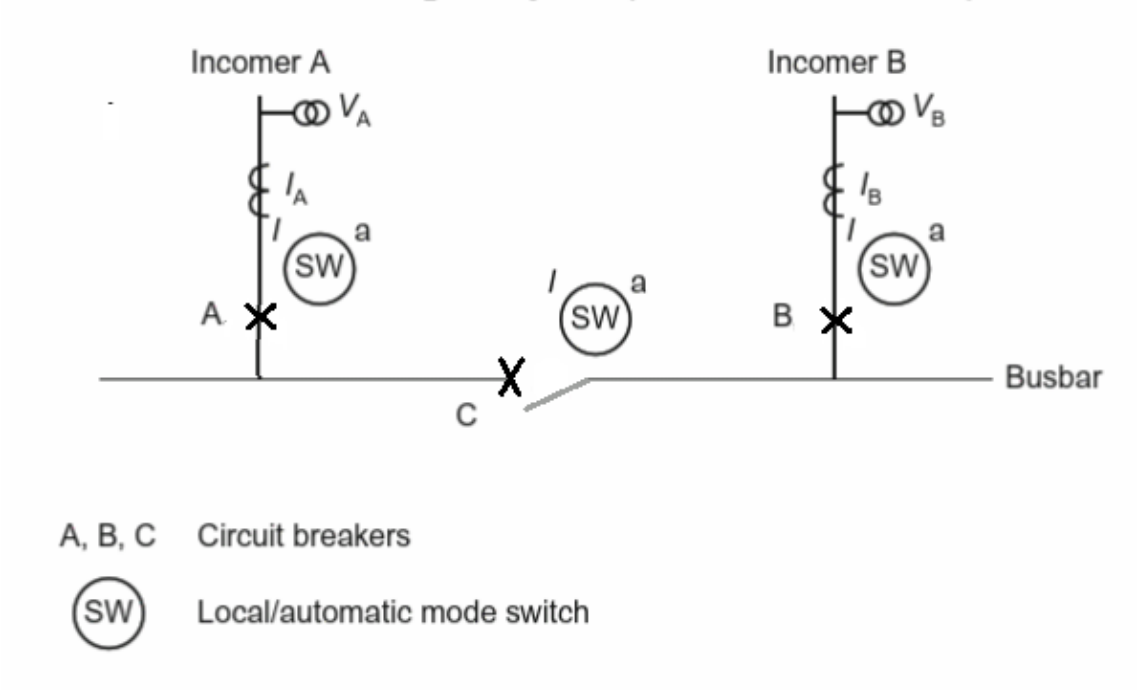

Figure 9. Circuit of operational situation example (Bayliss & Hardy 2011.)

Normal operating conditions apply under the following circumstances:

- Circuit breaker changeover control is under automatic mode
- The left hand side of the busbar is fed from incomer A with circuit breaker A closed and the bus section circuit breaker C open

- The right hand side of the busbar is fed from incomer B with circuit breaker B closed and the bus section circuit breaker C open

An abnormal condition would apply in the following situations:

- Power input failure or circuit breaker A faulty on incomer A. The whole busbar shall be fed from incomer B with the bus section circuit breaker C closed

- Power input failure or circuit breaker B faulty on incomer B. The whole busbar shall be fed from incomer A with the bus section circuit breaker C closed

The situation where the whole busbar is fed from incomer B is depicted in Figure 10.

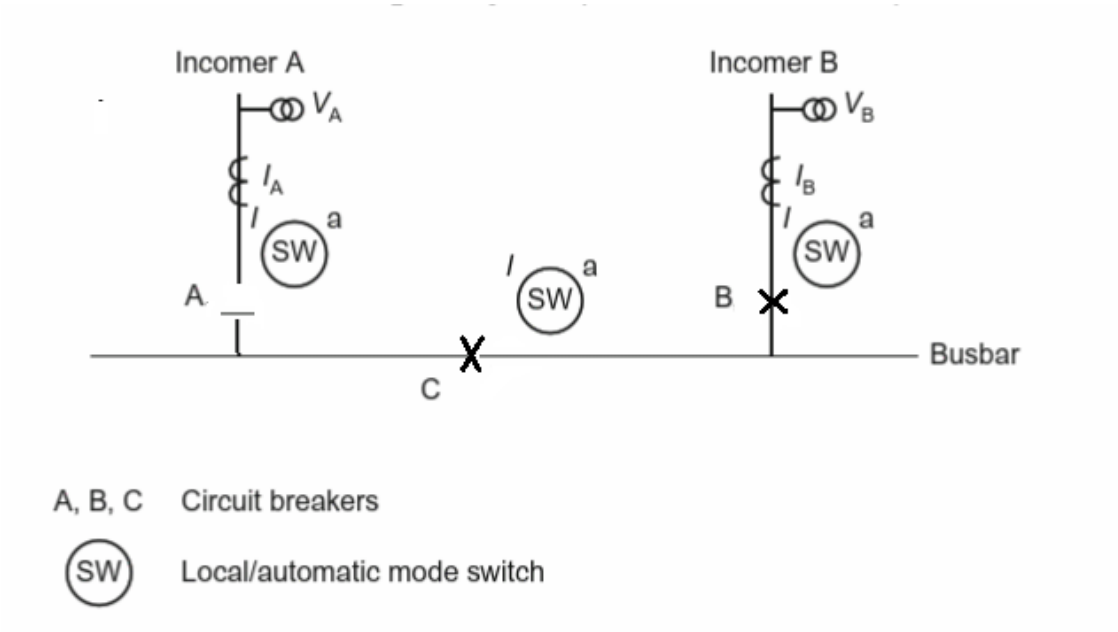

Figure 10. Circuit of abnormal condition example (Bayliss & Hardy 2011.)

In addition to specifying the operating conditions, the system requires specification of operational constraints, communication requirements and remote control definitions. Specifications can imply that e.g. automatic control is disabled as a safety constraint when a circuit breaker is on local mode or operator confirmation is required for actions after restoring the system from fault situations. (Bayliss & Hardy 2011).

When a SCADA system is used to operate the system in the example, a common setup would be to connect the PLC to the SCADA system either at the substation level or to a remote control center by an RTU. The system operator can then monitor the state of the process devices by values acquired from the PLC and send remote control commands to the devices to be carried out by the PLC. In a normal condition the operator would receive measurement values from the instrument transformers and circuit breaker status information.

In the abnormal condition where circuit breaker A becomes faulty, the information from the circuit breaker would be signaled to the SCADA system and an alarm would be generated to inform the operator about the situation. Depending on the configuration the operator could then manually execute the switching actions to close circuit breaker C and verify that the operation was successful, or there could be automatic logic to execute the operations when the faulty status of circuit breaker A is detected.

3 ABB MICROSCADA PRO

MicroSCADA Pro is a family of products for network management by ABB. The products include the SYS600 Control System, the DMS600 Distribution Management System and the SYS600C Compact System.

3.1 MicroSCADA Pro SYS600

MicroSCADA Pro SYS600 is a control system for monitoring and controlling process automation applications. While the system can be applied to several domains, e.g. electric power systems and district heating, it is primarily designed for substation automation and network control functionality. Supported main functionality includes local/remote process control and monitoring, gateway communication, reporting, system redundancy and mirroring. The automation systems can be built and customized to perform tasks according to application requirements, scaling from single system servers to large hierarchical systems. This is visualized in Figure 11 below. (ABB 2016a).

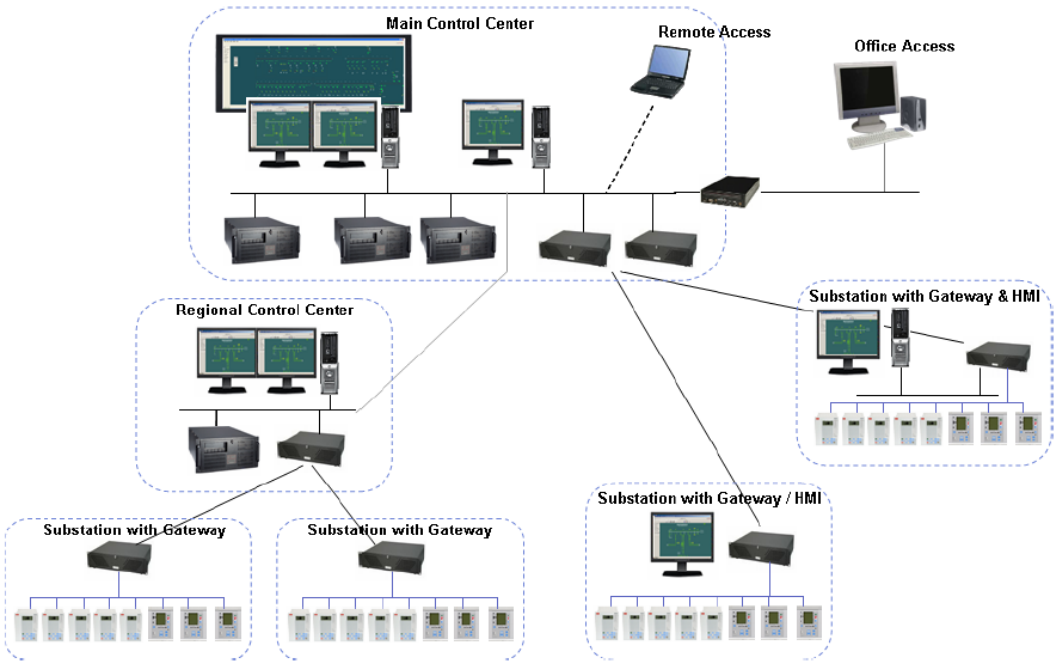

Figure 11. SYS600 hierarchical system example (ABB 2016a.)

The SYS600 system in Figure 11 includes several levels of control and SYS600 functionality. In the substation level, the SYS600 can function as a gateway for transmitting information to higher level control centers and also include an HMI for local use if needed. Control centers can access data from several lower level systems and form large control systems with hundreds of thousands of data points. This means in practice that in substation automation, for example, several substations can be controlled from one centralized location with SYS600. The system supports connectivity to other vendors' products along with several commonly used communication protocols. (ABB 2016a).

The SYS600 is characterized by scalability. Systems can be built for wide ranges of I/O data points and tasks while retaining performance needs required by applications. The main system performance considerations include (ABB 2016a.):

- Process communication load
- Number of simultaneous workplaces
- Intensity of archiving, calculations and reporting
- Possible other system specific functions

These characteristics show how the performance of the system is affected by both constant and varying factors. The number of simultaneous workplaces and similar system specific functions can be thought of as relatively fixed functionality while the process communication load and reporting functionality can cause more varying load for the system. The answers for performance needs are to use machines with suitable processing capacities and a various number of machines in the system (ABB 2016a.)

3.1.1 Power process applications

The application controlled with MicroSCADA Pro Control System SYS600 is the specific set of database objects, HMI displays, communication functionality and system

functionality configured for operation for the system users. In the case of power applications, more specifically substation applications, this means the database of substation objects (switching devices, transformers, measurements etc.), displays (single-line diagrams, event and alarm displays etc.), communication signals (monitoring and control) and system design (servers, redundancy, mirroring etc.). (ABB 2016b).

The SYS600 uses graphical process displays designed to contain the relevant information for the user of the system and serve as the main HMI between the operator and the process. Types of displays include process displays (shown in Figure 12), event displays, alarm displays and historical data displays (trends and measurement reports). (ABB 2016b).

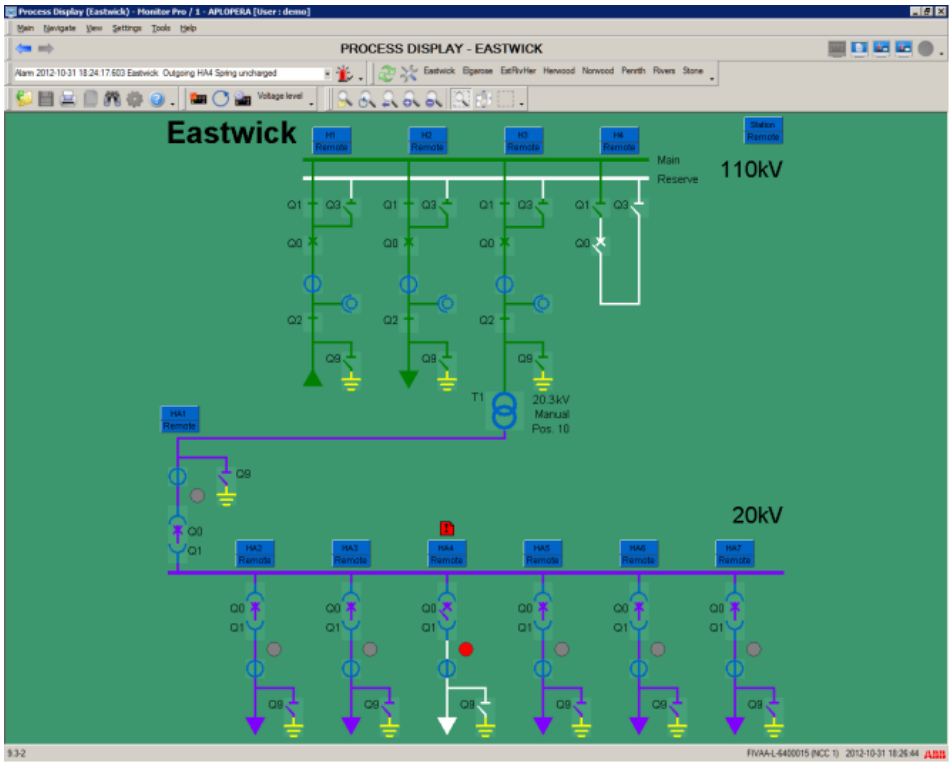

Figure 12. Process display of a demo application (ABB 2016c).

Symbols on the display represent the substation process devices and visualize the process status. The powered and unpowered parts of the station, device states, alarms and relevant object identifier information are shown. Several process displays can be combined to form overview displays of multiple substations, voltage levels and bays. What

happens on the HMI screen should also take place at the actual substation, giving the user opportunity to monitor system status and take action when needed. (ABB 2016b).

Event and alarm displays are lists of the events and alarms occurring in the system. They allow the user to get information about the system in the form of updating lists which are to be displayed in chronological or otherwise user defined order, with possible filters applied. In application testing, these lists present an opportunity to monitor the completion of system events and acknowledging alarms according to the application configuration. The event list is presented in Figure 13. (ABB 2016b).

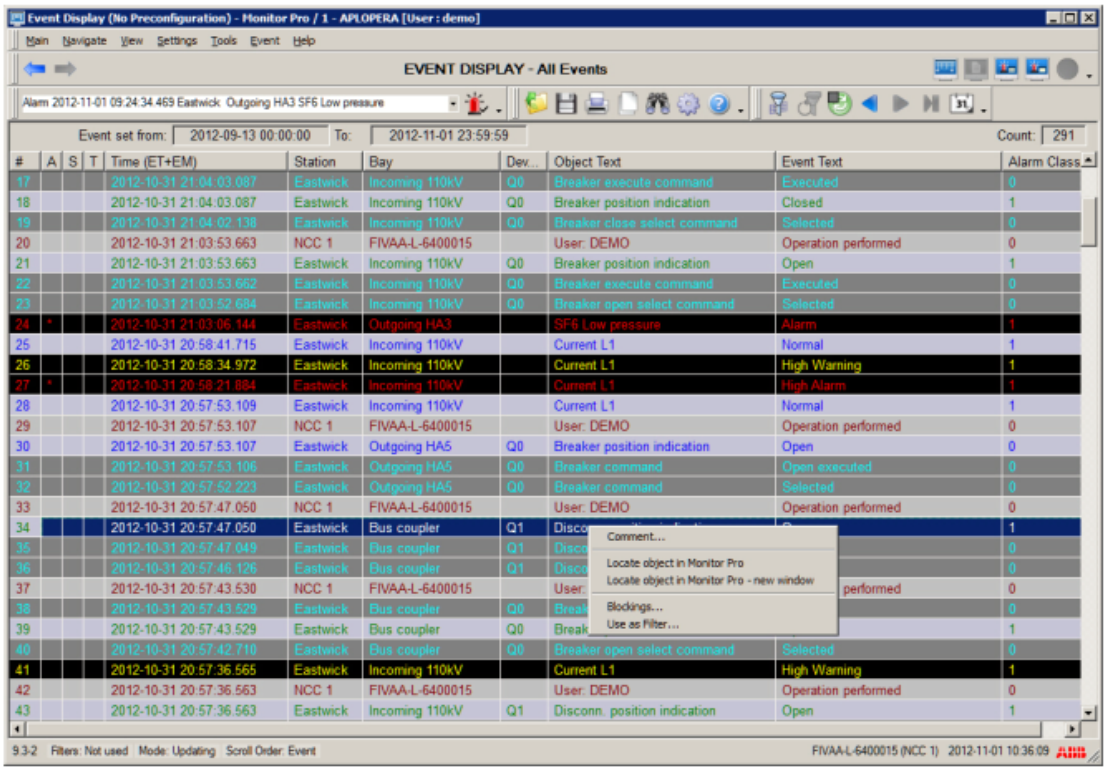

Figure 13. Event list view in SYS600 (ABB 2016c.)

The mapping of operational devices into software application objects is handled by the SYS600 process database. In this database the devices are presented as objects, as presented earlier in context of the operational devices, which have their set of defined and configurable attributes. The SYS600 will control the process with these application ob-

jects: output objects are used to send commands to the process devices and the input objects receive monitoring data from the process. (ABB 2016d).

Information flow in the system from the process to the operator view is presented in Figure 14. The process database is the link between the operator displays and the process devices. (ABB 2016d).

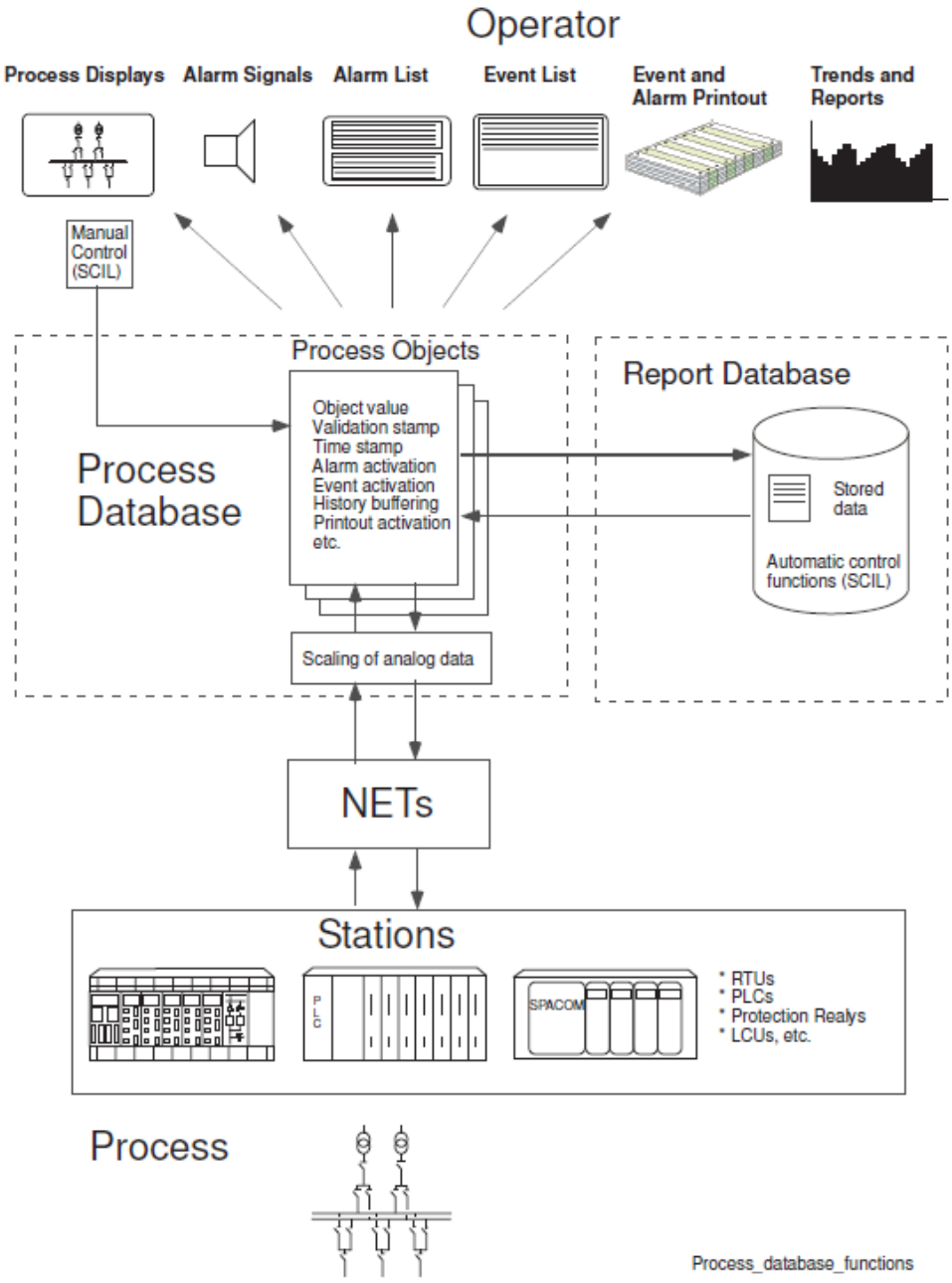

Figure 14. The functions of the process database (ABB 2016d.)

In Figure 14 the data sent by station level devices is mapped by the NET communication units to the process database where process object attribute values can be updated. These process objects and their attributes also link the process data to the report database for reporting and archiving purposes. (ABB 2016d).

The process device objects contain static and dynamic attributes, which describe object features and dynamic process functionality. Static attributes describe the object addresses and identities, activations, operational status, alarm and event handling, and related programs and expressions. The dynamic attributes contain information about object values, status and time data, and other real time process state indications. An example data object with sample attributes is presented in Figure 15. (ABB 2016d).

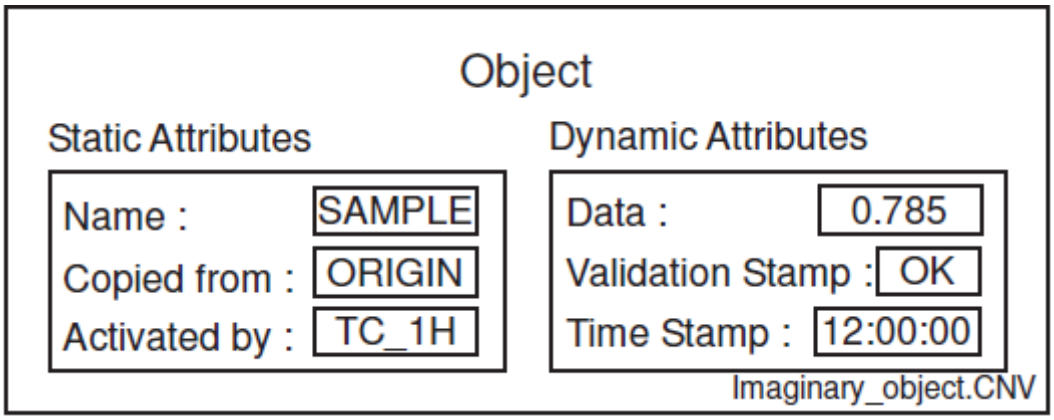

Figure 15. Sample data object with attributes (ABB 2016d.)

In operation the dynamic attributes of the object are updated when the process state is changing by input or output signals. The Object Value (OV) and Object Status (OS) indicate the value and reliability of the object state. For example a measurement object could receive an analog value update from the process station, which updates the OV attribute value. If the station has marked the object status as OK, the OS attribute would be set with the value OK. When the value is unreliable, e.g. faulty registration time, the OS would be set with a value indicating the unreliable state. (ABB 2016d.)

The objects can be configured for automatic and manual updating of the object attributes, which will determine whether the object attribute values will be connected to the

process or if they are operated within the system databases and memories only. For example the objects' Switch State –attribute (SS) describes the updating method of the object value attribute in four different states: off, manual, auto and fictitious. (ABB 2016d).

3.1.2 Process control and monitoring

The SYS600 initializes the process control by application specific object updates. When an application is prepared for operation and started up, the process database is updated with data from station level units. RTUs, PLCs, relays and other station IEDs send process data to the system according to station type. After the process database has been updated with process values, the process status in the system should match the status of the physical process. (ABB 2016d).

Control commands can be configured for manual and automatic operation of devices. The data transferred from the process to the system may initiate these control features for automatic operation or it can generate indications which inform the user and suggest control actions. The user can then take action, for example operate a switching device. Control dialog for disconnector operation is shown in Figure 16. (ABB 2016c).

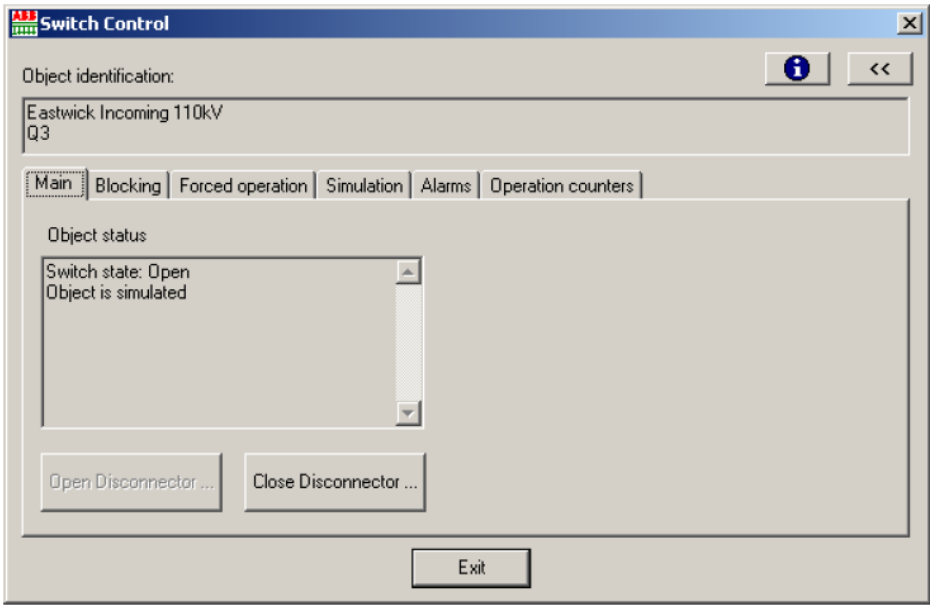

Figure 16. Disconnector control dialog (ABB 2016c.)

The control dialog in Figure 16 shows the status and control features, of the switching device. The “Object status” window contains the status information of the device, for example position indication and interlocking conditions. Available control actions depend on the device configuration and operator authority status. (ABB 2016c).

Similar control functionality is included for other substation process devices, such as transformer tap changers and measurement equipment. A current measurement control dialog is presented in Figure 17. The user can track the measured values received from station level devices and see the comparison to warning and alarm limits. (ABB 2016c).

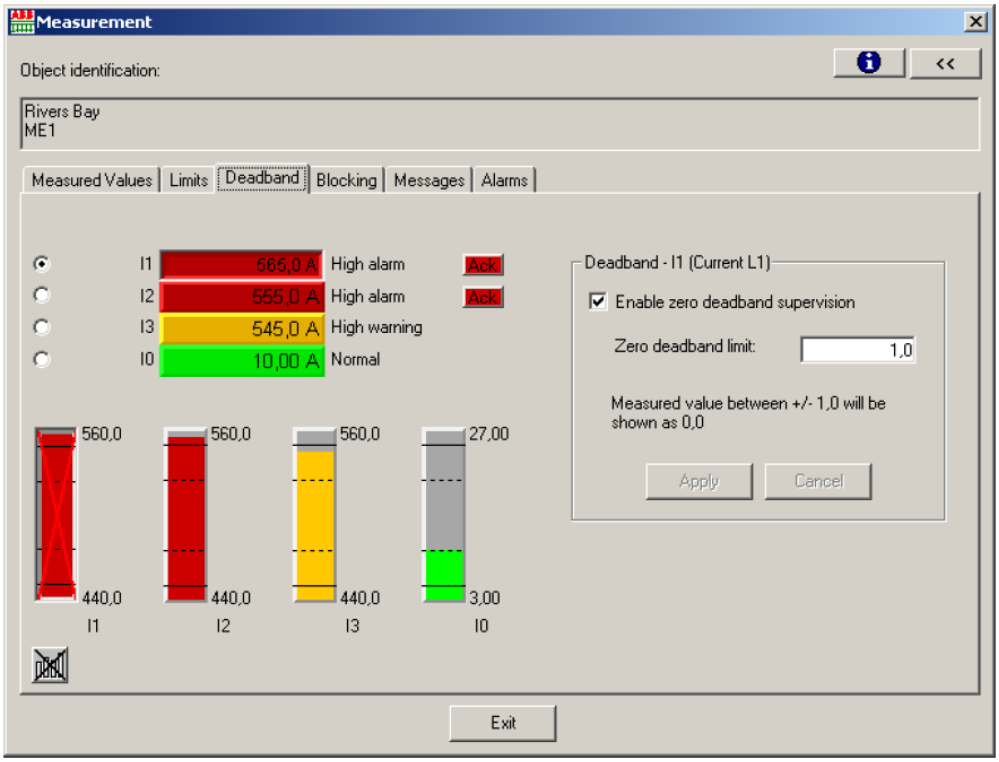

Figure 17. Measurement dialog (ABB 2016c.)

3.2 COM500i

The COM500i is a SYS600 SCADA communication server used for communication between station level processes and higher level systems, i.e. NCCs (Network Control Centers) in electrical power applications. It sends and receives signals from the process

and system users to connect systems with communication protocols, and manages tasks related to the communication environment. Application use can range from a stand-alone gateway to combined functionality with a substation control system (SCS). (ABB 2016f).

A COM500i application communicates through the SYS600 base system and runs in the SYS600 environment. The environment is presented in Figure 18, where the system components are connected by communication units. (ABB 2016f).

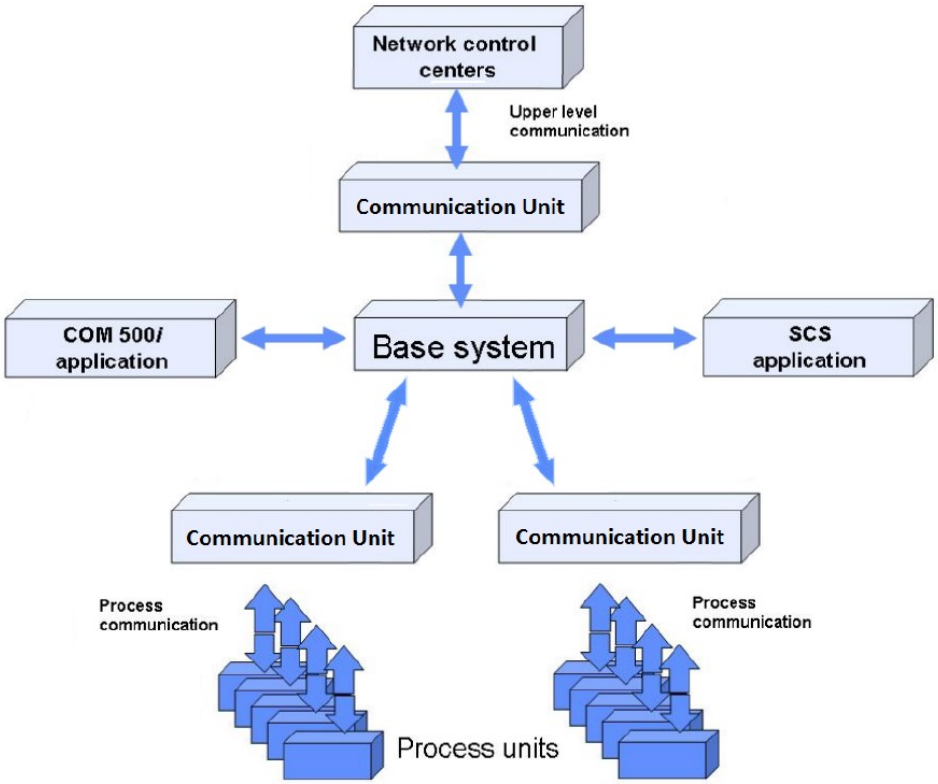

Figure 18. Communication environment of COM500i (ABB 2016f.)

The communication information from COM500i is transmitted to other system components by SCIL command procedure objects. In addition to control commands, the COM500i can also transfer general system and application data (interrogation commands, disturbance situations, start-ups). (ABB 2016f).

Data flow in the system for control command direction is detailed in Figure 19. Process objects of input types receive the commands and setpoints from the NCC through a communication unit (NET unit). This activates the command procedures defined for the process objects and the commands are forwarded to the process objects of output types. These outputs are then transmitted to the process devices. Object attributes that are relevant during these operations include the logical names and indexes (LN, IX), object values and time stamps (OV, RT) and signal handling attributes. The XREF data refers to the cross-reference information of the objects. (ABB 2016f).

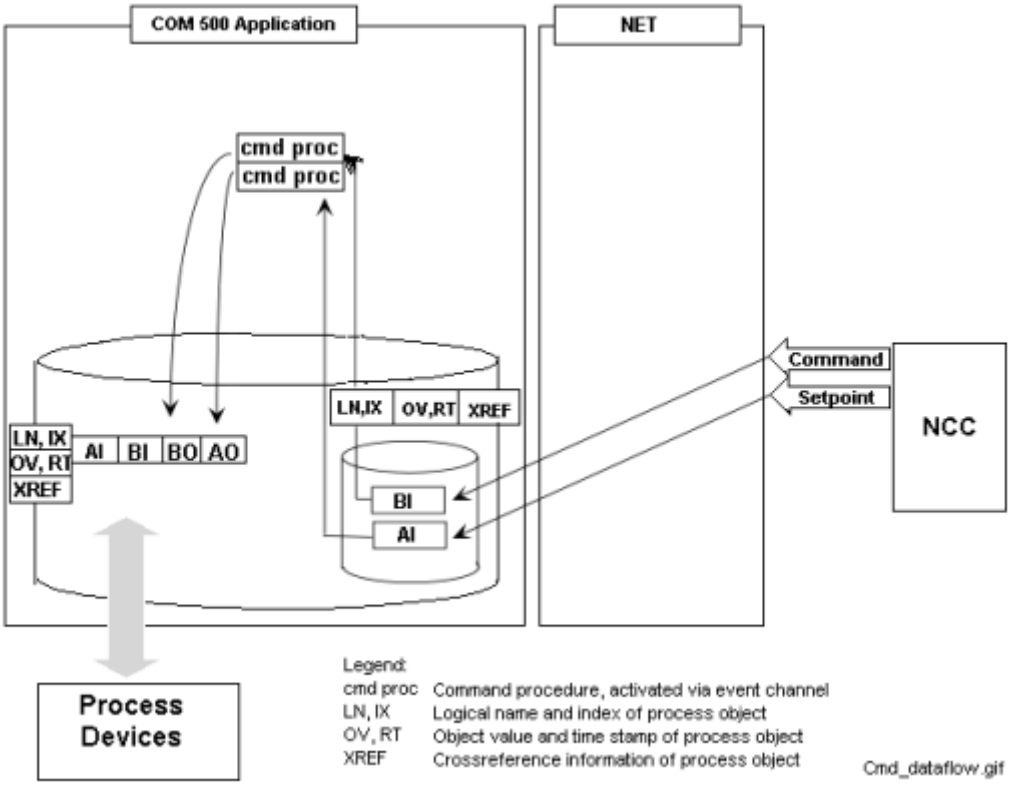

Figure 19. Command direction data flow of COM500i (ABB 2016f.)

Similar data flow process exists for the monitoring direction communication for measurements and indications from the process to upper levels. COM500i supports various communication protocols for master/slave communication between the system levels. This is the basis of the gateway functionality of the server and makes possible the linking of differing systems to one another. (ABB 2016f).

3.3 SCIL

SCIL (Supervisory Control Implementation Language) is the programming language used in application engineering and system configuration of the SYS600. With the language a wide variety of functionality can be implemented in SYS600. This functionality ranges from process control design, application user interface design and database processes to communications, system components management and process simulations. (ABB 2016e).

System configuration and application design are possible with SCIL. The SYS600 base system, which hosts the application specific functionality, can be configured with SCIL. In application design, SCIL supports both the functional design and user interface design. While the creation of databases and their application objects can be done with importing tools and standard library tools, creating the application objects and their properties can also be done manually with SCIL. (ABB 2016e).

Object accessing and handling is a basic feature of SCIL programming. The user can define programs that contain statements about the process application's objects for control and monitoring functionality. Functions for searching databases, reading values and writing information to objects are available for designing generic and application specific scripts. External applications can access SCIL data through protocol mapping services. (ABB 2016e).

In the below example script SCIL statements are used to demonstrate control and monitoring of process devices. Attribute values of a circuit breaker and a current measurement device are handled and output messages are shown afterwards.

```
#IF AA1E1Q1189A:PSS(10) == 3 #THEN #BLOCK
  #SET AA1E1Q1189A:POV(10) = 2
  t_Output = "Circuit breaker 89A opened"
#BLOCK_END
```

```
#ELSE t_Output = "Object not in fictive state"

#IF AA1E1Q11M:POV(10) > 50 #THEN -
-
  t_Output = "Phase L1 high current warning"
#ELSE_IF AA1E1Q11M:POV(10) < 30 #THEN -
-
  t_Output = "Phase L1 low current warning"
```

The first 9 rows represent the opening of a circuit breaker in a simulated process state. First the switch state (SS) attribute of the circuit breaker object named AA1E1Q1189A is read from the database, where the index of the signal object representing the position of the device is 10. If this attribute has the value 3, indicating fictive object state, the object value attribute OV will be set to 2, indicating the breaker open state. An output text variable t_output will be written with the information that the breaker has been opened. If the SS attribute has other value than 3, information about this will be written in the output variable.

In the latter part of the script the value of a current measurement object AA1E1Q11M is read from the database and possible warning messages are written to the output variable. An object value that is higher than 50 will result in the high current warning message and a value lower than 30 will result in the low current warning message.

Software tools for the SYS600 can be built with SCIL and integrated to the applications. The SCIL program developing tools are included in the SYS600 software, which means that writing, editing, testing and implementing SCIL programs to applications can be handled directly with SCIL. This gives the programming language an advantage over other external programming environments, as the compatibility of SCIL tools is optimal for SYS600 applications. Also the existing SYS600 tools created with SCIL provide useful information for developing new tools and methods.

3.4 Communication protocols in SYS600

3.4.1 Mirroring

Mirroring is an internal communication method of SYS600 applications. By mirroring process data can be shared between several applications in the same base system or in separate machines to build hierarchical systems. It is more efficient to use mirroring instead of a communication protocol when sharing data between applications and less engineering is required to set up the communication. Data can also be shared between applications in different domains like heating and electricity processes. (ABB 2016a).

Configuring mirroring requires defining host and image stations. The host is the source of the process data and contains connection to the process by protocols e.g. IEC 61850. The image is the mirrored copy of the host and receives the process data from the host station. All objects configured to the host station are mirrored to the image station so the process data is similar in both stations. Stations can also be configured as both host and image for building hierarchical systems. (ABB 2016a).

The main functions of the mirroring communication are (ABB 2016a):

- The host application replicates messages from the station device to each image application that has subscribed to the object address

- The process commands (#SET and #GET) executed in an image application are routed to be executed by the host application. The changed OV value is sent to the image applications by the host

- The host application replicates the system messages from the NET to each image application that has subscribed to the system messages

3.4.2 IEC 61850

IEC 61850 is the substation automation standard designed to enable advanced standardized communication technologies in substations and their automation systems. Main feature of substation automation standardization with IEC 61850 is the interoperability of products and systems: with standardized data models, configuration language, interface and mapping every component of the IEC 61850 system regardless of vendor can communicate with one another. (Giasis 2016).

MicroSCADA Pro supports IEC 61850 communication between the control system and station level devices. Signals from the devices are transmitted to the SYS600 base system through several components: from device to the IEC 61850 OPC Server, to external OPC DA client, to base system. This is visualized in Figure 20. (ABB 2016g).

The purpose of this specific system design is in the functionality of the components. An IED can't be directly connected to the base system with IEC 61850 because the base system is not designed for only IEC 61850 communication. The communication unit responsible for passing the signals to the base system is the OPC DA Client. This client needs to receive the signals from an IEC 61850 server to which several devices may connect. The client is connected to the base system with the ACP (Application Communication Protocol).

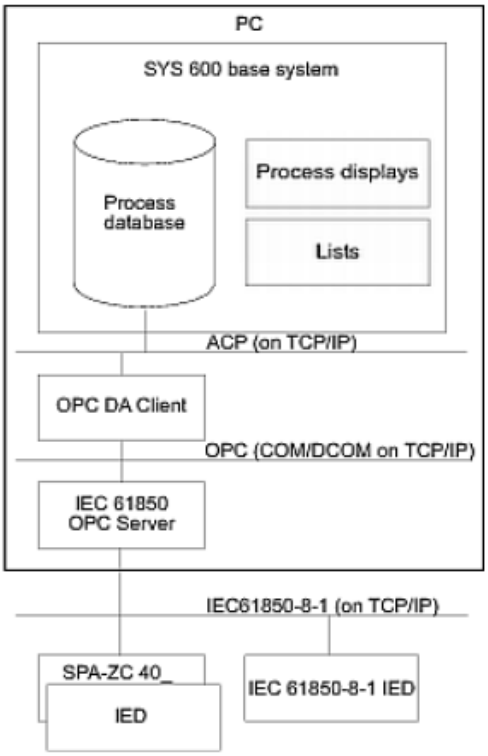

Figure 20. IEC 61850 system in MicroSCADA Pro (ABB 2016g.)

Designing an IEC 61850 system for MicroSCADA Pro requires configuring all the system components. This means creating all the necessary communication objects for the base system, configuring the OPC DA Client and OPC Server, and creating the application displays with the process objects. The objects created to MicroSCADA Pro databases are seen as IEC 61850 objects and their characteristics are used to control the application according to IEC 61850. (ABB 2016g).

This protocol is one of the most widely used for substation automation and is therefore important to consider for the testing tools design. As all these components are located in the same machine their management for testing purposes can also be investigated. The ITT600 is an IEC 61850 testing tool that is used by ABB for testing and simulation of IEC 61850 compliant IEDs.

3.4.3 IEC 60870-5-101 & IEC 60870-5-104

The 60870-5 series protocols are standards developed by the IEC for electric power systems communication technologies. They are developed to standardize communication methods for managing geographically widespread applications, for example several rural substations connected to a centralized network control center. The protocols enable connecting the SYS600 to process devices and upper level systems. While the protocols share many similar characteristics, they define different communication principles based on the serial and TCP/IP (Transmission Control Protocol/Internet Protocol) technologies. (ABB 2016h & ABB 2016i).

The 60870-5-101 protocol is usually implemented as the communication technique between the SYS600 and an upper level control system. Figure 21 presents this system architecture on a basic level. Data is forwarded from the base system to the IEC master system through a MicroSCADA Pro communication unit called the PC-NET. The 101 protocol describes coded serial bit data communication and applies the principles of serial communication to the controlled substation automation systems. (ABB 2016h).

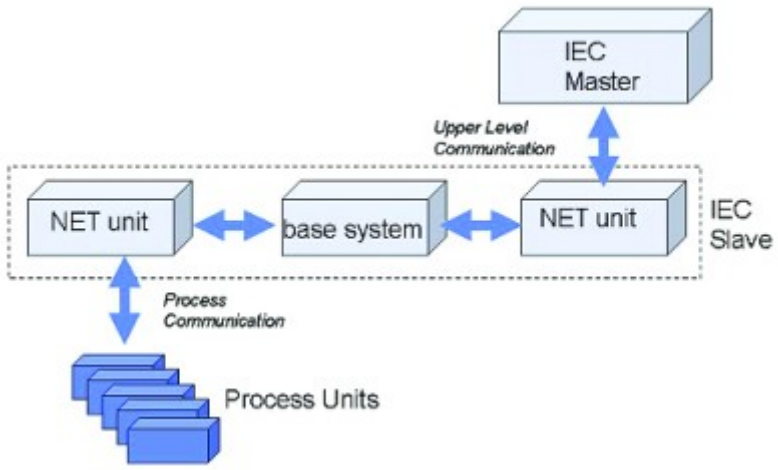

Figure 21. IEC 60870-5-101 communication to upper level system (ABB 2016h.)

The 60870-5-104 protocol is implemented to the system in a similar manner as the 101 protocol. The difference between these protocols is in the data transmission method: the

104 utilizes TCP/IP based transmission while the 101 is based on serial communication. System architecture has the similar components as presented in Figure 21. The communication modes of these protocols can be based on the unbalanced or balanced modes. In unbalanced mode only the master station can initiate data transfer by polling the slave station. Balanced mode allows both stations to initiate data transfer. While the 101 supports both modes, the 104 supports only balanced mode. (ABB 2016i).

These protocols are commonly used in MicroSCADA Pro systems and be applied to the testing tools functionality. For instance network control center applications can be tested with these protocols along with process communication testing, which results in comprehensive system tests.

4 TESTING OF MICROSCADA PRO SYSTEMS

4.1 Software testing

Software testing is the process of confirming that software works as specified and does not implement any unexpected functionality. The user of the software must know what to expect when using the software for the purpose it was designed for. During the development processes of software an unknown amount of errors and defective behavior is induced to the software. A proper approach to software testing is therefore to assume that the software has errors and design the tests to find as many of these defects as possible. (Myers, Sandler & Badgett 2011).

Exhaustive testing, meaning total test coverage of the software, can be in practice impossible. This means that designing effective test cases is important for finding errors in the software. Test cases are designed to test a part of the software functionality with case values or settings that should result in correct or incorrect execution of the software. Logical test case design, instead of using random inputs, is considered an effective methodology for finding errors in software. (Myers et. al. 2011).

Automating tests can reduce the time and effort used for software testing. Manually running large numbers of similar test cases can make the testing processes inefficient. The benefits of automating tests are especially seen in development processes where new versions and configurations of the software are produced constantly (Mantere 2003.) and the internal quality of software must fulfill the criteria for the potential shippable software increment (Bonsanque, Broek, Chaudron & Merode 2014.) This can be the case in e.g. agile software development environments, where new versions of products are developed in fast-paced development phases.

Testing strategies can be classified to white box testing, black box testing and grey box testing depending on test perspective. White box testing includes tracking of programming code and the internal structures of the software. Black box testing focuses on the

functional specifications of the software, tracking the inputs and outputs that are produced during tests. Gray box testing is a strategy combining white box and black box testing methods. (Mantere 2003).

Software test methods associated with the testing strategies with examples are presented in Table 4. (Robinson 2008).

Table 4. Types of software tests.

Test strategy	Test methods	Examples
White box	Control flow –based Data flow –based Conditions, decisions Branches, links, paths	Code review Static analysis Unit testing
Black box	Requirements-based Use case –based Scenario testing Statistical testing Random testing	Operational testing Functional testing Customer profiles
Gray box	Dependencies/relations Communications Protocol-based	Sequence and state-based testing

The methods of Table 4 with the right strategy lead to decreased costs, increased effectiveness and better management of testing. Main target is to detect software errors earlier in the development process where they are cheaper to fix. (Robinson 2008).

One of the main principles of software testing is context dependability. Software that is used in safety-critical systems will be tested with a more intense effort than software used for e.g. entertainment purposes. This context dependability also means that the

software tests must be kept up to date with the changes in the software versions and the evolution of the environment where the software is used. (Homés 2013).

4.2 Test specifications

Designing successful tests includes specifying test cases for the system to be tested. Specifications can be found from various sources related to the field of the system under test, e.g. the fields of substation automation and power grids. The sources can provide specifications with differing criteria based on principles such as safety and performance. Sources can include standards, company specific, customer specific and project specific information for specifications.

Regulatory and standards: these sources provide regulatory and standardized information which can be used to define test specifications. The regulatory requirements are usually concerned with the safety aspects of the system and may provide mandatory requirements for test case design. This means that in e.g. developing safety-critical systems a standard must be followed to comply with the regulatory requirements and the test cases are designed to pass the requirements presented in the standards (Heeager & Nielsen 2018.) The specification provided by standards can go into details of e.g. communication protocols used in the system and therefore offer information for critical safety and performance testing.

Companies define their own testing processes which are designed to provide the best business value for the company and the customers. Testing processes can include internal and external tools and methods for testing. Internal methods can be more specific to company technologies and products for more accurate testing of e.g. functionality and performance. External methods can be used for more general testing with time and cost efficiency benefits.

Customer and project specific testing is to design the tests according to customer requirements and project specifications. Companies can have standard testing procedures

for certain products and systems, but often there are needs to specify the test cases specific to the project. This can be the case in situations such as retrofitting new automation products to existing substations and combining products from different vendors to test interoperability's. No surprises are wanted at the last moments of systems being handed to the customer and this leads to systematic and exhaustive testing methods. The use of external consultant reviews of the system on behalf of the customer is common to ensure the quality standards of the project.

4.3 MicroSCADA Pro testing

4.3.1 Product testing

MicroSCADA Pro products are tested on a regular basis with automated testing tools for verification, validation and bug fixing. Testing activities can range from regression and sanity testing to performance and stress testing. Testing strategies are used to define how the tests are to be completed and organized. The processes used include waterfall and agile testing methods. (Jinesh CJ 2019).

While the test strategy target is to design tests to be automated there are still manual steps included in the testing processes. On MicroSCADA Pro backend side there are no comprehensive automated SCIL or Visual SCIL based tools to test the system parts which are built with these internal programming languages. For example process database values are controlled manually by utilizing specific scripts or similar value forcing. (Jinesh CJ 2019).

Testing setups of station level processes may include physical devices and simulation tools. Depending on the protocols used these setups apply the MicroSCADA Pro functionality for communication and process control and monitoring. System setups with NCC level communication are also tested and with these the similar protocol specific behavior is tested. At the moment testing for the communication gateway functionality

(COM500i) is not specifically conducted with any testing tools designed for this purpose. (Jinesh CJ 2019).

Reporting and managing of tests is handled by third party applications, which store the information in servers accessible to the testers and management. Information about tests and their results is available in a standardized form for tracking of testing activities. The applications provide the methods for bug reporting and fixing. In addition to the main applications used for managing tests, setup specific testing software is used. (Jinesh CJ 2019).

Similar testing as in the development that is the subject of this thesis can be found in the existing processes, but the SCIL-based functionality could be seen as an addition to existing practices. The interfaces used for testing at present cannot directly access the SCIL-built parts of the system, which would need their own tools to automate some of the test processes. The testing scripts being used can offer information about usage of SCIL within the ongoing testing processes. (Jinesh CJ 2019).

4.3.2 Station level devices in projects

The testing of station level units in MicroSCADA Pro projects is done for each unit depending on the configuration. For example RTU units, which are positioned between the SCADA system and the bay or process level devices, can function as a station level unit and as a link between the process and the SCADA system. With these types of configurations, two-way communication is tested for the system. Signals are generated from the SCADA system to the process and correct signal response is confirmed from the RTU side. The communication to other direction is tested in a similar way and the correct information is confirmed at the SCADA system side. The testing with the actual physical process devices takes place in later project stages. (Pirhonen 2018).

Methods of testing include individual signal testing and sequence testing. The interview did not provide information about the sequence methods, but it can be assumed these methods include logical sequences according to customer project needs. The station lev-

el units to be tested are manually configured and this means there exist no general automatic testing methods. (Pirhonen 2018).

Analog and digital signals are tested depending on the device configurations and used communication protocols. Most commonly used protocols in the tests include the Ethernet-based IEC 60870-5-104 and its serial counterpart IEC 60870-5-101. Modbus is also used occasionally. IEC 61850 is used, but was determined to be somewhat complex to implement. Pirhonen (2018) mentioned that dealing with the communication protocols is an impediment in the testing processes and he presented ideas for a protocol analyzer. The analyzer could figure out the communication issues in the system instead of manual interpretation of the received data. (Pirhonen 2018).

Pirhonen (2018) didn't mention specific issues regarding scalability testing of projects. Limitations may rise for example from device transfer capacities and speeds. Railway electrical systems were mentioned as an application, where the amounts of process signals and data points are substantial. These types of systems contain various station types such as feeder stations and switching stations. The number of station level devices can rise to several hundred in largest applications. Testing of primary processes scalability is presumably not conducted during station level device testing. (Pirhonen 2018).

A laborious working phase in projects is the creation of databases for large applications, and automated solutions are in demand for this phase. During primary process testing phases the test objects are also created manually to the databases and used to confirm the functionality of the station level devices. Depending on the project the databases may require various testing objects, and automation possibilities of this are unclear. Customer projects contain confidential information, which means their data is not directly available for development of the testing tools. (Pirhonen 2018).

The confidential customer information in projects is generally protected by the General Data Protection Regulation (GDPR) of the European Union. This regulation protects the confidential data related to the project's assessed system, system's operators and intermediary processes. (Jamil, Daud & Patel 2019).

4.3.3 Automated tests and sequences

Automated tests and sequences can be used in MicroSCADA Pro projects. Test automation can be applied to various applications with customized process objects and SCIL-based scripts can be used to execute the test sequences. In these cases the SYS600 tools for process database management and SCIL programming are used. Testing can take place at the control center level or substation level depending on the used devices and configurations. Depending on application the automated tests may need complementary manual verification. (Laine 2019).

For example automated tests can be applied to the control of fire security systems and their dampers. These dampers have similar principles in functionality as the switching devices in electric power applications. The automated test sequences are used to open and close the dampers in a functional part of the system and the results are monitored in the SCADA system process object values and picture symbols. Results can be logged to files for analysis and reporting. (Laine 2019).

By using sequences the operator can execute several control tasks in the system with a single action. For example switching devices can be operated as a sequence to separate a part of the electrical system from the powered parts. These sequences involve the methods of sending the control commands and verifying the successful operations. Before applying the sequences to the actual process the functionality can be confirmed by simulation of the process objects. The sequences can be written as SCIL scripts and added as objects to the process display pictures for operation. (Laine 2019).

The information from the use of automated tests and sequences in projects shows similarities to the targets in development of the testing tools. Project applications will contain devices and objects that require customization, but the general functionality could be achieved with SCIL and SYS600 features.

4.3.4 Factory acceptance & site acceptance testing

Factory Acceptance Test (FAT) and Site Acceptance Test (SAT) are completed in customer projects for verifying the delivered system's conformance to requirements. FAT testing is meant to test the system at factory level and check systematically that everything works as the customer intended. SAT testing takes place at the customer site where the actual components are connected to the delivered products and the system undergoes final verification processes. (Esala 2019).

Depending on the project the tests include hardware and software components. Generally during FAT testing parts of the system are simulated while at SAT testing the SCADA system is connected to the actual devices. Because the system can include components from various manufacturers, single customer side devices are used at FAT stages for verification. The required testing coverage is usually defined by the testing plan and the customer expects that to be sufficient. (Esala 2019).

Simulation methods include the use of protocol specific tools and MicroSCADA Pro SCIL methods. For example the IED simulation tool ITT600 is used for IEC 61850 simulation. SCIL is used for creating scripts for station level protocol slave devices. For upper level systems, e.g. IEC 60870-5-104 communication, another MicroSCADA Pro system can be used as the master device. Developing the simulation methods in projects FAT and SAT testing has been based on using existing MicroSCADA Pro functionality and there has been no active co-operation with research and development in the matter. (Esala 2019).

New testing tools could be used in FAT and SAT testing to improve the simulation methods. Slave devices that are simulated with MicroSCADA Pro scripts would benefit from a tool that sets values to the simulated devices and communicates with the upper level system. More detailed information about the capacity and scalability of the system would also be useful. (Esala 2019).

4.4 Internal testing tools

4.4.1 TestRun

TestRun is an internal tool for testing SYS600 applications. The tool can be used to simulate the indication process objects with all values that are applicable for the process object type. Binary, double binary and analog inputs can be tested simultaneously with defined execution intervals and test actions. The test actions can be e.g. flip/flop value changes or scripted test sequences. The test run time tracking is provided with outputs view, and statistics feature can be used to view graphical information about the tests.

The features of TestRun tool can be used to simulate operational situations in SYS600 applications, simulate cyclical system behavior and track the tests execution. Test automation is possible by creating cyclical scripts to be executed with defined intervals.

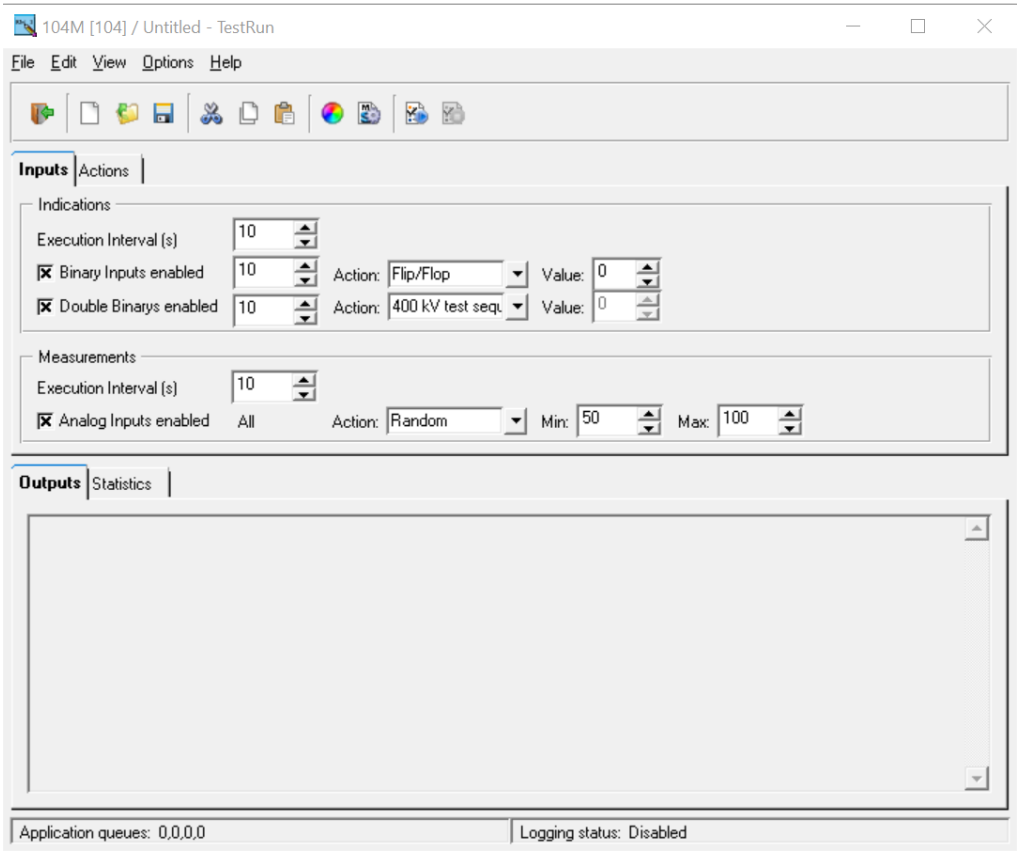

Figure 22. TestRun main view.

4.4.2 ITT600

ITT600 (Integrated Testing Tool) tools consist of the ITT SA Synthesizer and the ITT SA Explorer. These tools are used for testing and commissioning of IEC 61850 compliant substation automation systems. The Synthesizer tool is used to simulate the functionality of IEDs in 61850 substation environment by applying simulation models and scripts to imitate real IED behavior and network traffic. The Explorer tool is used for diagnostics and analysis of the network traffic and applications in IEC 61850 systems.

With the ITT SA Synthesizer IEC 61850 IEDs can be simulated from the same workstation where the MicroSCADA Pro applications are running. This enables the setup of simulated process level devices conventionally connected to the SCADA system to be tested. The tool can also be used from a separate workstation and connected to the SCADA system by TCP/IP communication. If necessary the Synthesizer tool can manage both real and simulated IEDs to form extensive substation automation test setups. An example setup is visualized in Figure 23.

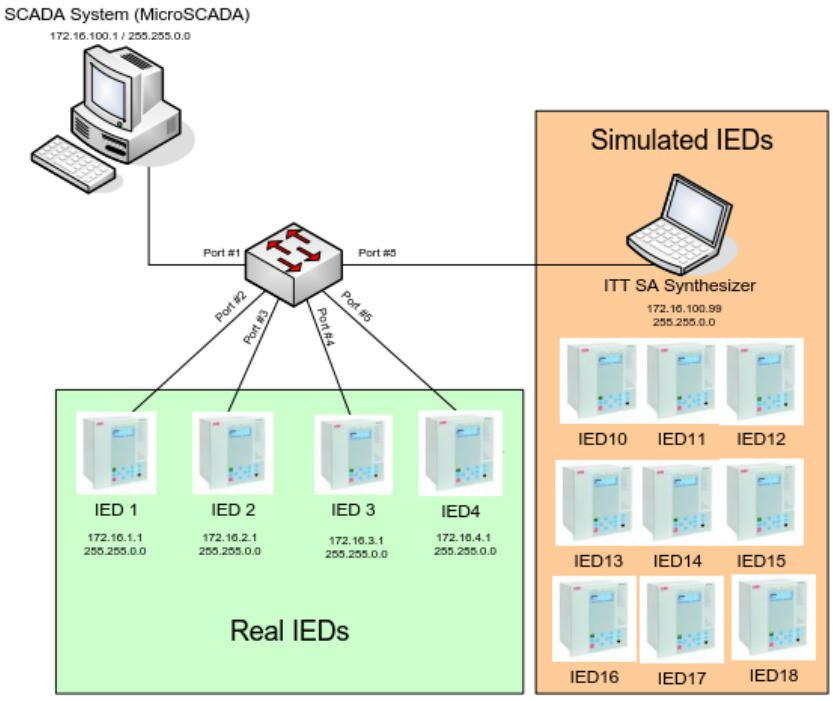

Figure 23. ITT SA Synthesizer test setup example.

The SCADA system connected to ITT SA Synthesizer IEDs sees no difference between the simulated or real IEDs. This makes possible to create realistic process control scenarios where IEDs behave as they would in real substation control systems. Indications, such as measurement values, can be sent to the control systems with actual IEC 61850 communication and control commands can be issued with remote control protocols to the simulated IEDs by communication gateways, such as a COM500i application in SYS600.

Scripts in ITT SA Synthesizer can be used to engineer logical behavior to the simulated IED models. Sequences where multiple indications or commands are used can be combined to simulate operational situations, e.g. switching sequences with interlocking schemes. Script functionality includes methods, which can be triggered by control commands sent from i.e. remote control SCADA system. Communication between several IEDs is possible by GOOSE message synthesizing.

Main functions of the ITT SA Explorer include the analysis of single IED configurations and the analysis of SCL models, for example bay or station level models. In addition to this, the tool is an IEC 61850 network analyzer which can be used to explore Ethernet, GOOSE and sampled value (SV) traffic in the network. The features of the tool make it useful for analyzing the tests executed within the IEC 61850 process level. The data can be presented in practical forms for fast analysis, for example as graphs like in Figure 24.

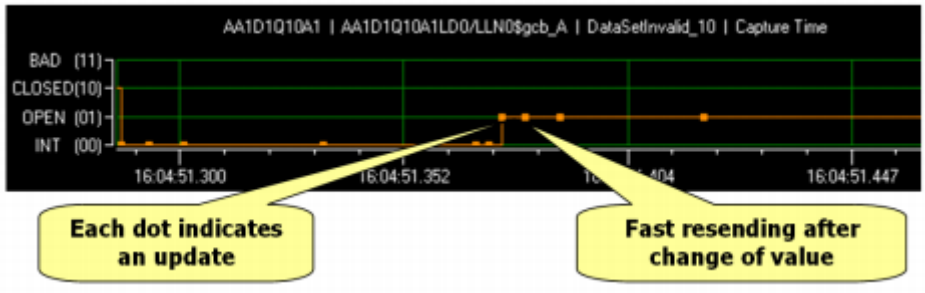

Figure 24. ITT SA Explorer GOOSE traffic capture.

4.4.3 COM500i tester

The COM500i tester is a tool used for signal testing of cross-referenced signals to the monitoring direction in COM500i applications. The tool recognizes all configured signals which are connected to upper level systems. These signals can then be simulated by manually changing the object values of the signals by using the functions of the dialog. Patterns can be created to run sequences of changing values.

The signal simulation results can be verified from the upper level system or the COM500 diagnostics functionality can be used to track transmitted signals. The tool provides no outputs tracking or statistics from the executed tests. The functionality of the tool could be used to test specifically the COM500i communication in test system setups.

An addition to the test functionality of the tool would be to include the signal reliability in the tests. This means that the objects could be tested with various object status values together with the object values. To achieve this, the Object Status (OS) attributes of the objects could be simulated with similar functionality as the Object Value (OV) can be simulated at present.

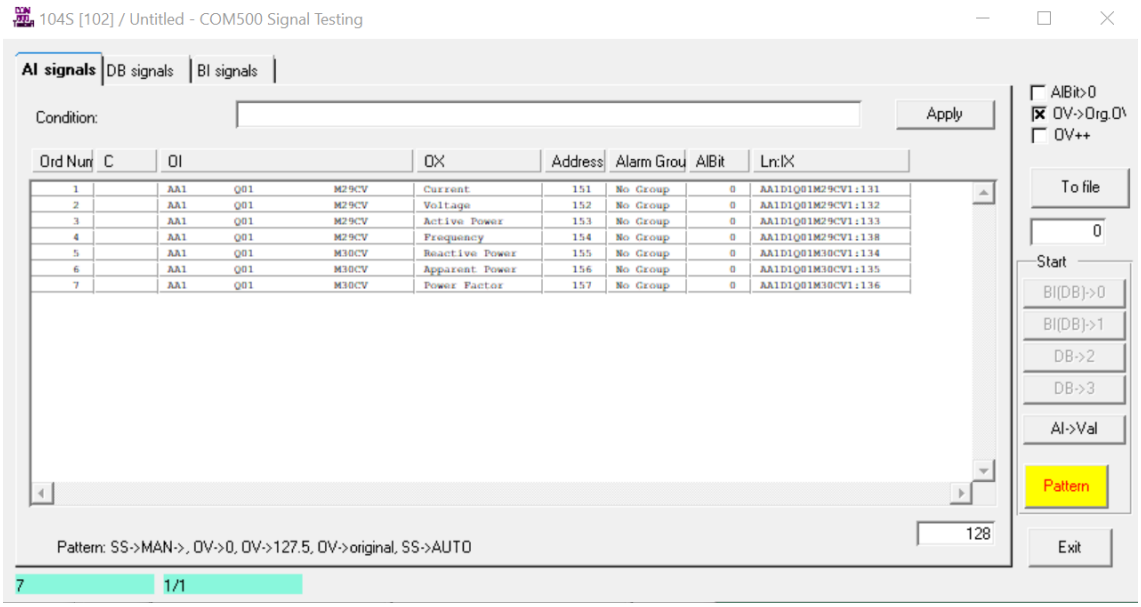

Figure 25. COM500i tester view.

4.4.4 SATEEN

SATEEN (Substation Automation Test Environment) is an internal test environment tool used for testing substation automation products. The tool includes features to enable a high degree of test automation. The SATEEN software architecture consists of two major parts: the generic testing framework TEEN which could be applied to test systems in practically any domain, and the SA libraries which contain the features for testing substation automation systems. The test environment creates extensive models where the system under test, containing various physical and virtual elements, can be modeled as one system. This is called a context in SATEEN and it is presented in Figure 26.

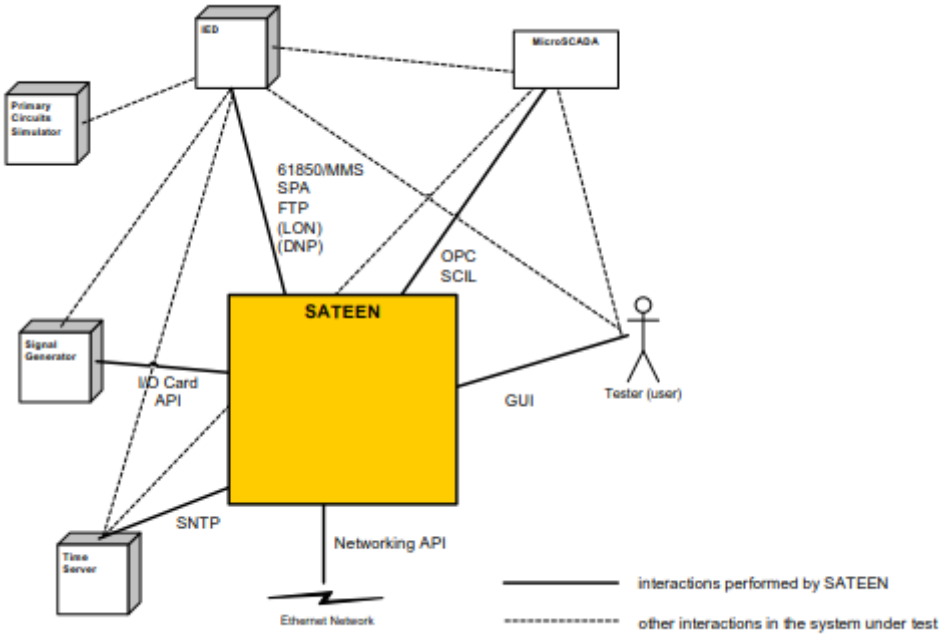

Figure 26. SATEEN context system view.

The SATEEN is developed to test complete substation automation systems where MicroSCADA Pro products can be a part of the system under test. The tool includes specific libraries for MicroSCADA Pro among other components such as IEDs and signal generators. The tool can connect to a MicroSCADA Pro application on a remote machine to perform e.g. switching operations as steps in a test case.

5 DEVELOPMENT ARRANGEMENTS

5.1 Test environment specification

The test environment consists of the system to be tested, the testing tools and other related components. Specifying these parts properly is grounding for defining the development approach and defining the test functionality to develop.

A single substation automation test environment can include components from the process level all the way to the control center products. This is the chosen approach in this work and a test system with extensive hierarchy is presented in Figure 27. The system comprises of a wide range of MicroSCADA Pro functionality and enables the creation of operational test cases. Explanations for components are provided in Table 5.

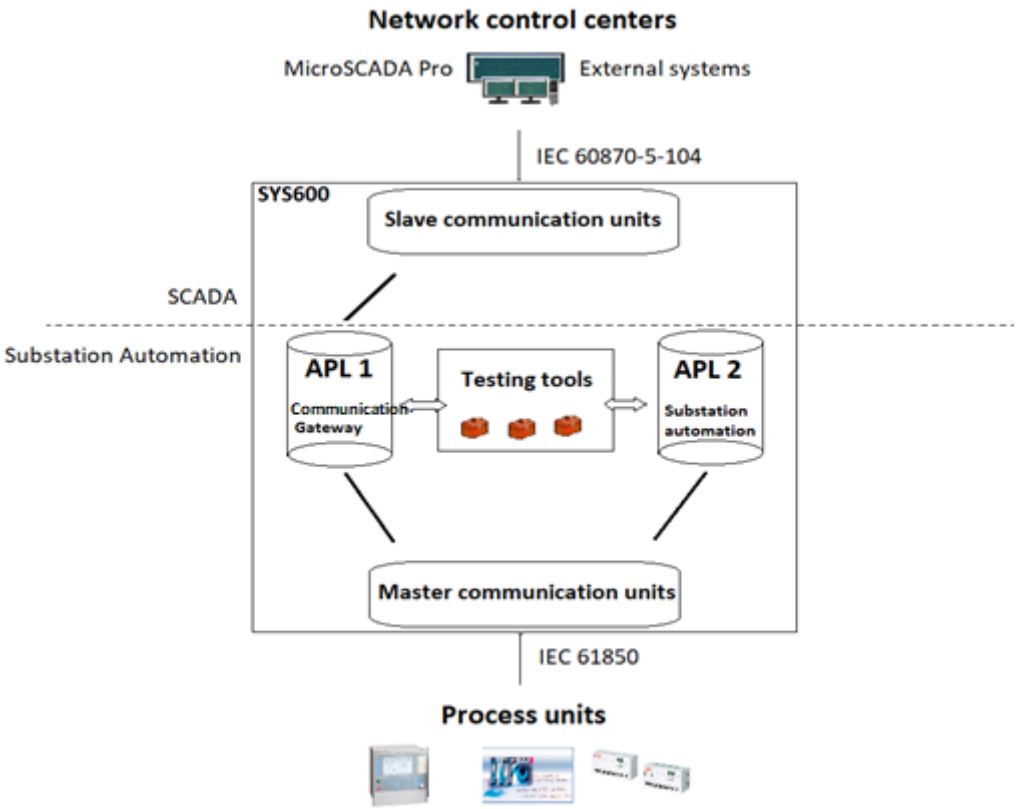

Figure 27. Test system components.

Table 5. Test system component explanations.

Component	*Explanation*
Process units	Units linking the controlled process to the MicroSCADA Pro system, such as IED protection relays, RTUs, PLCs or their equivalent simulation models controlled with testing tools.
Master communication units	MicroSCADA Pro communication units connected to the process. Command signals are transmitted to the process and monitoring signals are received from the process. Protocol e.g. IEC 61850.
Testing tools	Developed tools to execute test cases in the system. Positioned within the SYS600 base system and run from applications for enabling SCIL and Visual SCIL test functionality.
APL 1 **Communication gateway application**	Application where COM500i handles gateway communication between the SYS600 and upper level systems. Cross-referenced signals in monitoring and control direction.
APL 2 **Substation automation application**	Substation automation application for station level process control. Connection to the station level devices via the master communication units. Combined testing with the COM500i application.
Slave communication units	MicroSCADA Pro communication units which connect the SYS600 to upper level control centers. Monitoring signals are sent to the upper level and command signals are received to the SYS600. Protocol e.g. IEC 60870-5-104.
Network control centers	The remote operation control centers for remote operation testing. Control center products from the MicroSCADA Pro product family or external vendors.

The test system can be divided to the substation automation and SCADA sectors. The substation automation sector contains the functionality typically included at the substation level in practical applications. The SCADA sector includes the remote control functionality of the SYS600 and the network control centers. These sectors can be used to view the test system as two functional entities where separate or combined test cases can be designed.

5.2 Test system configuration

An initial test system was configured before starting the development process. This system features three applications in one SYS600 base system according to the structure presented in the system model picture in Figure 27. All applications are configured to monitor and control the same substation process at different control system levels with communication protocols according to the structure.

Application 1 (APL 1) is configured as the substation automation application. It is connected with the IEC 61850 protocol to the station level devices simulated in the ITT600 SA Synthesizer simulator. This application contains a single substation bay including switching devices, measurements and protection functionality. The bay is replicated to the process databases of the two other applications. Configuration is imported from an SCD (Substation Configuration Description) file according to IEC 61850.

The first application setup is presented in Figure 28.

The bay process picture is presented in Figure 29. The bay contains an incoming line feeder component, which could represent connection to a station busbar. Main bay circuit breaker is positioned between two disconnectors. After the circuit breaker are the measurement devices for measuring the values presented on the right hand side of the bay. As the last device an earthing switch is included for the bay. Outgoing line indicator can be used to show the power flow from the bay.

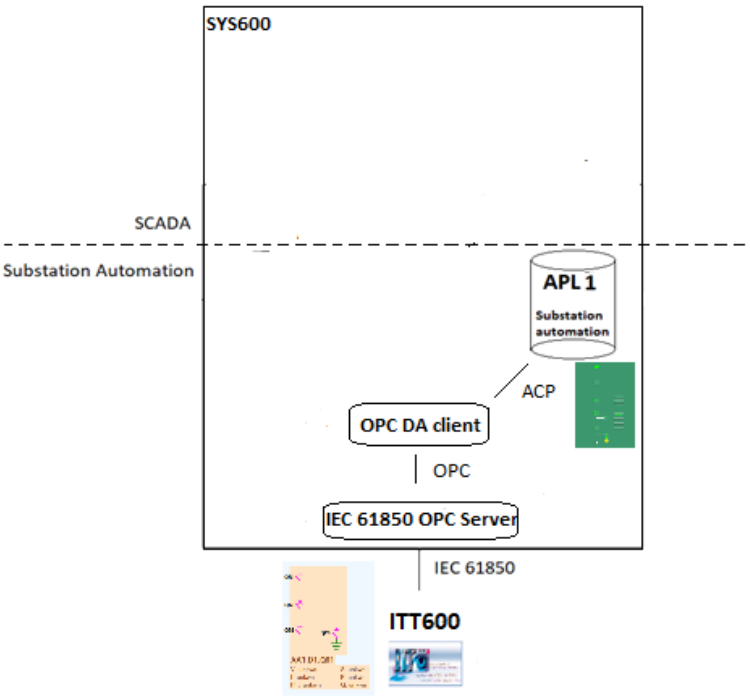

Figure 28. First application configuration.

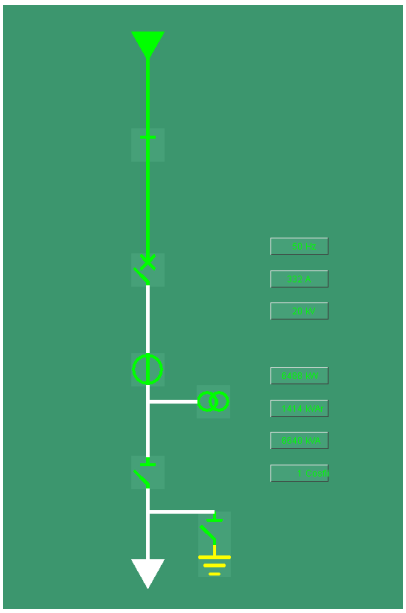

Figure 29. Test system bay process picture.

The objects in this bay are MicroSCADA Pro visual appearances of the objects, as can be seen in Figure 30 containing the input and output objects for the bay circuit breaker. The most important objects for the monitoring direction are the position indication and

interlocking conditions, while the selection and execution objects are applied for the control direction. Figure 31 contains attribute details for the position indication input object.

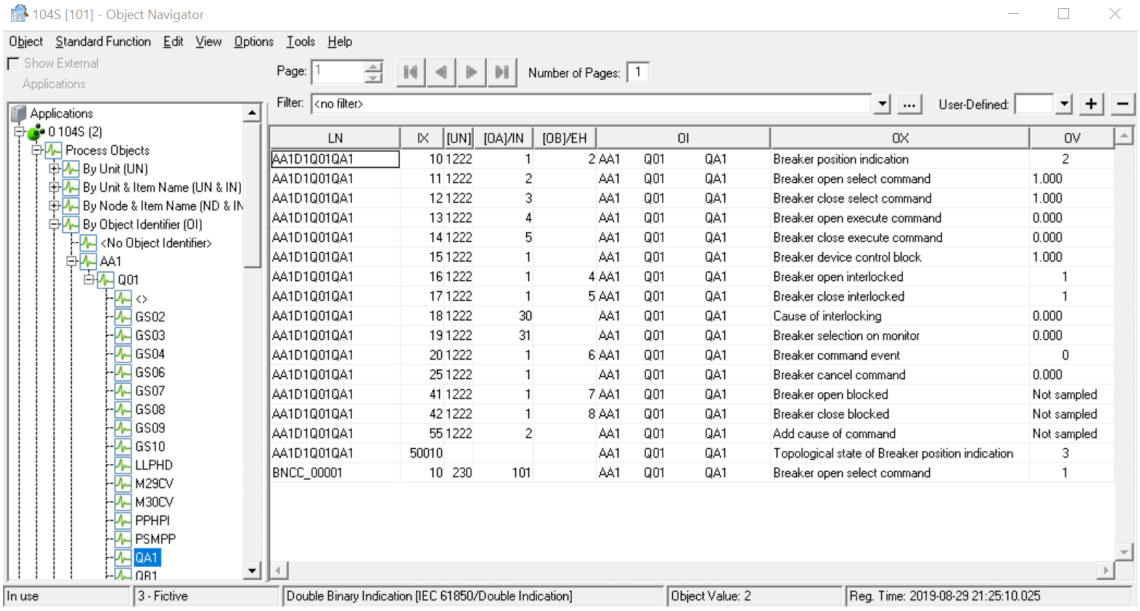

Figure 30. SYS600 Object Navigator view for test system circuit breaker.

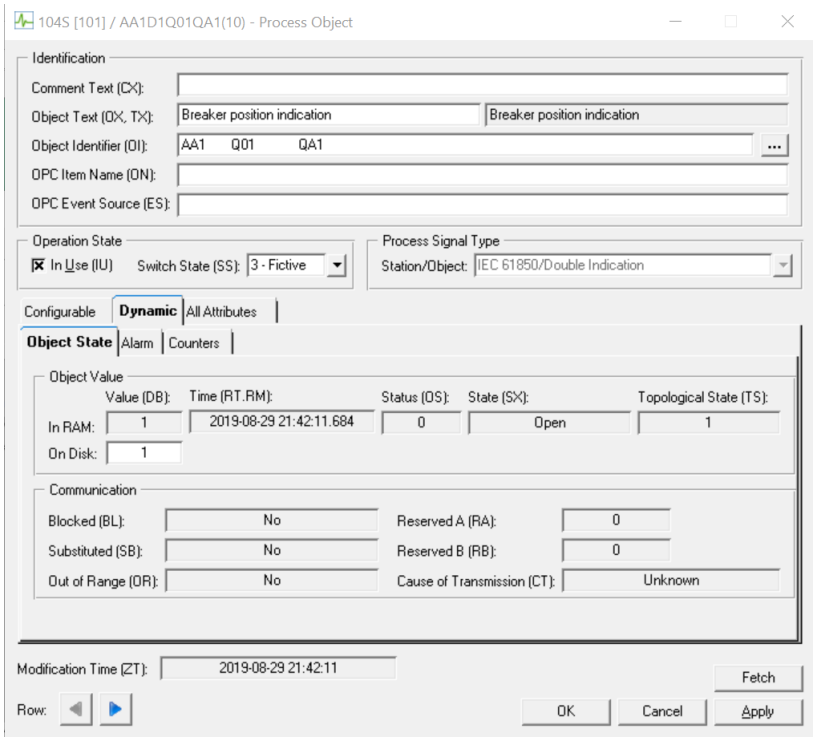

Figure 31. Breaker position indication object attributes.

The test system application two (APL 2) is configured as the gateway application (COM500i). It receives process data from application 1 by internal application mirroring and is connected as IEC 60870-5-104 slave to the master application three (APL 3). This configuration makes it possible to simulate master-slave communication of IEC 60870-5-104 within the same base system as if the application was connected to a real remote system. APL 2 is presented in Figure 32.

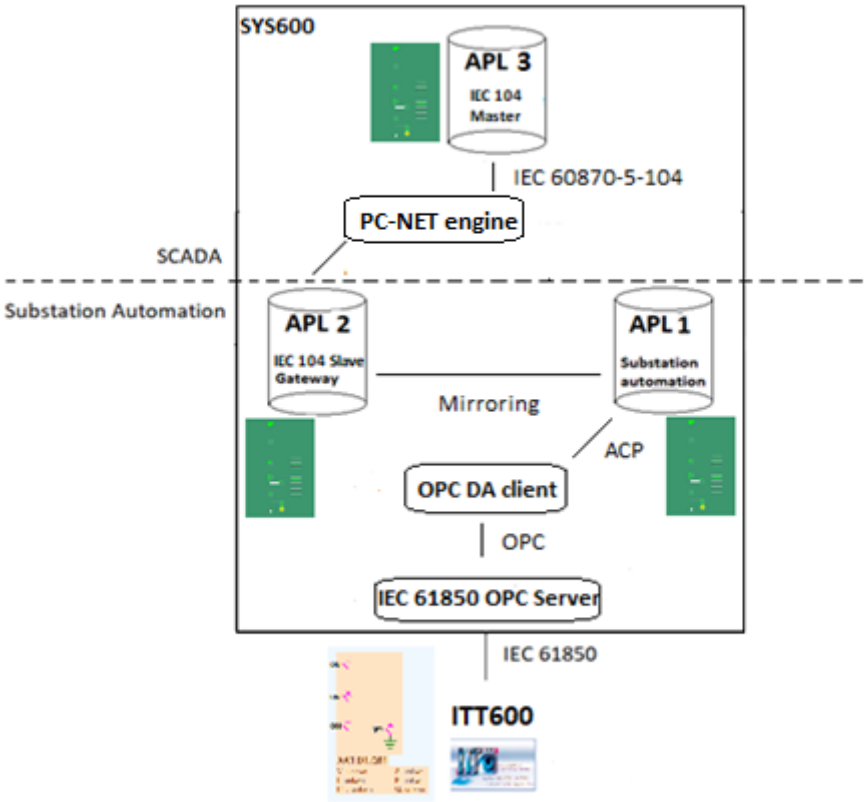

Figure 32. APL 2 configuration.

The motivation to set up the test system with this simple setup is to first test the communication and focus on the core functionality of the testing tools. The goal is to enable communication from the process all the way to the network control center level and then start developing tests with that setup. After successfully connecting the components of the bay, the system can be scaled up by creating new objects or importing larger configuration files with more devices.

The configuration of the application connections was done with the SYS600 System Configuration Tool. The tool is used to configure the base system to include the necessary internal system and communication objects. Connecting the process devices is based on stations, which represent common devices or endpoints for the process objects. The stations can be e.g. RTU devices or IEDs. In this setup there is a single bay representing one station level device, which requires one station object in the base system for each communication link between applications. Figure 33 presents the station configuration for the application 2, where the IEC 60870-5-104 slave station is configured with the System Configuration Tool.

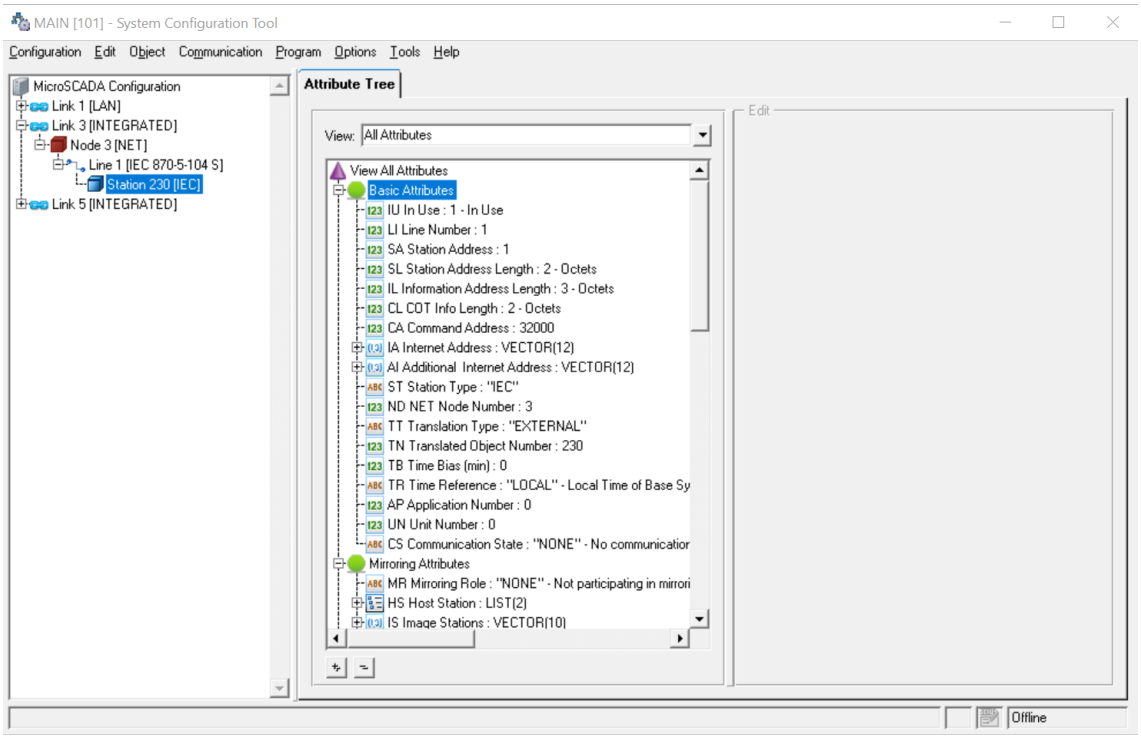

Figure 33. System configuration for the IEC 60870-5-104 slave station.

The communication gateway (COM500i) is configured for application 2 with cross-referenced signals to application 3. The cross-reference tool Signal X-References is shown in Figure 34. The table includes configuring the signal types, command groups, signal purposes, related indication signals, addresses and signal handling information. With this configuration the communication gateway can handle the received command signal from the NCC application 3 and forward it to the process device.

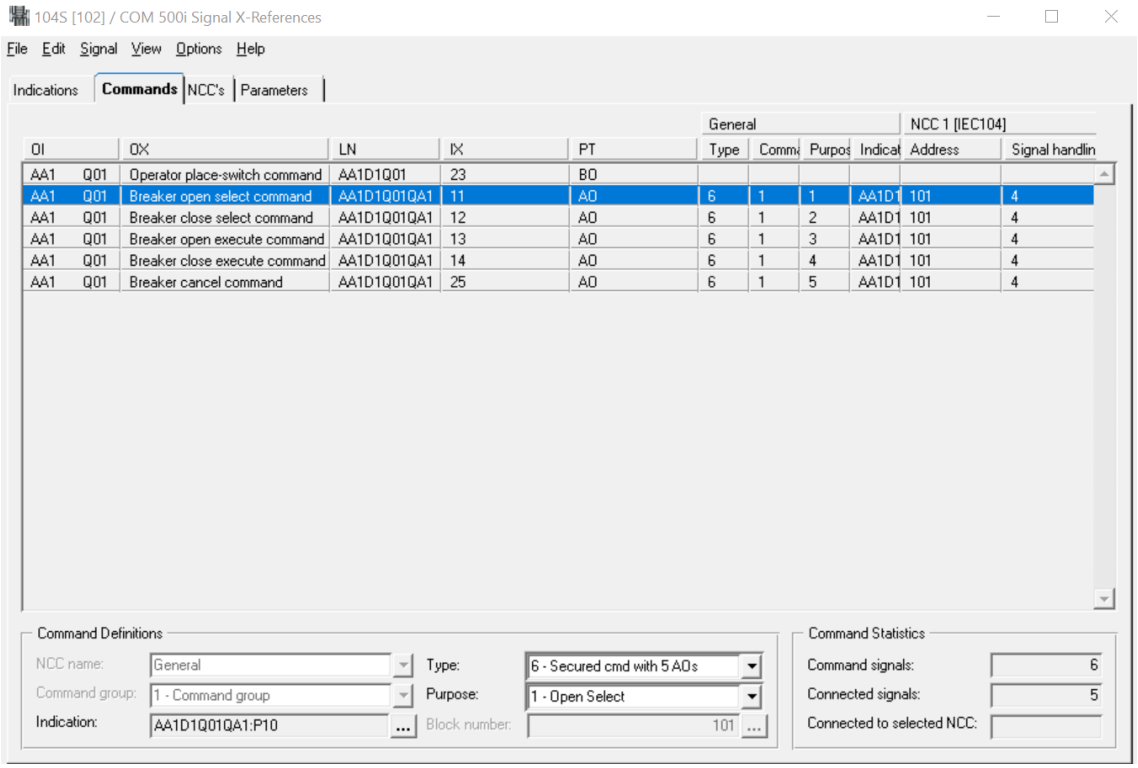

Figure 34. COM500i Signal X-References tool in application 2.

Application 3 is the network control center application and IEC 60870-5-104 master connected to the IEC 60870-5-104 slave application 2. It is used to simulate the remote NCC functionality. The application receives process data upstream from the COM500i in application 2 and sends control commands downstream to application 2, from where they are routed to the process units by the COM500i and internal mirroring between applications 2 and 1. Configuration for this application was done in a similar way as with application 2, this time the station object was configured as an IEC 60870-5-104 master station.

Figure 35 presents the complete test system with connections. The layout of the system is very similar to the test environment presented in Figure 27.

In Figure 35 the test system includes communication from the bay IED, simulated with ITT600, to the IEC 104 Master application. The bay IED could be connected to the process devices with e.g. signal wiring or an Ethernet process bus. The IED in ITT600 connects to SYS600 application 1 with the IEC 61850 using the IEC 61850 OPC Server

and the OPC DA client. The data is forwarded to application 2 by internal mirroring where application 1 is the host and application 2 is the image. From application 2 the data is forwarded to the PC-NET engine and from there to application 3 with IEC 60870-5-104 protocol. COM500i provides the gateway functionality here. The communication is set up and tested for both upstream (monitoring) and downstream (control) directions.

The testing tools can be run in any of the applications for application specific testing. As all applications contain the same bay setup, the tests can be run on various levels.

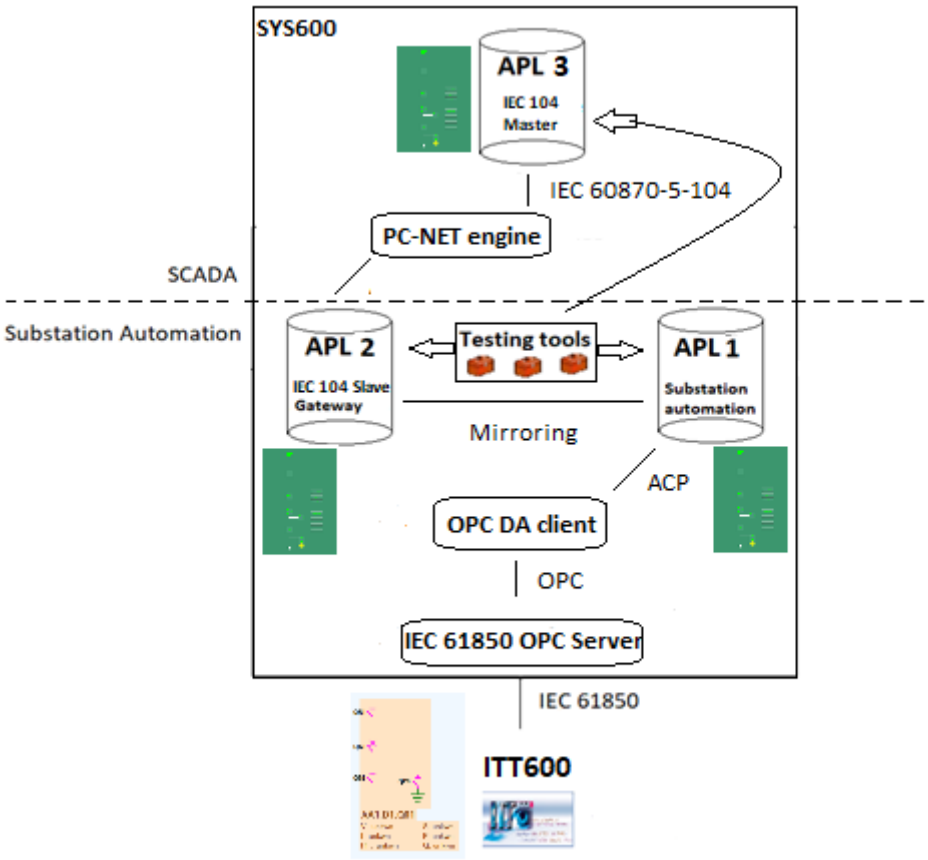

Figure 35. Test system configuration.

5.3 New testing functionality

The information from the existing test processes and tools shows what could be implemented as new testing functionality. Information was collected via interviews, discussions with R&D and experiences with the existing testing methods of MicroSCADA Pro products. There exist multiple teams performing testing activities within various processes applying several testing tools and test strategies. Main target is to identify the new functionality that would provide meaningful business value and future development possibilities. The following list of functionalities is considered:

- Operational testing with main protocols in hierarchical systems
    - Logical testing in operational situations for both monitoring and control directions to test several hierarchical levels, e.g. NCC and substation levels in the test system with the new tools
- Scalability testing in operational situations
    - Test the scalability of processes when e.g. new bays or busbars are added to a substation control system
- Automated process database testing scripts
    - Search and test objects in databases to automate the testing of all objects in an application by object type, e.g. test all circuit breakers or all current measurement objects
- Integration of internal tools for combined testing
    - Integrate existing tools to combine features and functionalities

5.4 Tools implementation

Implementing the tools is considered in two main approaches: how the tools are implemented to the MicroSCADA Pro products and how are they implemented to the testing processes.

The tools are implemented to MicroSCADA Pro products using the methods of ABB R&D for implementing new functionality to existing products. This is known as system implementation. The system implementation will consider what the development process will bring to the products and how: what are the requirements for the new testing tools, what is the new or changed functionality, description of the implementation in detail, security and compatibility information etc. The information is gathered to a system implementation proposal document, which is written for the testing tools in this thesis work. In this implementation proposal the new testing tools were initially named “Operational Situations Testing Tools”.

Testing processes take place during various stages of business activities from the R&D of the products until the commissioning and acceptance tests for the customer. The tools used in these processes can be either used for general purposes, or provide features for more sophisticated applications. Implementing new testing tools to these testing processes means considering what the testing tools will bring to improve the process and if the information from the test results is applicable in the process.

6 DEVELOPMENT PROCESS

6.1 Development approach

Agile development methods are commonly used in software development environments. They involve cyclical development, strong customer focus, quality assurance methods and continuous testing processes. Agile methods are especially suitable for iterative and incremental software development where new or changed functionality is introduced in existing products. In agile development the testing is frequent and defects can be tracked to short development cycles. Figure 36 shows the similarities and differences between traditional waterfall and agile development processes.

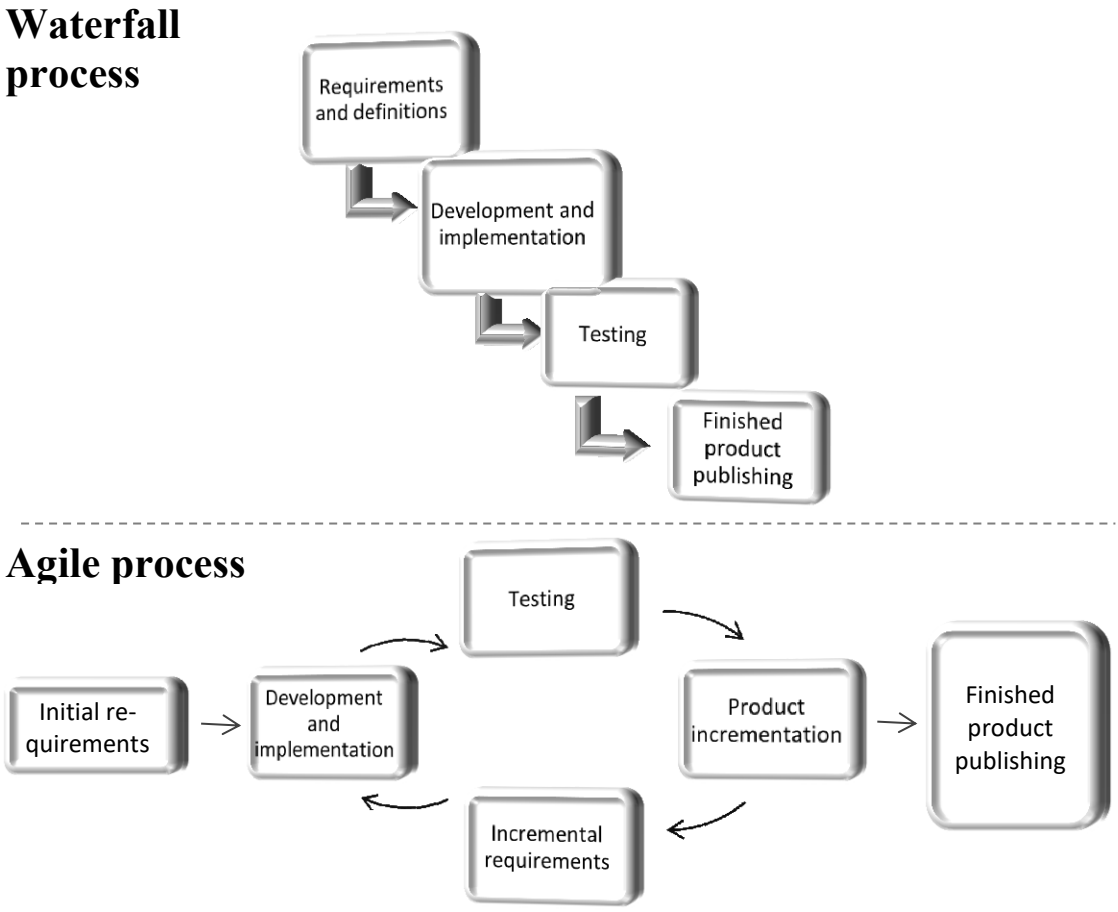

Figure 36. Waterfall process compared to agile process.

As can be seen in Figure 36 both the traditional waterfall approach and the agile process start with focus on the requirements and definitions of the development. The difference is that in the waterfall process these requirements are seen as stable with very limited chances to change in later process stages. The agile process features iterative development cycles where the product is incremented in shorter intervals, which promotes for changes in requirements and features.

When developing safety-critical software the testing stage calls for exhaustive testing, which can result in heavy workload for the short iterative development cycles (Heeager & Nielsen 2018.) Designing automated tests can help cope with these challenges.

The new testing tools are initially going to be used in development environments where agile methods are present. This led to a development approach where the testing tools are designed to provide good compatibility for agile environments. The testing tools are developed as customizable components which can support adaptability to changing requirements of testing. Automating tests can be supported by the testing tools with automated features for e.g. automated simulation of applications.

During the development process the testing tools are built with both general and specific testing functionality, taking example from the SATEEN environment used for MicroSCADA Pro product testing. The general functionality such as test management and logging is developed with compatibility for the various test situations.

The operational test situations are the functionality specific for the developed tools. To simulate the operational situations the testing tools will need functionality to run test cases which can be customized according to the situation and the system to be tested. The test cases can be accessed from libraries located in the file system of the test machine. Test case editing is included in tool functionality.

Figure 37 presents the testing tools in context to test activities and testing functionality. The timeline shows how the testing effort can be divided to the test setup, manual testing and automated testing with focus on the automated test effort.

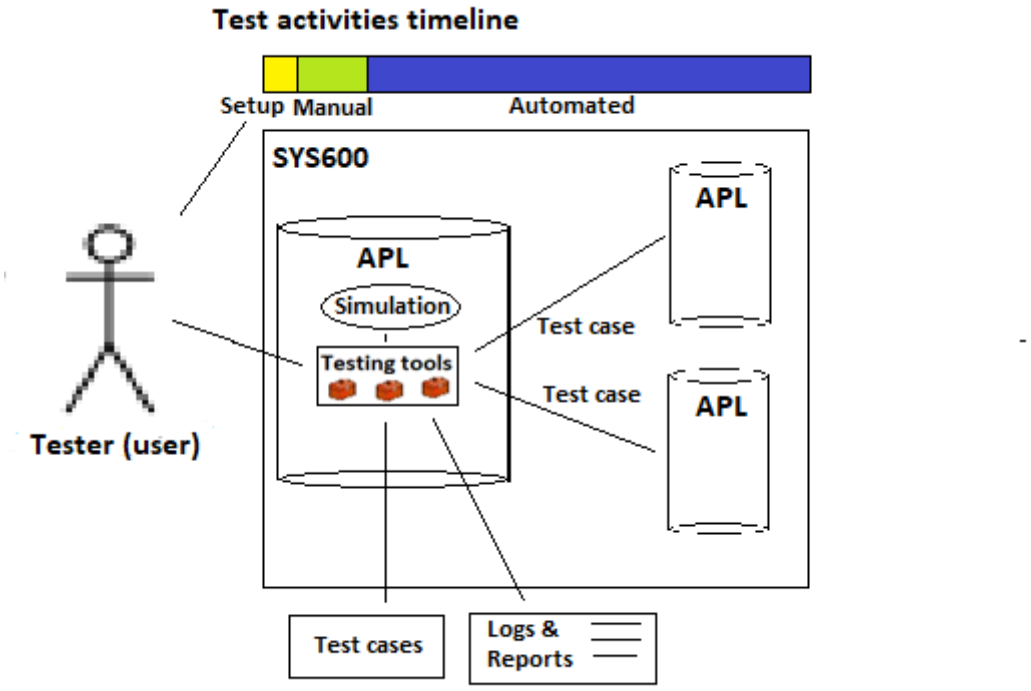

Figure 37. Testing tools in context to test activities and functionality.

The simulation of operational situations in larger scales than individual test cases can provide a high degree of automation and test parts of the system with near full test coverage. Writing test cases and running them may require heavy manual effort, and instead of designing specific test cases more general simulation can be used. This simulation could e.g. scan the application for certain operational devices, and then apply state changes and values to these devices with defined time intervals.

The new testing tools could be used in both product R&D and customer project environments. The testing approaches in these environments differ in principle: the R&D environment requires more generic testing of the products to consider all possibilities how the products are used in practical applications. The project environment focuses more on testing the project specific functionality and configurations. The testing tools are developed with the approach to be generic enough for R&D testing but provide possibilities for testing in project environments also.

6.2 Development progression

The usefulness of the existing testing tools and information acquired prior to development show that feature related changes can happen in development of the new testing tools. Several functionalities and features need to be developed, added as modified or added as readily compatible to the new tools. During the development the usefulness of every feature needs to be evaluated to find out the most potential solution. Therefore the usage of the cyclical agile development method can provide benefits for the development of the new testing tools to cope with the challenges brought by changes in the process.

The development progressed from initial dialog template creation to the cyclical development stages. In these stages the tools were developed, implemented, tested and demoed with newfound incremental requirements. The starting point of the development was far from a finished tool, but many of the features could be developed independently of each other which made possible the demoing and evaluation of the results in short development cycles. When the tool had been developed close to finished, it was applied to operational test environments to test the features in practice. Diagram of the progression is presented in Figure 38.

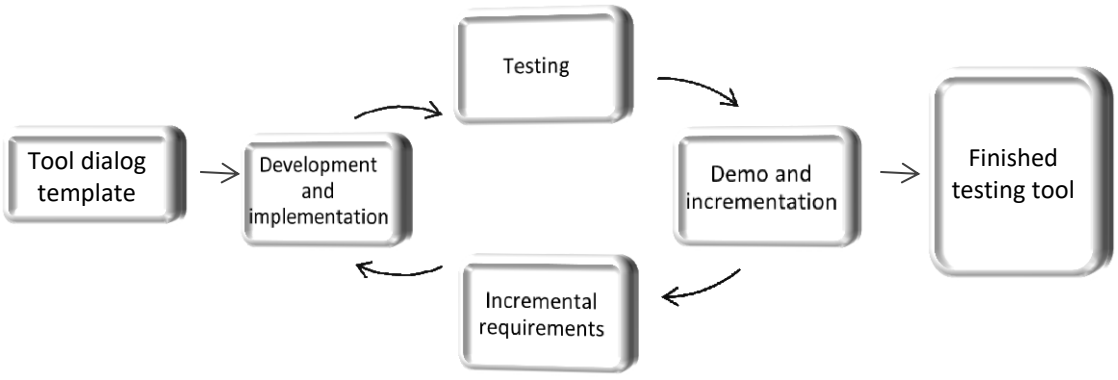

Figure 38. Development progression.

The development started by combining the functionality of the existing testing tools to a single Visual SCIL dialog developed with SYS600 internal development tools. This dialog is presented in Figure 39. The tool contains the testing features that were planned to be implemented and the general and tool specific testing functionality. The dialog features several dependent or independent testing functionalities in tab pages, which makes it possible to develop the new testing tools in single dialog format. The new testing tools can now be also thought of as a single tool with several testing features related to the operational situations.

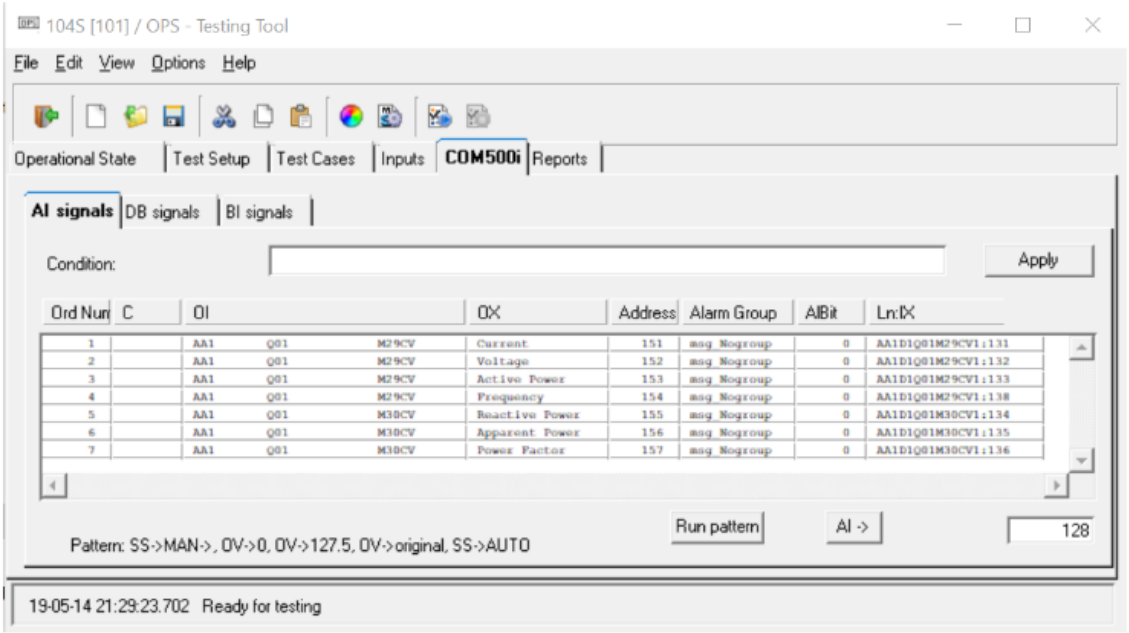

Figure 39. Initial testing tool Visual SCIL dialog.

The features and visual dialog elements were created with the SYS600 Dialog Editor tool. This tool is used to create and modify visual dialogs for the SYS600. Everything from simple text containers and labels to complex hierarchical dialog structures can be created with this tool. Several existing testing tools have been created using the same principle with the dialog editor, which means that the development information is fully accessible with the editor tool. This makes it simple to combine the functionality with the new tools. Visual SCIL dialog editor tool is presented in Figure 40.

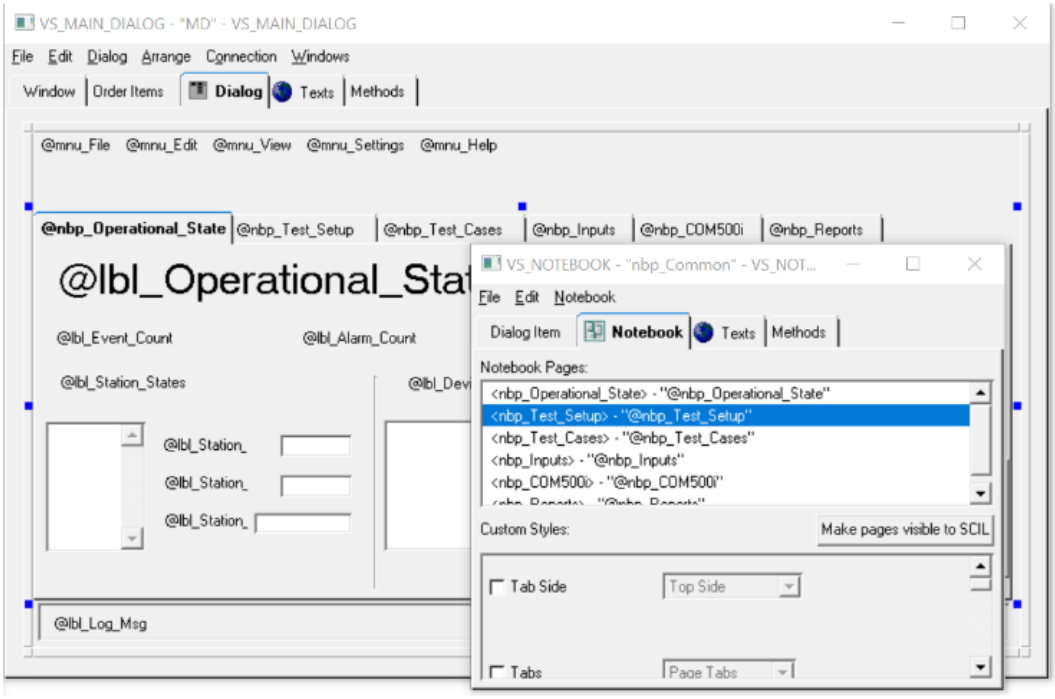

Figure 40. Visual SCIL dialog editor.

Developing the testing tools with the dialog editor requires SCIL programming and management of Visual SCIL objects. Because the applications to be tested can vary in compatibility to certain test cases and the programming work for detailed testing functionality can require high effort, including predesigned test cases within the tool can be impractical. Decisions were made to include core testing functionality within the testing tool and leave the test case specific functionality outside to test case files which contain detailed commands to execute the actual tests. The testing tools will provide a way to run these test case files and track the execution and test results.

Implementing features for the testing tools started by selecting the existing features that could be applied directly to the new tools. Visual SCIL objects can be copied and pasted from one dialog to another. This way objects from the existing testing tools could be selected and added to a new dialog template. SCIL methods related to the visual objects must also be added to keep the functionality consistent.

Objects and methods from the TestRun and COM500i Tester tools were chosen to be included in the new tool. The test execution logics and test setup features of the

TestRun tool were seen as useful for the new testing tools, which led to include them in the new tools. COM500i Tester is a specific tool which can be included in the new tool as it is with some modifications. The tester could be used to test the COM500i signals individually or combined with other features of the new tool.

After existing functionality had been included, new functionality was developed and combined with the existing features. The existing testing tools did not include specific features for test case execution or operational situations. These features needed to be created for the new tool and their development was guided by the development requirements. Tool specific logging and test reporting features were also developed.

The development with Visual SCIL and SCIL involved creating user interface objects, defining their functionality and writing methods for the tool operation logic. The Visual SCIL objects have their own predefined attributes and methods based on the actions and events of the object. For example clicking a button object would execute a SCIL method defined to be run when the button is clicked.

The configured test system was used during the development to test and apply the tool to test environments. Every addition to the tool could be tested in small scale to verify correct functionality. For example the COM500i tester could be first tested as a separate tool in the system and then after adding it to the new testing tool dialog to see if anything changes. The benefit of having the complete system configuration was that the tool features could be tested in various scenarios and the tool would behave expectedly in similar setups in larger scale.

The tool functionality was tested with:

- Sample test setups
- Sample test cases
- Simulation runs

- Combined test cases and automated simulations in same setups
- Customer project replica application

The process also included review meetings with the instructors of this work to hear opinions and improvements during the process. This way features could be implemented incrementally and issues could be resolved earlier in the development stages instead of having a complete tool being reviewed after the development.

Demos were presented during the process to present the state of development and hear opinions about increments to the tool. This stage took place cyclically when development had progressed enough and the testing showed that the tool was ready to be presented in action. Mainly two types of demos were presented: short demos with only couple of features showed and full demos with the complete tool being reviewed with focus on latest increments. The feedback received from the demos was valuable in improving and shaping the tool features.

A typical demo included an overall explanation of the tool with presentation of operational functionality. Sample test cases and simulation runs were used to show the tool in operation in a system under test. Test cases were written to present how a situation could be tested with the tool. For example situations for breaker state changes and loss of communication to a station level device were used as test cases. Automated application simulations were also presented in demos.

The features of the tool could be developed based on dependencies on other features. Some of the features had dependencies on other functionalities, while others could be developed independently of other features. This means that for example the test case based testing functionality could be developed almost independently of the automated simulation functionality. Test setup functionality was developed with high dependency on the test execution functionality, which made it to be continuously changing during the process.

Figure 41 depicts a timeline where features of the tool have been separated by the development effort during the process. The horizontal axis shows the iterations 1-6, which present when a point, such as a demo presentation, was reached and affected the development process. The iterations lasted usually 2-3 weeks. The figure can be thought of more as directional instead of a linear depiction of the development efforts.

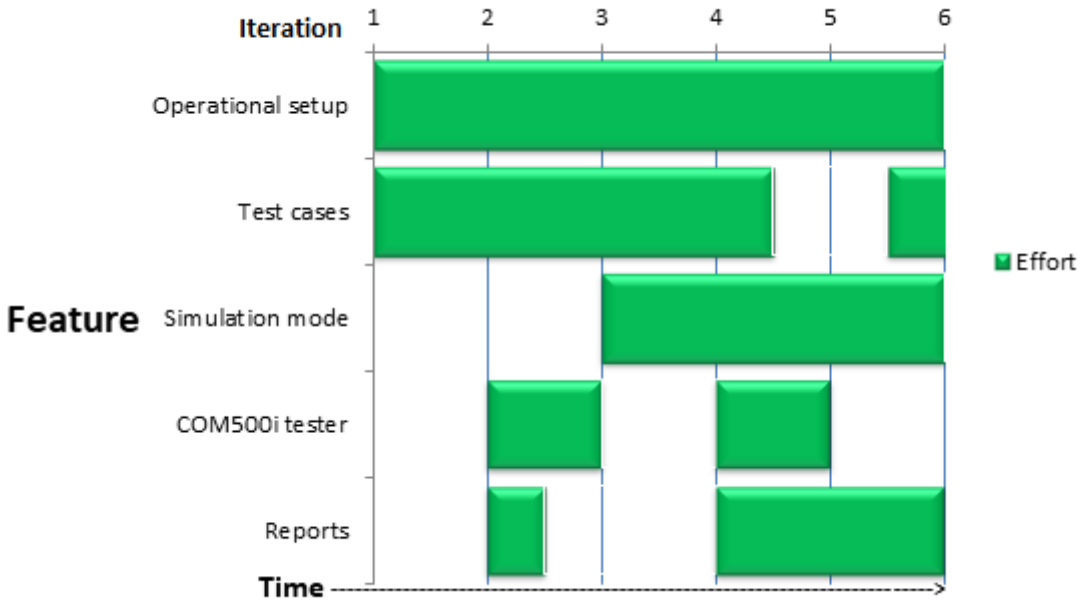

Figure 41. Development effort timeline.

The figure above shows how other features were in development all the time during the process while some of the features could be developed in shorter intervals. The starting point presented as 1 presents the point in development where the initial dialog template had been created and the real development started. Either the test case or simulation features were in development all the time, while the more supporting features of the tool such as reporting and COM500i tool integration were developed more independently. Towards the end of the development all the features were developed for integration of the complete tool functionality.

When the tool had been successfully developed with new features and functionality it was tested and operated in practice. In this stage the tool was applied to an application

based on a real substation automation and SCADA project with several thousands of process points to test and simulate. The tests with this application proved that the new tool was capable of testing large scale customer project applications.

6.3 Tool features

The new testing tool is called the Operational Situations Testing Tool. The tool includes five main features: operational setup, test case mode, simulation mode, COM500i tester and reporting features. The operational setup and reporting handle the supporting functions for testing with the tool, while the other modes provide the testing functionality. All features have been integrated to a Visual SCIL dialog where switching between testing functions is simple and straightforward. The testing tool dialog is presented in Figure 42.

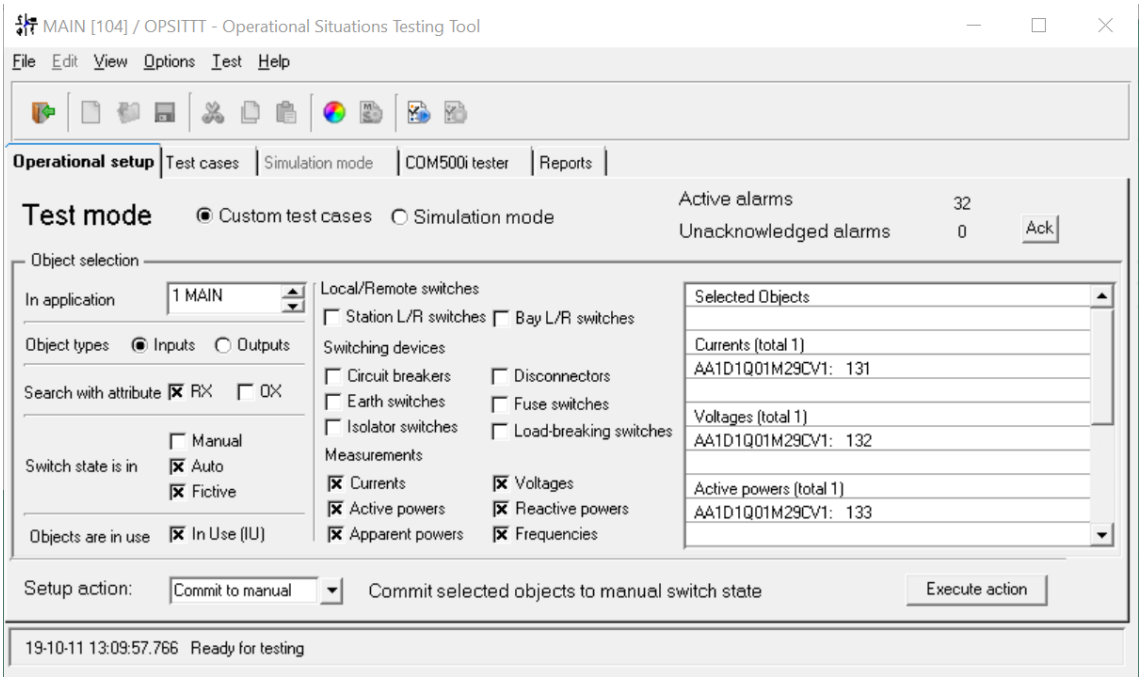

Figure 42. Testing tool dialog with Operational setup.

The operational setup feature contains the test mode selection and setup functions. Process objects in system applications can be searched based on selections such as switch

state and object type. The selections are then searched in the process database and displayed on the selection list, where the objects' values and switch state are also presented. After selection the setup functions can be used to set up the objects with initial values and prepare them for the tests. The available application objects value states can also be saved and rolled back to original state with the setup functions. A total of 14 object types are supported for automated search of databases, including local/remote switches, switching devices and measurements. The operational setup tab is the tab visible in Figure 42.

Test case mode is the first main testing feature of the tool. This mode is based on the selection and execution of test case files written in SCIL. Test case files can be selected from libraries to a test case execution list, where the test cases can be executed in any application in the system. Execution can include repeat cycles or an interval to run the test cases cyclically. The test cases can be edited within the tool with the test case editor. This makes it possible to modify the test case files quickly according to the requirements or results found during tests with the tool. The test case mode is shown in Figure 43.

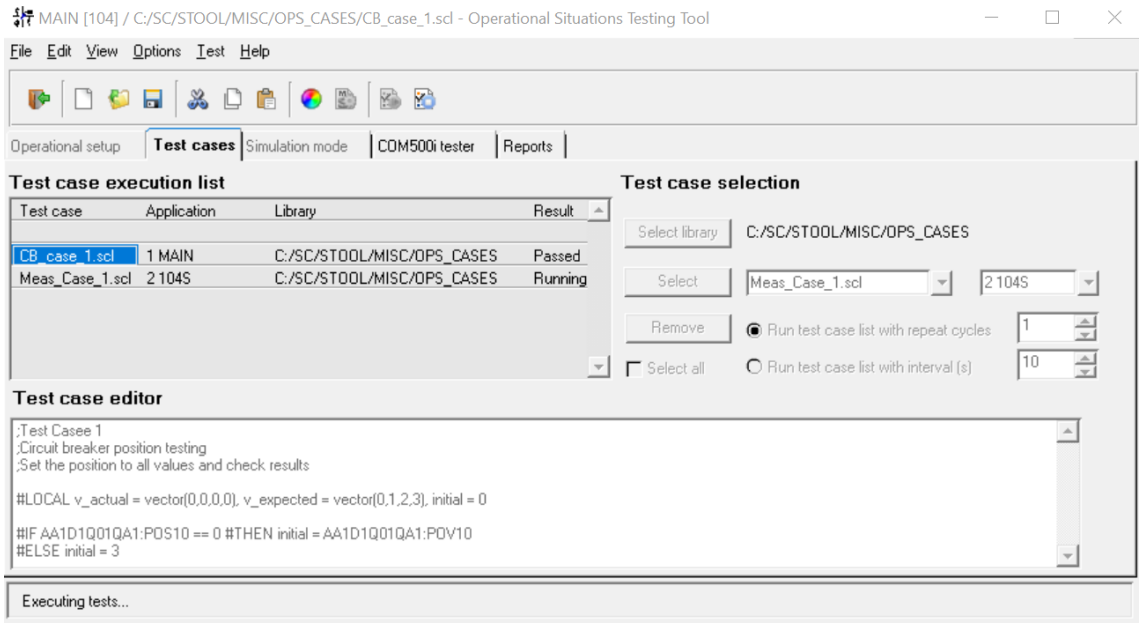

Figure 43. Test case mode.

The second main testing feature is the simulation mode. With this mode large scale automated tests can be run for any application in the system. The tests are based on cyclical test execution with simulation runs for selected process objects and parameters. The simulation runs are configured with the selection of the simulated application, direction, duration, and object type specific parameters such as simulation actions and values. The simulation actions, executed with defined time intervals, include the setting of random, static or alternating values to the objects. The same 14 object types that are supported in the setup functions can be chosen to be searched in process databases for simulation runs. The runs are selected to a list where from they are executed in order. Figure 44 shows the simulation mode tab.

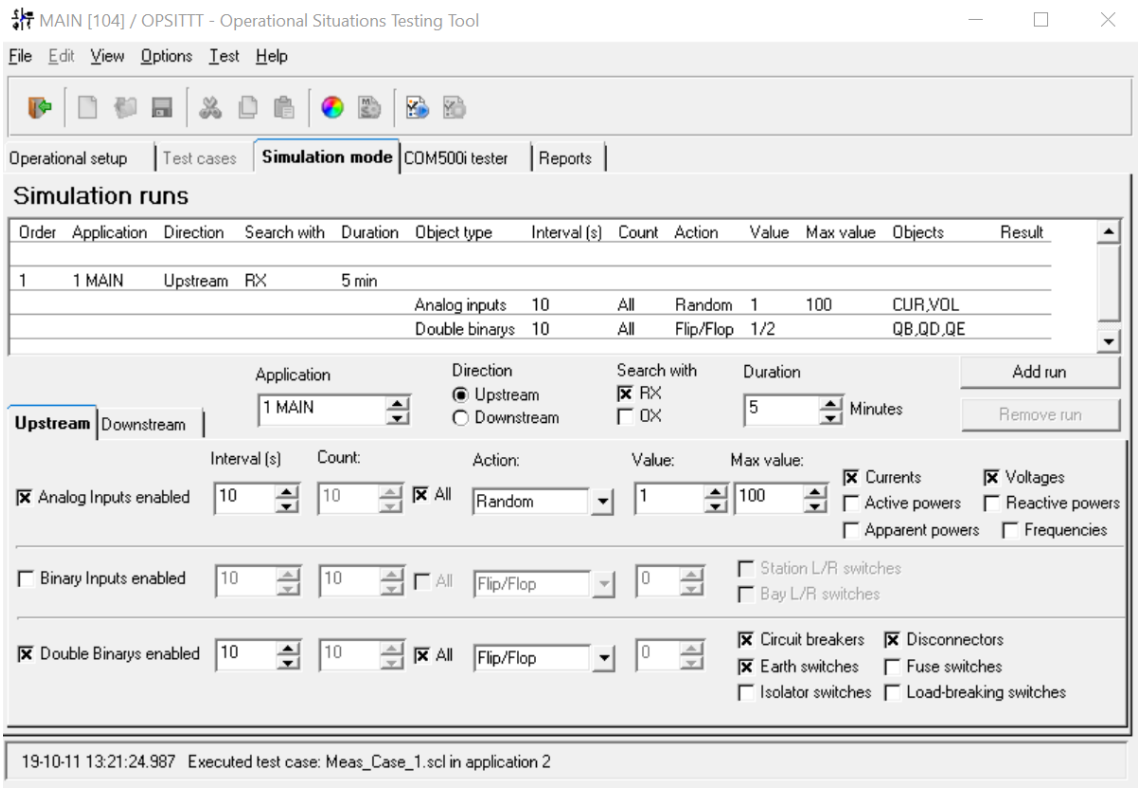

Figure 44. Simulation mode.

COM500i tester is the third main testing feature in the tool. The tester is modified from the original gateway testing tool and integrated to the Operational Situations Testing Tool as a tab in the main dialog. The feature includes upstream signal testing for all analog, binary and double binary signal type objects found in the gateway application. The objects can be tested by setting object value and object status attributes to send signals to the gateway master application. A predefined sequence of values can be run with the pattern action. The tester tab is presented in Figure 45.

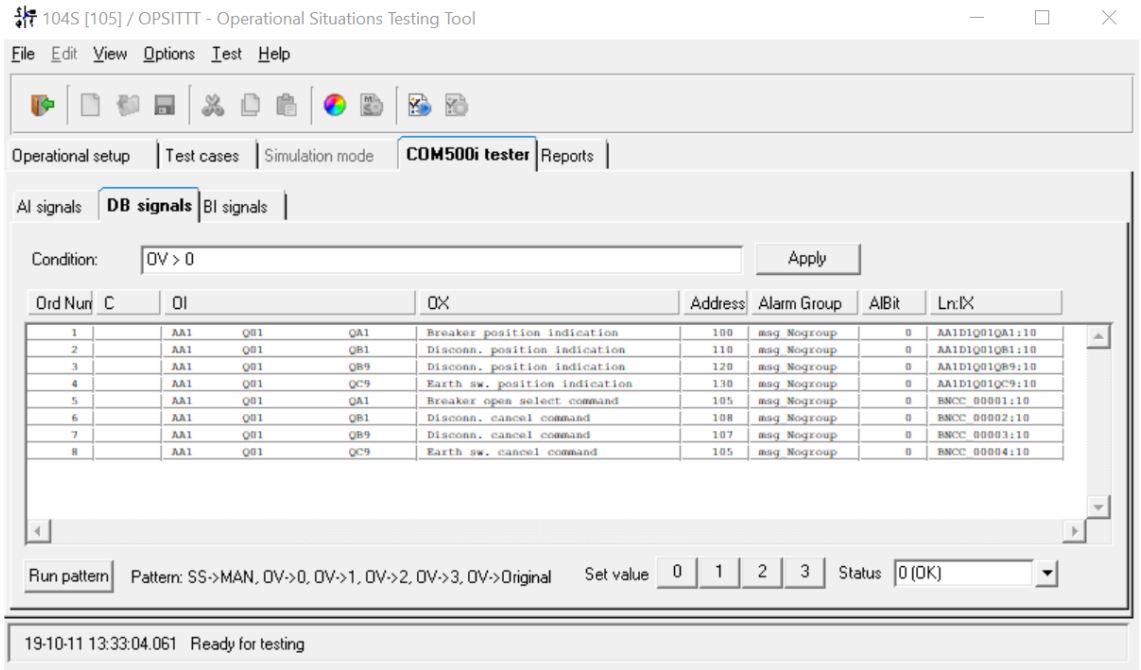

Figure 45. COM500i tester.

The reporting functionality of the tool is the fifth tab of the dialog. This tab contains the operation log and test results log of the tool. The operation log shows information about main operations done with the tool, such as test setup actions. The test result log shows results for all tests run with the test features: test cases, simulation runs and COM500i tests results. The test result log can be saved to a file with the save function of the tool. The reports tab is in Figure 46.

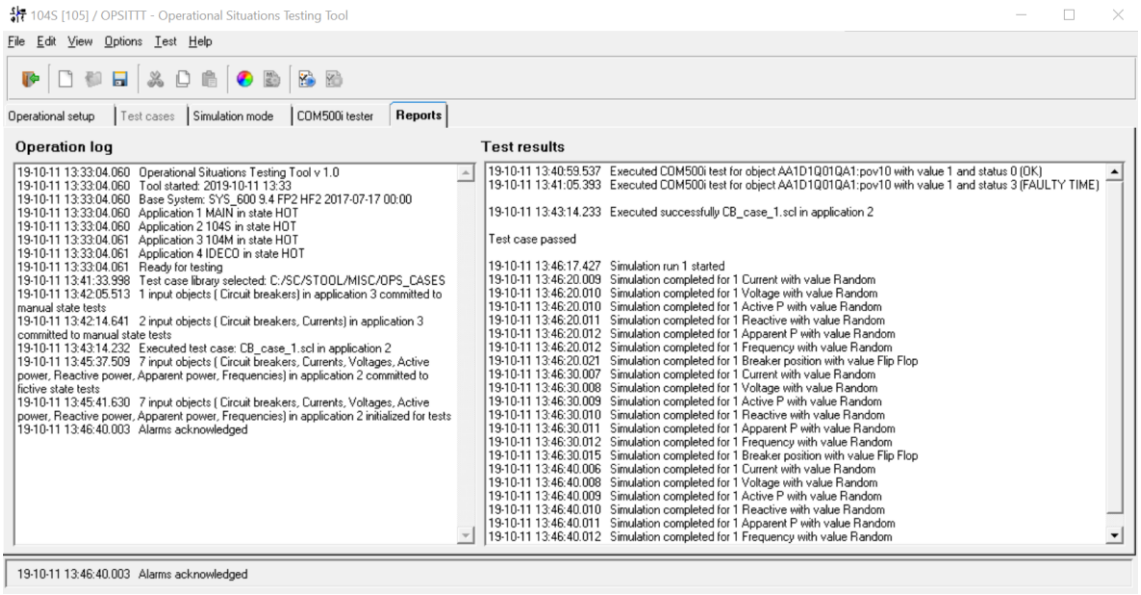

Figure 46. Report logs.

The features development followed the progression presented in previous chapters. Most of the features were completely or partially based on the existing testing tools features. The Operational setup feature has been developed by extending the functionality of the TestRun tool setup features. The Simulation mode is based on TestRun test execution functionality. The COM500i tester is almost completely integrated to the new tool as the same it was as a separate tool. Integrating the existing features included mostly designing user interface solutions and modifying the programs related to these features to suit the new tool functionality.

The general features of the tool included a toolbar containing action buttons and header menus containing the same or similar actions. The action buttons are used to execute e.g. file modifying and editing actions. Test start and stop buttons are used to run the tests in test case and simulation modes. Availability of the features depends on the selected tab and test mode. In general all the features were designed with safeguards related to other features so that unwanted actions could not be performed, e.g. to prevent the execution of setup actions while test cases are running.

6.4 Tool operation

Operating the tool is based on operating it within substation automation and SCADA applications in the SYS600. The tool is prepared by adding it to the Tool Manager function of the application and started as a Visual SCIL tool from the manager.

The tool can be used to test all applications which contain process object types available for the test setup and simulation mode features. With custom test cases even other object types can be tested. Combining the various test features in the system can result in testing with higher test coverage than testing with a single feature. When the focus is on automating tests, the testing modes in the tool can be used to create lists and logics of tests that can be run automatically for long times.

Examples of operating actions with test type are described in Table 6 below.

Table 6. Operating actions with the testing tool.

Action	Test type	Description
Setup objects switch state	Manual	Set selected objects to switch state according to test
Set initial values	Manual	Set initial non-alarming values to selected objects
Test cases	Manual/Auto	Run customized test case files in a test case execution list
Simulation runs	Auto	Run cyclically executed simulation runs
Gateway monitoring direction testing	Manual	Run communication gateway COM500i signal testing with value and status

Example steps of operating the tool with a simulation run downstream from NCC application 3 to process device level simulated in the ITT SA Synthesizer in the test system that was configured for the development are described next.

1. The test system applications, ITT SA Synthesizer tool and Operational Situations Testing Tool are started and prepared for the tests

2. The Operational setup feature of the testing tool is used to set all switching device objects to AUTO switch state to enable process device communication

3. Initial switching device positions are received from the process device simulated in ITT SA Synthesizer

4. The simulation run is configured with the simulation mode feature of the testing tool to send commands for setting alternating values to the switching devices with 10 second intervals

5. Simulation mode is started and the tool will start sending the commands downstream to the process device

6. Test results are presented in the Test results log of the tool, and the process pictures and values in SYS600 and ITT SA Synthesizer are monitored to verify the device state changes

7. When the simulation run has finished the tests can be carried on with e.g. more detailed test case testing or the COM500i tester

This operating example of the tool is shown in Figure 47 from the NCC application side and in Figure 48 from the process side.

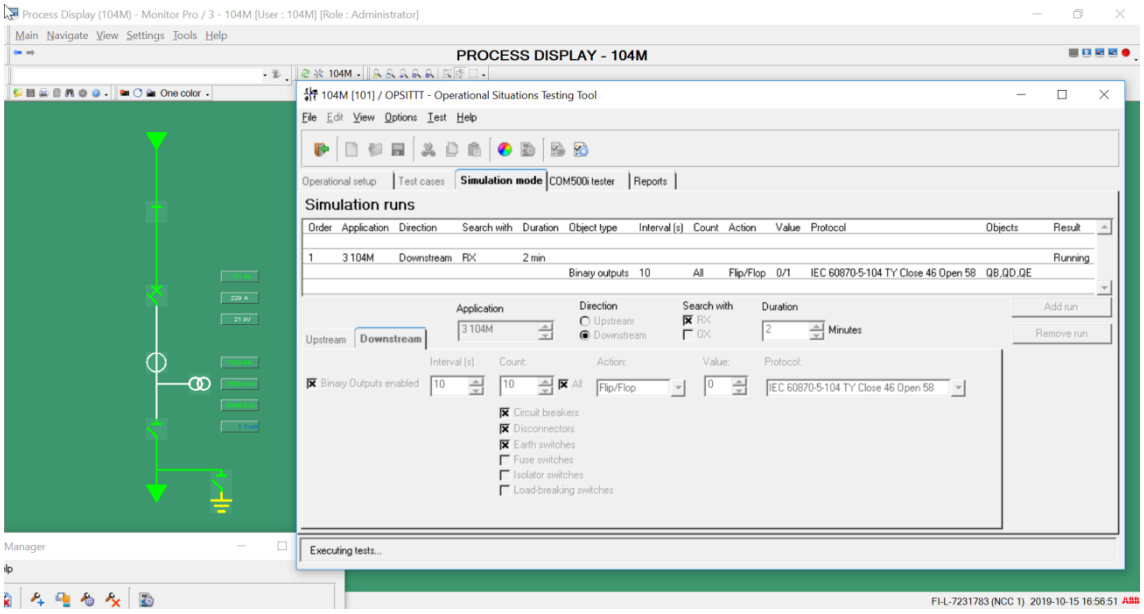

Figure 47. Operating the tool in the test system application 3.

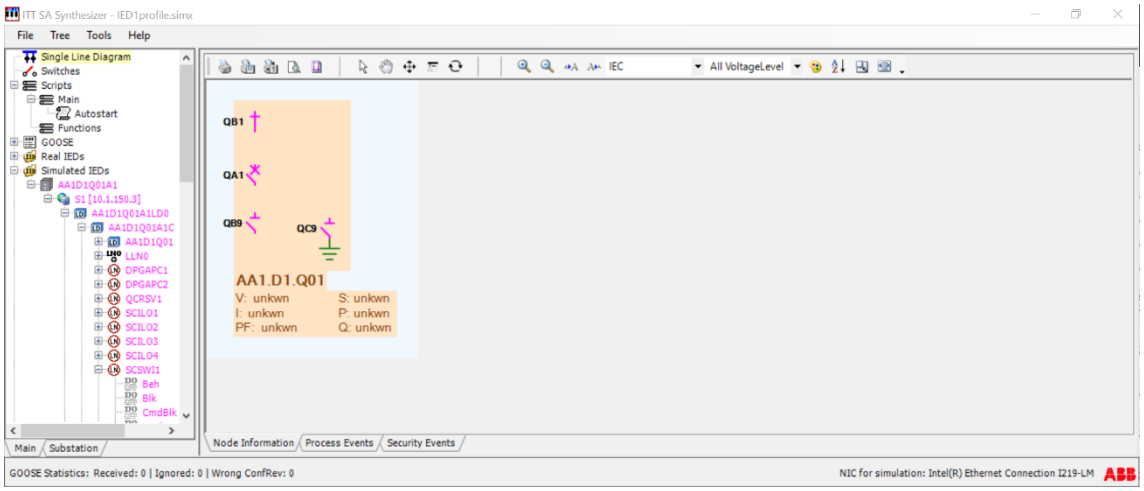

Figure 48. Operating the tool in the test system viewed in the ITT SA Synthesizer.

A user manual for the tool was written to describe the features and operation of the tool in detail. The manual provides the necessary information for the tester to learn and use all features of the tool. Based on the manual the tool could even be customized with SCIL to modify features or add new functionality.

6.5 Future work

The Operational Situations Testing Tool is open for further development in the context of internal development in MicroSCADA Pro products and projects. When the tool was finished it was designed for compatibility in both R&D and project environments. By applying the development methods presented in this thesis the tool can be extended and modified in both environments according to user needs.

Many of the features of the tool are designed to be open for new additions and extensions. Setup and simulation actions can be added with little effort to the existing action lists, and the main testing features could be customized to suit the needs of specific applications. For example new communication protocols could be supported by the simulation runs. One main feature in automating tests is the search for object types in the application process databases, and this feature can be extended with the addition of new object types and search methods to the present supported 14 types.

The programming of the tool was made to support future development by adding comments and solution information in the programs. This way anyone familiar with SCIL development can modify or extend the code of the tool to develop the tool further.

When extending the tool the usability and user experience may change as a result of adding extensively new features. The limits of development can be found when adding new functionality would not serve the original purposes of the tool anymore, making the testing features too general or complex to be used for efficient testing. The testing of operational situations in substation automation and SCADA systems is a complex and wide task in itself, and handling all testing related to this with a single tool is a very challenging approach. The future development of the tool should consider the changes in the whole testing environment it is being used in, and the tool should be kept up to date with the development of this environment.

7 CONCLUSIONS

The testing of substation automation and SCADA systems was investigated and a new testing tool was developed to test operational situations in ABB's MicroSCADA Pro testing environments. The tool can be used to set up and run both manual and automated software tests in hierarchical systems with various power grid components and objects. Tests run with the tool can provide information about systems behavior in operational situations in the R&D and customer project environments.

Information was collected from a wide variety of internal and external sources to define the requirements and features for the new tool. Internal sources such as expert interviews provided insight how various testing processes were in use in similar environments where the new testing tool will be implemented. External sources like literature reviews gave the development general guidelines and strengthened the theoretical background of the work in this thesis.

Features of the tool were successfully developed by including functionality from existing testing tools with the newly developed features. Three main testing features were developed: test case based testing, simulation run testing and communication gateway testing. Supporting setup action and test result reporting features were also included in the new tool. Feedback from demo presentations guided the development and made possible the applying of cyclical agile development practices. The scale and complexity of operational situations in the substation automation and SCADA systems left many possibilities for future work and extensions for the tool.

The thesis work was supported by prior work in MicroSCADA Pro R&D testing and provided interdisciplinary challenges in electrical engineering, information and communication engineering, and software engineering. Understanding of the products and processes in the development environment supported the development as well as the strong background information collected during the work. The way from the beginning of the work until the end results was not totally straightforward, but learning and support provided during the work made the development succeed.

8 REFERENCES

ABB. (2015). *Protection and Control System Utilization of NCIT & Process Bus* [online]. Accessed 16.4.2019 at: [https://new.abb.com/substation-automation/systems/whitepapers/protection-and-control-system-utilization-of-ncit-process-bus]

ABB. (2016a). *SYS600 System Configuration manual*.

ABB. (2016b). *SYS600 Application design manual*.

ABB. (2016c). *SYS600 Operation manual*.

ABB. (2016d). *SYS600 Application objects manual*.

ABB. (2016e). *Programming Language SCIL manual*.

ABB. (2016f). *Communication Gateway COM500i manual*.

ABB. (2016g). *SYS600 IEC 61850 System Design manual*.

ABB. (2016h). *SYS600 IEC 60870-5-101 Slave Protocol manual*.

ABB. (2016i). *SYS600 IEC 60870-5-104 Slave Protocol manual*.

ABB. (2017). *Fast Forward, Newsletter for engineers, managers and technologists in the electrical power transmission and distribution sector*. *Focus on Power Grids*. 2/17 [online]. 40 p. Accessed 13.4.2019 at: [https://new.abb.com/docs/librariesprovider53/magazines-downloads/abb-ffwd-2-17.pdf?sfvrsn=4017af13_2]

ABB. (2019a). *High Voltage Products* [online]. Accessed 31.3.2019 at: [https://new.abb.com/high-voltage]

ABB. (2019b). *Transformers* [online]. Accessed 31.3.2019 at: [https://new.abb.com/products/transformers/]

ABB. (2019c). *RTU 540 product line* [online]. Accessed 28.8.2019 at: [https://new.abb.com/substation-automation/products/remote-terminal-units/rtu540]

ABB. (2018). *We are bridging the gap. Enabling the ABB digital substation* [online]. Accessed 26.3.2019 at: [https://new.abb.com/substation-automation/systems/digital-substation/bridging-the-gap]

Bayliss, C. & B. Hardy (2011). *Transmission and Distribution Electrical Engineering* [online]. 4th Edition. Elsevier Science & Technology. [accessed 16.4.2019].

Bonsanque, M., Broek, R., Chaudron, M., & Merode, H. (2014). Integrating Testing into Agile Software Development Processes. *Proceedings of the International Conference on Model-Driven Engineering and Software Development*. p. 561–569.

Elovaara, J. & L. Haara. (2011). *Sähköverkot II. Verkon suunnittelu, järjestelmät ja laitteet*. Helsinki: Otatieto Helsinki University Press 2011.

Esala, Antti (2019). Project engineer, ABB. Interview, Vaasa 5.2.2019.

Giasis, Alexandros (2016). *Evaluation of the IEC61850 Communication Solutions*. University of Vaasa. Department of Computer Science. Master's thesis.

Heeager, L. T. & P.A. Nielsen (2018). A conceptual model of agile software development in a safety-critical context: A systematic literature review. *Information and Software Technology* Vol. 103, p. 22–39.

Hiltunen, H. (2016). *Optimal Stand-by Supply during Substation Interruption*. Vaasa University of Applied Sciences. Electrical Engineering. Bachelor's thesis.

Homés, B. (2013). *Fundamentals of software testing* [online]. John Wiley & Sons Inc. [accessed 23.2.2019].

Jamil, N., Daud, M. & Patel, A. (2019). A review of security assessment methodologies in industrial control systems. *Information and Computer Security*. Vol. 27(1), p. 47–61.

Jinesh, CJ (2019). Test engineer, ABB. Interview with Skype Vaasa-Bangalore, 30.1.2019.

Laine, Paavo (2019). Project engineer, ABB. Interview with Skype Vaasa-Helsinki, 31.1.2019.

Lehtosaari, M. (2011). *Integration Testing of Protection Relays*. Novia University of Applied Sciences. Electrical Engineering. Bachelor's thesis.

Madrigal, M. & R. Uluski. (2014). *Practical Guidance for Defining a Smart Grid Modernization Strategy : The Case of Distribution* [online]. World Bank Publications. [accessed 28.8.2019].

Mantere, T. (2003). *Automatic Software Testing by Genetic Algorithms*. University of Vaasa. Department of Computer Science. Doctoral thesis.

Martikainen, E. (2017). *Electrical Distribution Network Asset Management from the Aspect of Authority Regulation*. Tampere University of Technology. Faculty of Computing and Electrical Engineering. Master's thesis.

Myers, G. J., Sandler, C. & Badgett, T. (2011). *The Art of Software Testing*. 3rd edition [online]. John Wiley and Sons Inc. [accessed 23.2.2019].

Pirhonen, Jani (2018). Project engineer, ABB. Interview, Vaasa 12.12.2018.

Robinson, B. (2008). *VI2P Test Strategy Presentation*. ABB Corporate research.

Warne, D.F. (2005). *Newnes Electrical Power Engineer's Handbook*. 2nd edition [online]. Elsevier Science & Technology. 456 p. [accessed 28.8.2019].