VAASAN AMMATTIKORKEAKOULU
VASA YRKESHÖGSKOLA
UNIVERSITY OF APPLIED SCIENCES

Taneli Sillanpää

PREDICTIVE MAINTENANCE
IN MICROSCADA APPLICATION

Technology and Communication

2013

PREFACE

First of all I would like to thank Håkan Hultholm, Project Manager and my mentor from ABB, whose advices and support were great help during making of this thesis. I would also like to thank Harri Paulasaari, Engineering Manager who gave me this thesis and Jari Koski, my supervisor from Vaasa University of Applied Sciences. Also thanks to the MicroSCADA product support and other employees at ABB who gave help and support for this thesis.

Vaasa 11.4.2013

Taneli Sillanpää

VAASAN AMMATTIKORKEAKOULU
Sähkötekniikan koulutusohjelma

TIIVISTELMÄ

Tekijä	Taneli Sillanpää
Opinnäytetyön nimi	Ennakoiva ylläpitotoiminta MicroSCADA-ohjelmassa
Vuosi	2013
Kieli	englanti
Sivumäärä	54 + 3 liitettä
Ohjaaja	Jari Koski

Tämä opinnäytetyö on tehty ABB Oy Substation Automation Systems -yksikköön. Työssä tutkitaan miten ennakoivaa ylläpitoa voidaan soveltaa MicroSCADA-järjestelmän ylläpidossa ja rakennetaan sovellusesimerkki automaattista raportointia varten.

Työn teoriaosuudessa verrataan erilaisia ylläpitomenetelmiä ja kerrotaan miten ennakoiva ylläpitotoiminta eroaa tavallisista ylläpitomenetelmistä. Työssä selvitetään haastatteluihin pohjautuen, minkälainen informaatio Micro-SCADA-järjestelmästä on hyödyllistä ylläpidolle. Tiedon automaattista keräämistä varten rakennetaan sovellusesimerkki ja selvitetään perinpohjaisesti kaikki, mitä sovelluksen luoma raportointitiedosto sisältää. Viimeisessä osiossa ohjeistetaan, miten sovellusesimerkki asennetaan ja miten raportin lähetys saadaan hoidettua automaattisesti.

Opinnäytetyön tuloksena syntyi sovellusesimerkki, joka kerää informaatiota MicroSCADA-järjestelmästä ja luo tästä informaatiosta zip-muotoisen raporttitiedoston. Työssä selvitettiin myös, miten tämä tiedosto pystytään lähettämään automaattisesti sähköpostin välityksellä ylläpidosta vastaaville henkilöille.

Avainsanat MicroSCADA, ennakoiva ylläpito

VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES
Sähkötekniikan koulutusohjelma

ABSTRACT

Author	Taneli Sillanpää
Title	Predictive Maintenance in MicroSCADA Application
Year	2013
Language	English
Pages	54 + 3 Appendices
Name of Supervisor	Jari Koski

This thesis is made for ABB Oy Substations Automation Systems. In the thesis it is researched how predictive maintenance can be applied in the maintenance work of the MicroSCADA system. An application example for automatic reporting was also built for this thesis.

In the theory part of the thesis, different maintenance methods are compared and it is told how predictive maintenance differs from ordinary maintenance methods. Based on the interviews, it was clarified what kind of information from the MicroSCADA system is useful for the maintenance support. An application example was built and all information included in the report file that the application creates is explained thoroughly. In the last chapter it is instructed how the application example is installed and how the system should be configured in order to send the report automatically.

As an outcome of this thesis, an application example was created, which collects information from the MicroSCADA system. This information is included in a report file in zip format. It was also solved, how this file can be sent automatically via e-mail to the people responsible for the maintenance.

Avainsanat MicroSCADA, Predictive Maintenance

CONTENTS

TERMS AND ABBREVIATIONS

COM 500	Communication gateway
CPU	Central Processing Unit
DoS	Denial of Service attack
GNU GPL	GNU General Public License
GPS	Global Positioning System
HMI	Human Machine Interface
IED	Intelligent Electronic Device, usually a protective relay
PLC	Programmable Logic Controller
ODBC	Open Database Connectivity
OPC	Open Process Control
RTU	Remote Terminal Unit
SCADA	Supervisory Control and Data Acquisition
SCIL	Supervisory Control Implementation Language
SNMP	Simple Network Management Protocol
SMTP	Simple Mail Transfer Protocol
SSS	System Self Supervision
TLS	Transport Layer Security
UNIX	Open source operating system
Zip	File format used for data compression

1 INTRODUCTION

This thesis has been made for ABB Substation Automation Systems.

The maintenance support for MicroSCADA systems is dealing with many kinds of issues occurring in the customers systems. Prediction improves reliability because the oncoming problems could be detected before any major failures happen within the system.

For example an increase in communication failures is typically a sign of oncoming problem. Also an increase in system resources used by the MicroSCADA application and other processes is a warning of problems that are on their way. There is some useful information in MicroSCADA that could be interesting, but some data is only available from the information registered by the operating system.

The objective for this thesis was to build an application example which automatically gathers useful information from MicroSCADA application and the operating system and gathers this information in to a form of a report. This report also has to be sent automatically to the e-mail address of the maintenance support.

This first part of this thesis introduces different maintenance methods and the concept of predictive maintenance. In the next chapters it is told what kind of information is useful for the maintenance support and how this information can be accessed and gathered. A solution for sending the report by e-mail is introduced after that. The last part of this thesis is instructions for the installation and configuration of the components required for the reporting tool to work properly.

2 BACKGROUND

2.1 ABB Substation Automation Systems

ABB is a global leader in power and automation technologies headquartered in Zurich, Switzerland. The Company employs 145 000 people in approximately 100 countries. ABB in its current form was born in 1988 when Swedish ASEA and Swiss BBC Brown Boveri united. Finnish company Oy Strömberg Ab was bought by ASEA in 1987. /1/

Substation Automation Systems is part of the ABB Power Systems division’s Network Management business unit. It designs and engineers advanced substation automation, protection and control solutions.

2.2 MicroSCADA

MicroSCADA is supervision software developed by ABB. The development started in 1981 by Strömberg Oy in Vaasa, Finland. It was originally developed for substation automation but today it is also widely used for supervising and controlling electrical power networks. /16/

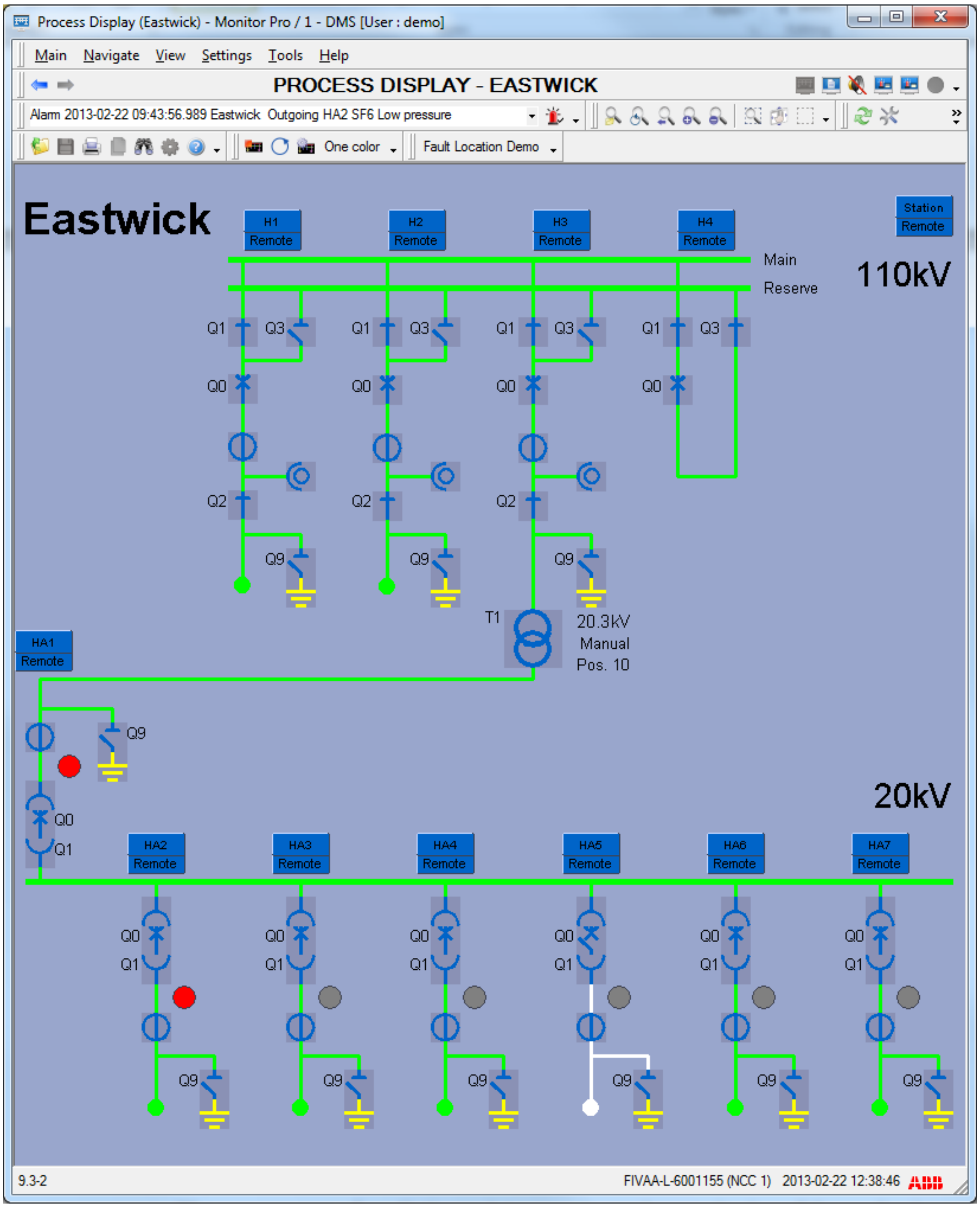
SCADA is an abbreviation for Supervisory Control and Data Acquisition. It is a common type of industrial control system that is used for real-time monitoring and control of system, processes or even specific machinery. MicroSCADA is mainly used for substation automation and network control, but it can also be used to control and monitor non-electrical applications, such as railroads, water and sewage utilities, district heating, or even oil and gas distribution.

MicroSCADA systems usually consist of:

- Workstations with a human-machine interface (HMI) which presents the data and measurements in a graphical user interface to a human operator who monitors and controls the process
- A system server where MicroSCADA is running

- Communication devices like switches, routers and modems and communication servers
- Protection relays (IED) capable of performing various mathematical calculations and communicating with other devices
- Remote terminal units (RTU) that convert the sensor signals to digital data and send that data to the computer systems
- All kinds of peripheral equipment, such as printers and GPS device /3/ /14/

An example of a process window in MicroSCADA can be seen in Figure 1.



Process Display (Eastwick) - Monitor Pro / 1 - DMS [User : demo]

Alarm 2013-02-22 09:43:56.989 Eastwick Outgoing HA2 SF6 Low pressure

9.3-2

FIVAA-L-6001155 (NCC 1) 2013-02-22 12:38:46

Figure 1. Process Window of MicroSCADA

Breakers and disconnectors and their states are shown in the picture. They are named with a letter Q in the beginning. T1 is a power transformer which steps down voltage from 110 kV to 20 kV. Flashing red circles presents active alarms. Busbars are colored green when there is a voltage and white when they are disconnected. Busbar coloring can also be voltage dependent, when different voltage levels are colored differently. Coloring options can be changed and indicators can be self-designed. Event displays and alarm lists can be found from the top right corner of the picture.

3 MAINTENANCE MANAGEMENT METHODS

Maintenance contracts are an important part of engineering projects in ABB Substation Automation Systems. In case of a problem the customer can immediately contact the company and engineers responsible for the maintenance. In the best case scenario the problem can be fixed by using a phone and Remote Desktop Connection but sometimes the problem demands repairing it on the location. By buying maintenance service the customer can shorten and prevent interruptions in their process or in service they are offering. This improves reliability and usability of the system, saves the customer's money and also provides earnings to the deliverer of the system after the commissioning.

There are several maintenance management methods that could be considered. Typically industrial plans use run-to-failure or preventive maintenance.

3.1 Run-to-Failure Management

The idea behind of run-to-failure is very simple. There is no preventive or predictive maintenance done, so if something breaks up, it will be fixed. This is a very old and still widely used method of maintenance. When run-to-failure management is used, there are no maintenance costs until the system fails to operate.

In run-to-failure management, there is actually no maintenance made. It is the most expensive maintenance method because of high spare parts inventory cost, high costs of labor overtime, high downtime of the system and low production availability. When this management method is used, a company must be able to react to all possible failures with spare parts inventories that include all critical components or even spare machines. Another option is to rely on equipment vendors that can provide quick delivery of all spare parts required. The maintenance personnel must also be available and react immediately to all system failures.

Studies indicate that the costs of run-to-failure maintenance are about three times higher, compared to those repairs made with scheduled or preventive maintenance management. When the repairs are scheduled, it minimizes the repair time and labor costs. /11/

3.2 Preventive Maintenance

All preventive maintenance management programs are time-driven. Maintenance work is based on the operation time of the system. The statistical life of a machine can be calculated and it can be illustrated by a mean-time-to-failure curve. An example of the statistical life of a machine-train is shown in Figure 2.

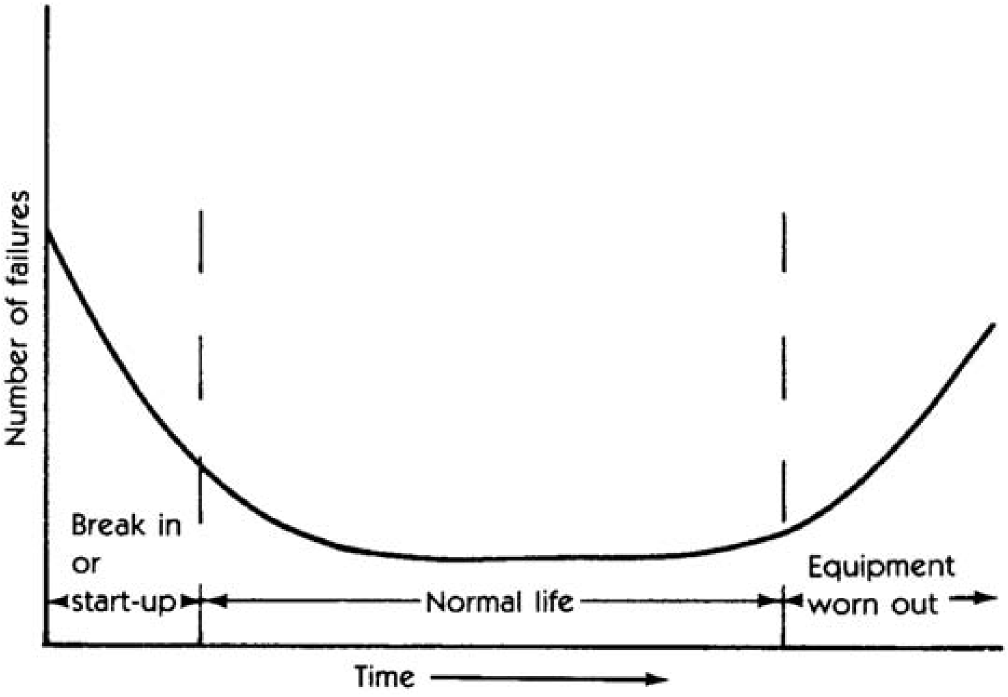


Figure 2. Statistical life of a machine-train /11/

It can be seen from the curve that a new machine has a high probability of failure. This is caused by installation problems during the few weeks after the start of the operation. After these few weeks, the system normalizes and the probability of failure is relatively low. After this normal life, the failure becomes more and more likely. In preventive maintenance, repairs are scheduled based on these statics.

Common for all the preventive maintenance management programs, is the assumption that machines will degrade within a calculated time frame. For example, if a pump normally runs for 20 months before break down, using preventive maintenance, it would be removed from use and repaired after 19 months of operation. The problem with this is that calculations are not always correct. There are many variables that affect the normal operating life.

Using statistics and calculations to schedule maintenance is often causing unnecessary repairs. The pump in the example could have operated for over 20 months without any problems, so the repairs made were basically a waste of money. Or the pump could have failed in 10 months and it must be repaired with costly run-to-failure techniques. /11/

3.3 Predictive Maintenance

In predictive maintenance the condition of in-service system is monitored in order to predict when something is wrong and maintenance work should be performed. The main reason and value of predictive maintenance for the customer is to prevent unexpected failures and interruptions in the process. In some cases advantages include increased lifetime of equipment, increased safety and optimized spare parts handling.

The common use of predictive maintenance is regular monitoring of the mechanical condition, efficiency of the operation and other indicators of the operating condition. The indicators could be the vibration of rotating machinery or an infrared image of electrical switchgear, motors and other electrical equipment. Instead of relying on statistics to schedule repairs and other maintenance activities, like mean-time-to-failure in preventive maintenance, predictive maintenance monitors the real mechanical condition, efficiency and other indicators to define the real mean-time-to-failure.

This data of the actual mechanical condition provide actual data for scheduling maintenance activities. A predictive maintenance program can minimize unscheduled breakdowns and ensure that the condition of repaired equipment is acceptable. The program can also identify problems before they become critical. Most problems can be minimized by early detection and repairing. Major problems can be usually prevented, if the problem is detected early.

When predictive maintenance is used for a computer system, such as MicroSCADA system, the indicators are slightly different but the main principle is the same. All components with moving parts are the first one to break e.g. cooling fans and hard disks. But computer systems have also non-physical indicators, like excessive memory or CPU usage that can be monitored to predict oncoming problems. /11/

By interviewing the people involved in maintenance work, the indicators in the system attributes can be found out that could be used as signs of an oncoming problem in the MicroSCADA system.

4 ACQUIRING DATA FROM MICROSCADA

4.1 Relevant Data

There is some information that could be used to predict oncoming problems in MicroSCADA.

Communication Problems

Communications between different stations and devices are the most important part of the MicroSCADA systems to work properly. Measurements, controlling, indications etc. all need a connection to the station in order to be able to control devices or read data from them in MicroSCADA. When the communication is lost, it may be caused by a power failure in the cabinet where the network switch is located, improperly functioning RTU or interruption in wireless mobile connection.

The system is usually configured so that the lost communication is generating an alarm in the MicroSCADA. If these alarms are occurring frequently, it can be assumed that there is something wrong with the system. MicroSCADA also has a lot of different status information and action counters that can be used as a sign of a problem.

System Resources Used by MicroSCADA

Running MicroSCADA application uses a certain amount of system resources. These resources are the usage of memory, CPU power, handles and threads. The handles represent open instances of basic operating system objects applications interact with, such as files, registry keys and shared memory. The threads are small processes within processes that share resources, such as memory. When MicroSCADA is in the normal idle mode, it only uses small amount of system resources. If there are some actions taking place within the program, the memory and CPU power usage both raise. If the usage of system resources rises without any particular reason, it could be a sign of a problem.

4.2 Gathering the Information

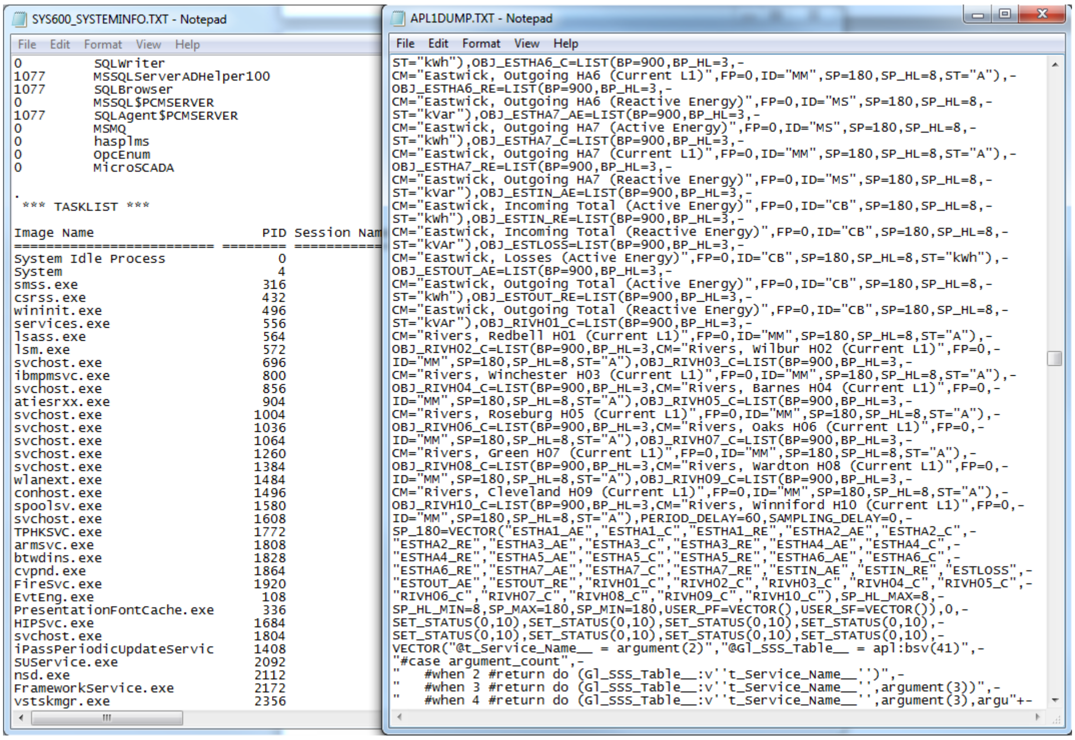
The data in MicroSCADA can normally be viewed only in MicroSCADA itself. But the data can be accessed by using SCIL programming language or by using an interface, like ODBC or OPC to access data in MicroSCADA with another program. SCIL is a programming language especially designed for the application engineering of the SYS 600 supervisory control system. Each MicroSCADA application has a process database for handling process supervision and a report database for calculations, automatic activation and data storage. These databases are composed of application objects. SCIL programming can be used to control the entire system, not only controlling the objects, but also defining the system configuration and communication. With SCIL it is possible to read attributes in the system objects and for example create a text report using these attributes. /10/

The MicroSCADA product support team located in Vaasa has made a program that gathers all the data from MicroSCADA and the operating system as well, in to a compressed zip file. The file is meant to be sent by an operator to the product support team, in case the system or MicroSCADA is having some problems. With the information that the file contains, the product support team can examine what programs are installed in the operating system, memory usage of the programs running at the time and MicroSCADA application configurations. In fact the data gathering program is included in all new MicroSCADA Pro versions. It is located in MicroSCADA Control Panel under the admin menu. There you can push the Support-button which opens a wizard that creates the system info log file.

In order to be able to run the System Info Logging tool automatically, some changes have to be made to it. The standard program starts a wizard that requires a few mouse clicks from the user and it is not possible to add any additional information to the zip file. The program is an executable file, which cannot be modified without the source code. The developers of the System Info Logging tool were very helpful and made a custom version of the program for this thesis. This new executable file does not require any approvals or additional mouse click from the user. It can be placed in to /prog/exec/ folder of MicroSCADA.

4.3 Information Included

The zip file contains a number of simple text files that each shows different information of the system. The information in some of the files is written in vector form and it is meant to be read in MicroSCADA. Some information, such as the operating system and running tasks are in a clear text form. An example of the information is shown in Figure 3.


SYS600_SYSTEMINFO.TXT - Notepad
File Edit Format View Help
APL1DUMP.TXT - Notepad
File Edit Format View Help

```
0          SQLWriter
1077       MSSQLServerADHelper100
1077       SQLBrowser
0          MSSQL$PCMSERVER
1077       SQLAgent$PCMSERVER
0          MSMQ
0          hasplms
0          OpcEnum
0          MicroSCADA

.
 *** TASKLIST ***

Image Name                     PID Session Nam
========================= ======== ==========
System Idle Process              0
System                           4
smss.exe                       316
csrss.exe                      432
wininit.exe                    496
services.exe                   556
lsass.exe                      564
lsm.exe                        572
svchost.exe                    696
ibmpmsvc.exe                   800
svchost.exe                    856
atiesrxx.exe                   904
svchost.exe                   1004
svchost.exe                   1036
svchost.exe                   1064
svchost.exe                   1260
svchost.exe                   1384
wlanext.exe                   1484
conhost.exe                   1496
spoolsv.exe                   1580
svchost.exe                   1608
TPHKSVC.exe                   1772
armsvc.exe                    1808
btwdins.exe                   1828
cvpnd.exe                     1864
FireSvc.exe                   1920
EvtEng.exe                     108
PresentationFontCache.exe      336
HIPSvc.exe                    1684
svchost.exe                   1804
iPassPeriodicUpdateServic     1408
SUService.exe                 2092
nsd.exe                       2112
FrameworkService.exe          2172
vstskmgr.exe                  2356
```

```
ST="kWh"),OBJ_ESTHA6_C=LIST(BP=900,BP_HL=3,-
CM="Eastwick, Outgoing HA6 (Current L1)",FP=0,ID="MM",SP=180,SP_HL=8,ST="A"),-
OBJ_ESTHA6_RE=LIST(BP=900,BP_HL=3,-
CM="Eastwick, Outgoing HA6 (Reactive Energy)",FP=0,ID="MS",SP=180,SP_HL=8,-
ST="kVar"),OBJ_ESTHA7_AE=LIST(BP=900,BP_HL=3,-
CM="Eastwick, Outgoing HA7 (Active Energy)",FP=0,ID="MS",SP=180,SP_HL=8,-
ST="kWh"),OBJ_ESTHA7_C=LIST(BP=900,BP_HL=3,-
CM="Eastwick, Outgoing HA7 (Current L1)",FP=0,ID="MM",SP=180,SP_HL=8,ST="A"),-
OBJ_ESTHA7_RE=LIST(BP=900,BP_HL=3,-
CM="Eastwick, Outgoing HA7 (Reactive Energy)",FP=0,ID="MS",SP=180,SP_HL=8,-
ST="kVar"),OBJ_ESTIN_AE=LIST(BP=900,BP_HL=3,-
CM="Eastwick, Incoming Total (Active Energy)",FP=0,ID="CB",SP=180,SP_HL=8,-
ST="kWh"),OBJ_ESTIN_RE=LIST(BP=900,BP_HL=3,-
CM="Eastwick, Incoming Total (Reactive Energy)",FP=0,ID="CB",SP=180,SP_HL=8,-
ST="kVAr"),OBJ_ESTLOSS=LIST(BP=900,BP_HL=3,-
CM="Eastwick, Losses (Active Energy)",FP=0,ID="CB",SP=180,SP_HL=8,ST="kWh"),-
OBJ_ESTOUT_AE=LIST(BP=900,BP_HL=3,-
CM="Eastwick, Outgoing Total (Active Energy)",FP=0,ID="CB",SP=180,SP_HL=8,-
ST="kWh"),OBJ_ESTOUT_RE=LIST(BP=900,BP_HL=3,-
CM="Eastwick, Outgoing Total (Reactive Energy)",FP=0,ID="CB",SP=180,SP_HL=8,-
ST="kVAr"),OBJ_RIVH01_C=LIST(BP=900,BP_HL=3,-
CM="Rivers, Redbell H01 (Current L1)",FP=0,ID="MM",SP=180,SP_HL=8,ST="A"),-
OBJ_RIVH02_C=LIST(BP=900,BP_HL=3,CM="Rivers, Wilbur H02 (Current L1)",FP=0,-
ID="MM",SP=180,SP_HL=8,ST="A"),OBJ_RIVH03_C=LIST(BP=900,BP_HL=3,-
CM="Rivers, Winchester H03 (Current L1)",FP=0,ID="MM",SP=180,SP_HL=8,ST="A"),-
OBJ_RIVH04_C=LIST(BP=900,BP_HL=3,CM="Rivers, Barnes H04 (Current L1)",FP=0,-
ID="MM",SP=180,SP_HL=8,ST="A"),OBJ_RIVH05_C=LIST(BP=900,BP_HL=3,-
CM="Rivers, Roseburg H05 (Current L1)",FP=0,ID="MM",SP=180,SP_HL=8,ST="A"),-
OBJ_RIVH06_C=LIST(BP=900,BP_HL=3,CM="Rivers, Oaks H06 (Current L1)",FP=0,-
ID="MM",SP=180,SP_HL=8,ST="A"),OBJ_RIVH07_C=LIST(BP=900,BP_HL=3,-
CM="Rivers, Green H07 (Current L1)",FP=0,ID="MM",SP=180,SP_HL=8,ST="A"),-
OBJ_RIVH08_C=LIST(BP=900,BP_HL=3,CM="Rivers, Wardton H08 (Current L1)",FP=0,-
ID="MM",SP=180,SP_HL=8,ST="A"),OBJ_RIVH09_C=LIST(BP=900,BP_HL=3,-
CM="Rivers, Cleveland H09 (Current L1)",FP=0,ID="MM",SP=180,SP_HL=8,ST="A"),-
OBJ_RIVH10_C=LIST(BP=900,BP_HL=3,CM="Rivers, Winniford H10 (Current L1)",FP=0,-
ID="MM",SP=180,SP_HL=8,ST="A"),PERIOD_DELAY=60,SAMPLING_DELAY=0,-
SP_180=VECTOR("ESTHA1_AE","ESTHA1_C","ESTHA1_RE","ESTHA2_AE","ESTHA2_C",-
"ESTHA2_RE","ESTHA3_AE","ESTHA3_C","ESTHA3_RE","ESTHA4_AE","ESTHA4_C",-
"ESTHA4_RE","ESTHA5_AE","ESTHA5_C","ESTHA5_RE","ESTHA6_AE","ESTHA6_C",-
"ESTHA6_RE","ESTHA7_AE","ESTHA7_C","ESTHA7_RE","ESTIN_AE","ESTIN_RE","ESTLOSS",-
"ESTOUT_AE","ESTOUT_RE","RIVH01_C","RIVH02_C","RIVH03_C","RIVH04_C","RIVH05_C",-
"RIVH06_C","RIVH07_C","RIVH08_C","RIVH09_C","RIVH10_C"),SP_HL_MAX=8,-
SP_HL_MIN=8,SP_MAX=180,SP_MIN=180,USER_PF=VECTOR(),USER_SF=VECTOR()),0,-
SET_STATUS(0,10),SET_STATUS(0,10),SET_STATUS(0,10),SET_STATUS(0,10),-
SET_STATUS(0,10),SET_STATUS(0,10),SET_STATUS(0,10),SET_STATUS(0,10),-
VECTOR("@t_Service_Name__ = argument(2)","@Gl_SSS_Table__ = apl:bsv(41)",-
"#case argument_count",-
"    #when 2 #return do (Gl_SSS_Table__:v''t_Service_Name__'')",-
"    #when 3 #return do (Gl_SSS_Table__:v''t_Service_Name__'',argument(3))",-
"    #when 4 #return do (Gl_SSS_Table__:v''t_Service_Name__'',argument(3),argu"+-
```

Figure 3. An example of the system information

In the example on the left there are all the running tasks of the operating system at moment and on the right there is information about the current application running on MicroSCADA.

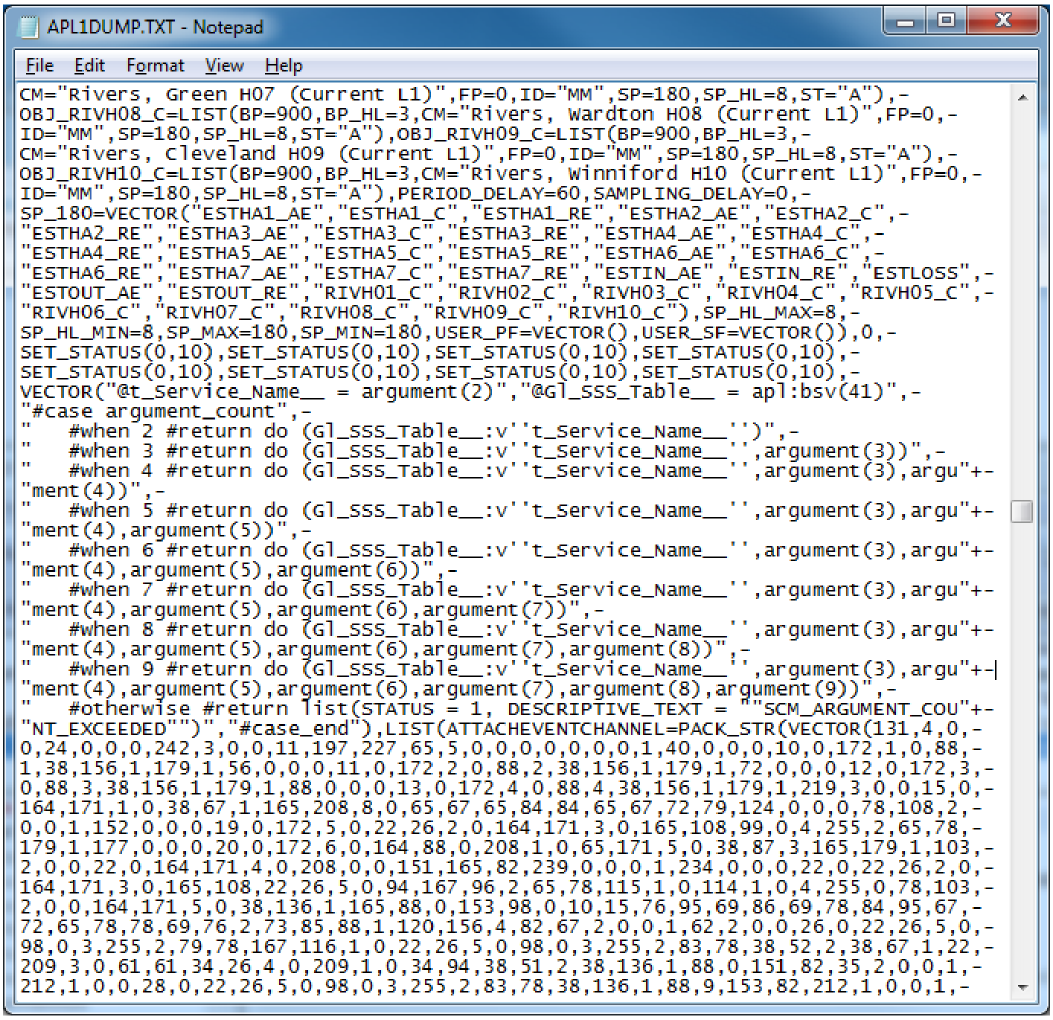
Here is a summary of all the information about MicroSCADA that the zip file contains.

Scdirlist.txt

This text file tells information about the installation folder of MicroSCADA. It lists every file in every folder and subfolder that there is in the installation folder. The information includes the creation day, the size and of course the name of the file or folder. This is important because with the list it can be examined if some unwanted files or folders are created. New files could be created by the operator or a computer virus. With this file it is also possible to check that every file that should be located in the installation folder really is located there.

APL1DUMP.TXT

The file includes information and every parameter there is set on the current application, so it is a very useful file. It is written in the SCIL programming language so it has to be inserted in to the MicroSCADA’s SCIL editor to get something useful in readable form out of it. An example of the “APL1DUMP.TXT” can be seen in Figure 4.



```
CM="Rivers, Green H07 (Current L1)",FP=0,ID="MM",SP=180,SP_HL=8,ST="A"),-
OBJ_RIVH08_C=LIST(BP=900,BP_HL=3,CM="Rivers, Wardton H08 (Current L1)",FP=0,-
ID="MM",SP=180,SP_HL=8,ST="A"),OBJ_RIVH09_C=LIST(BP=900,BP_HL=3,-
CM="Rivers, Cleveland H09 (Current L1)",FP=0,ID="MM",SP=180,SP_HL=8,ST="A"),-
OBJ_RIVH10_C=LIST(BP=900,BP_HL=3,CM="Rivers, Winniford H10 (Current L1)",FP=0,-
ID="MM",SP=180,SP_HL=8,ST="A"),PERIOD_DELAY=60,SAMPLING_DELAY=0,-
SP_180=VECTOR("ESTHA1_AE","ESTHA1_C","ESTHA1_RE","ESTHA2_AE","ESTHA2_C",-
"ESTHA2_RE","ESTHA3_AE","ESTHA3_C","ESTHA3_RE","ESTHA4_AE","ESTHA4_C",-
"ESTHA4_RE","ESTHA5_AE","ESTHA5_C","ESTHA5_RE","ESTHA6_AE","ESTHA6_C",-
"ESTHA6_RE","ESTHA7_AE","ESTHA7_C","ESTHA7_RE","ESTIN_AE","ESTIN_RE","ESTLOSS",-
"ESTOUT_AE","ESTOUT_RE","RIVH01_C","RIVH02_C","RIVH03_C","RIVH04_C","RIVH05_C",-
"RIVH06_C","RIVH07_C","RIVH08_C","RIVH09_C","RIVH10_C"),SP_HL_MAX=8,-
SP_HL_MIN=8,SP_MAX=180,SP_MIN=180,USER_PF=VECTOR(),USER_SF=VECTOR()),0,-
SET_STATUS(0,10),SET_STATUS(0,10),SET_STATUS(0,10),SET_STATUS(0,10),-
SET_STATUS(0,10),SET_STATUS(0,10),SET_STATUS(0,10),SET_STATUS(0,10),-
VECTOR("@t_Service_Name__ = argument(2)","@Gl_SSS_Table__ = apl:bsv(41)",-
"#case argument_count",-
"    #when 2 #return do (Gl_SSS_Table__:v''t_Service_Name__'')",-
"    #when 3 #return do (Gl_SSS_Table__:v''t_Service_Name__'',argument(3))",-
"    #when 4 #return do (Gl_SSS_Table__:v''t_Service_Name__'',argument(3),argu"+-
"ment(4))",-
"    #when 5 #return do (Gl_SSS_Table__:v''t_Service_Name__'',argument(3),argu"+-
"ment(4),argument(5))",-
"    #when 6 #return do (Gl_SSS_Table__:v''t_Service_Name__'',argument(3),argu"+-
"ment(4),argument(5),argument(6))",-
"    #when 7 #return do (Gl_SSS_Table__:v''t_Service_Name__'',argument(3),argu"+-
"ment(4),argument(5),argument(6),argument(7))",-
"    #when 8 #return do (Gl_SSS_Table__:v''t_Service_Name__'',argument(3),argu"+-
"ment(4),argument(5),argument(6),argument(7),argument(8))",-
"    #when 9 #return do (Gl_SSS_Table__:v''t_Service_Name__'',argument(3),argu"+-
"ment(4),argument(5),argument(6),argument(7),argument(8),argument(9))",-
"    #otherwise #return list(STATUS = 1, DESCRIPTIVE_TEXT = ""SCM_ARGUMENT_COU"+-
"NT_EXCEEDED"")","#case_end"),LIST(ATTACHEVENTCHANNEL=PACK_STR(VECTOR(131,4,0,-
0,24,0,0,0,242,3,0,0,11,197,227,65,5,0,0,0,0,0,0,0,1,40,0,0,0,10,0,172,1,0,88,-
1,38,156,1,179,1,56,0,0,0,11,0,172,2,0,88,2,38,156,1,179,1,72,0,0,0,12,0,172,3,-
0,88,3,38,156,1,179,1,88,0,0,0,13,0,172,4,0,88,4,38,156,1,179,1,219,3,0,0,15,0,-
164,171,1,0,38,67,1,165,208,8,0,65,67,65,84,84,65,67,72,79,124,0,0,0,78,108,2,-
0,0,1,152,0,0,0,19,0,172,5,0,22,26,2,0,164,171,3,0,165,108,99,0,4,255,2,65,78,-
179,1,177,0,0,0,20,0,172,6,0,164,88,0,208,1,0,65,171,5,0,38,87,3,165,179,1,103,-
2,0,0,22,0,164,171,4,0,208,0,0,151,165,82,239,0,0,0,1,234,0,0,0,22,0,22,26,2,0,-
164,171,3,0,165,108,22,26,5,0,94,167,96,2,65,78,115,1,0,114,1,0,4,255,0,78,103,-
2,0,0,164,171,5,0,38,136,1,165,88,0,153,98,0,10,15,76,95,69,86,69,78,84,95,67,-
72,65,78,78,69,76,2,73,85,88,1,120,156,4,82,67,2,0,0,1,62,2,0,0,26,0,22,26,5,0,-
98,0,3,255,2,79,78,167,116,1,0,22,26,5,0,98,0,3,255,2,83,78,38,52,2,38,67,1,22,-
209,3,0,61,61,34,26,4,0,209,1,0,34,94,38,51,2,38,136,1,88,0,151,82,35,2,0,0,1,-
212,1,0,0,28,0,22,26,5,0,98,0,3,255,2,83,78,38,136,1,88,9,153,82,212,1,0,0,1,-
```

Figure 4. APL1DUMP.TXT

SYSDUMP.TXT

Similarly to the previous file, this is also a collection of all parameters and options, but on a system level. The system parameters are the same for every different application on the system. The parameters on “APL1DUMP.TXT” were only set for one of the applications and they differ between every application.

Registry folder

The program creates a folder that contains a copy of the registry file, for example “ABB.reg”, which MicroSCADA has created. The “ABB.reg” file contains all sort of information about MicroSCADA program, such as license file information and installed libraries and update packages. With this information it easy to check that the program is up to date and all the needed libraries are installed.

MS_Interfaces folder

This folder contains copies of some configuration and log files of ELCOM communication interface. The files are assumed to be located in the folder “C:/MS_Interfaces/ELCOM”. If the folder does not exist or ELCOM interface is not installed, this folder only contains an empty folder named “ELCOM”. ELCOM is an interface made for communication between substations and control centers.

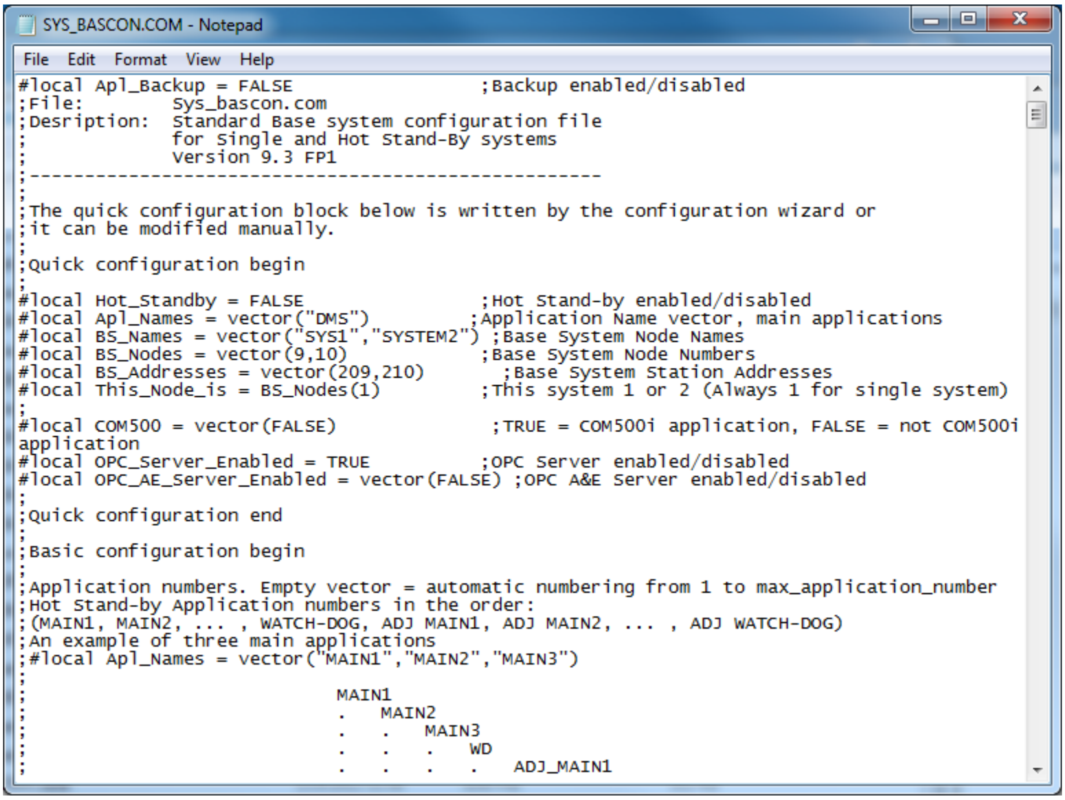
sc folder

This is the most important folder about the MicroSCADA program itself because it contains copies of all the crucial folders required to run the application. The copies are made of “apl”, “LIB4”, “prog”, “Setup” and “SYS” folders.

The “apl” folder is a folder where different MicroSCADA applications are located under their own subfolders. Only the most space consuming files have been left out, like station and process pictures. There is a subfolder named “COM500” which includes settings for the COM 500 communication gateway. COM 500 provides a communication gateway between the substation and the Network Control Center. If there are problems in the communication, this may be the place to find the solution.

The folder named “Setup” includes a large number of log files that were created during the installation of MicroSCADA. They show every component and library that has been installed. The log files also show what the installation location was, when the installation took place and if the installation was successful. This is a good place to check that every required component really was created during the installation.

“SYS_” folder includes a very important file “SYS_BASCON.COM”. It is the main settings file which defines which application MicroSCADA launches automatically when it is opened. Many parameters and functions are also set from this file. The file can be created by a wizard built in the program but it is still normally made by hand using a simple text editor. An example of the “SYS_BASCON.COM” can be seen in Figure 5.

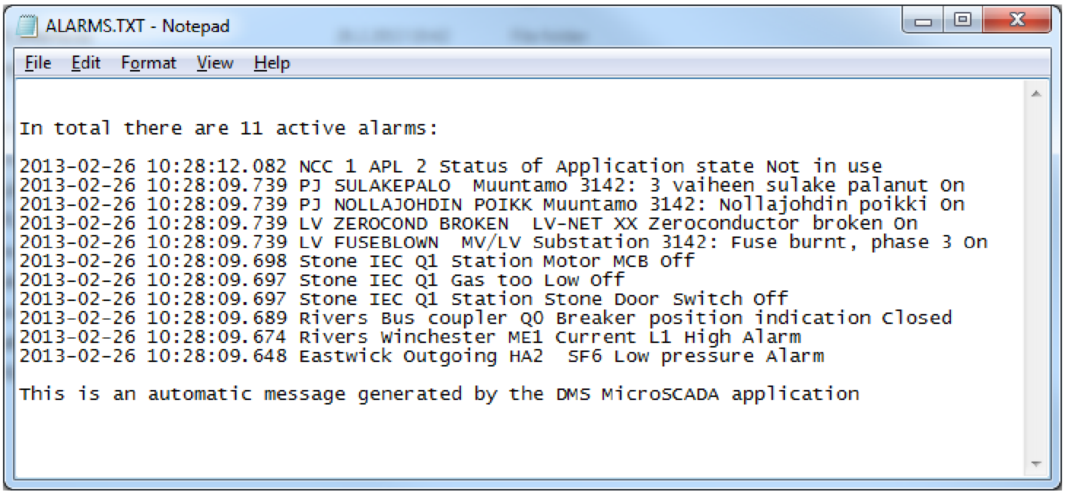


```
#local Apl_Backup = FALSE                  ;Backup enabled/disabled
;File:          Sys_bascon.com
;Desription:    Standard Base system configuration file
;               for Single and Hot Stand-By systems
;               Version 9.3 FP1
;----------------------------------------------------
;
;The quick configuration block below is written by the configuration wizard or
;it can be modified manually.
;
;Quick configuration begin
;
#local Hot_Standby = FALSE                 ;Hot Stand-by enabled/disabled
#local Apl_Names = vector("DMS")          ;Application Name vector, main applications
#local BS_Names = vector("SYS1","SYSTEM2") ;Base System Node Names
#local BS_Nodes = vector(9,10)             ;Base System Node Numbers
#local BS_Addresses = vector(209,210)        ;Base System Station Addresses
#local This_Node_is = BS_Nodes(1)          ;This system 1 or 2 (Always 1 for single system)
;
#local COM500 = vector(FALSE)               ;TRUE = COM500i application, FALSE = not COM500i
application
#local OPC_Server_Enabled = TRUE           ;OPC Server enabled/disabled
#local OPC_AE_Server_Enabled = vector(FALSE) ;OPC A&E Server enabled/disabled
;
;Quick configuration end
;
;Basic configuration begin
;
;Application numbers. Empty vector = automatic numbering from 1 to max_application_number
;Hot Stand-by Application numbers in the order:
;(MAIN1, MAIN2, ... , WATCH-DOG, ADJ MAIN1, ADJ MAIN2, ... , ADJ WATCH-DOG)
;An example of three main applications
;#local Apl_Names = vector("MAIN1","MAIN2","MAIN3")
;
;                           MAIN1
;                           .   MAIN2
;                           .   .   MAIN3
;                           .   .   .   WD
;                           .   .   .   .   ADJ_MAIN1
```

Figure 5. SYS_BASCON.COM file

ALARMS.TXT

This text file shows all active alarms in MicroSCADA. An alarm can be caused for example by overvoltage trip, overcurrent trip, broken conductor or lost communication. This is interesting information because with it, the maintenance engineer could predict upcoming failures. If there are, for example lost communication alarms occurring regularly, there is something wrong with the system. An example of the alarm list can be seen in Figure 6.



```
In total there are 11 active alarms:

2013-02-26 10:28:12.082 NCC 1 APL 2 Status of Application state Not in use
2013-02-26 10:28:09.739 PJ SULAKEPALO  Muuntamo 3142: 3 vaiheen sulake palanut On
2013-02-26 10:28:09.739 PJ NOLLAJOHDIN POIKK Muuntamo 3142: Nollajohdin poikki On
2013-02-26 10:28:09.739 LV ZEROCOND BROKEN  LV-NET XX Zeroconductor broken On
2013-02-26 10:28:09.739 LV FUSEBLOWN  MV/LV Substation 3142: Fuse burnt, phase 3 On
2013-02-26 10:28:09.698 Stone IEC Q1 Station Motor MCB Off
2013-02-26 10:28:09.697 Stone IEC Q1 Gas too Low Off
2013-02-26 10:28:09.697 Stone IEC Q1 Station Stone Door Switch Off
2013-02-26 10:28:09.689 Rivers Bus coupler Q0 Breaker position indication Closed
2013-02-26 10:28:09.674 Rivers Winchester ME1 Current L1 High Alarm
2013-02-26 10:28:09.648 Eastwick Outgoing HA2  SF6 Low pressure Alarm

This is an automatic message generated by the DMS MicroSCADA application
```

Figure 6. MicroSCADA active alarms

This list is generated by a modified version of MicroSCADA E-mail Alarm application, which is a program for automatically sending alarms generated in the MicroSCADA application as e-mails to a list of predefined mail receivers. This modified version only generates a text file with a list of active alarms.

4.4 System Self Supervision

MicroSCADA has its own supervising and monitoring tool, called System Self Supervision (SSS). The SSS provides status information of hardware and software using the supervision symbols of MicroSCADA. The supervision information is shown in the event and alarm lists and usually also in an additional supervision display designed for supervision purposes. An example of a supervision display is illustrated in Figure 7. The dedicated SSS symbols and statuses can be seen in the display. The main function of supervision monitoring is providing the visual information about the supervised objects. The symbols show the states and statuses with different use of colors. For example, green usually indicates a good status and red indicates a failure status. An alarming symbol is indicated by a blinking red color.

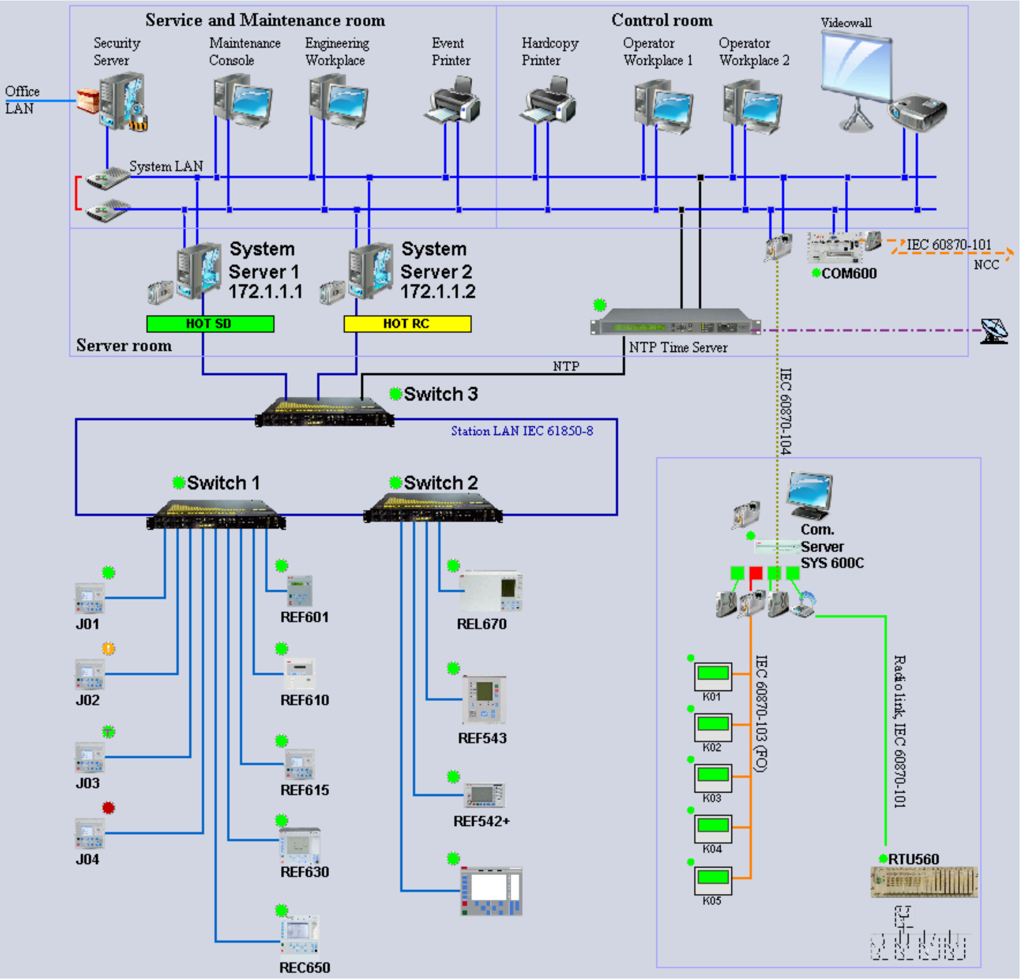


Figure 7. System Self Supervision display /8/

The statuses of the supervised objects are received to MicroSCADA via system messages or predefined event channels. The object types that can be monitored by the SSS are following: base system, application, communication node, communication line, IED, SNMP devices, displays and printers. In order to obtain full operability of the SSS, it has to be enabled and configured from the SYS_BASCON.COM configuration file of MicroSCADA.

With the SSS it is possible to monitor the states of the devices from the supervision display but it is also possible to use this status information in the maintenance reports if it is necessary. The status objects can be accessed just like the station alarms in the Station Alarm Counter program created for this thesis.

5 ACQUIRING DATA FROM THE OPERATING SYSTEM

In the interviews with the engineers involved in the maintenance work, it became clear that information about the server computer and the operating system seems to be even more important than the information about MicroSCADA program and its measurements.

5.1 Relevant Data

Common for almost all the relevant information is the unchangeability of configuration settings and installed programs of the system. It is important that everything in the system is in the exact same state as it was when the system was delivered to the customer. The system is usually configured so that the operator user cannot change anything or open any other program except MicroSCADA. But if the customer logs into the system as an administrator user, it is possible to make changes or even install his own programs to the system. To make it possible to compare if anything has changed, a base copy or so called image should be made of the system hard drive, or at least make a thorough system information file before the delivery. With the image, it is also possible to return the whole system in a case of a hard drive failure or critical system error.

General Hardware and Operating System Information

When problems occur, it is crucial to know what components and which operating system there are installed in the system. This information should have been gathered before the delivery, but the customer may have changed some of the components. The information includes the types and manufacturers of all the main components, for example the size of the memory modules and hard drives. The operating system information includes the version of the installed operating system and information about the installed service packs.

Localization Settings

Localization settings may not seem very important, but they affect quite many operations. All the devices connected to the SCADA system should be time synchronized, so that the time stamps of the events and alarms are right and scheduled operations are performed on time. Time synchronization is usually made by using a GPS. A GPS receiver is connected to the SCADA system and all the devices get synchronized with the same exact time from a GPS satellite. Typical accuracies range from 500 nanoseconds to a few milliseconds depending on the synchronization protocol.

Installed Programs

It is important to know what programs there are installed in the system. There should not be any other programs than the ones that are needed for MicroSCADA and communications to work properly. Unknown programs could be something installed by the customer on in the worst case viruses.

Windows Services

Microsoft defines services as follows: “Microsoft Windows services enable you to create long-running executable applications that run in their own Windows sessions. These services can be automatically started when the computer boots, can be paused and restarted, and do not show any user interface. These features make services ideal for use on a server or whenever you need long-running function-ality that does not interfere with other users who are working on the same computer”. /6/

So services are just like installed programs. There should not be any services running, except the ones that are needed for MicroSCADA and communications.

Windows Registry

Microsoft itself describes registry as follows: “A central hierarchical database used in Microsoft Windows used to store information that is necessary to configure the system for one or more users, applications and hardware devices. The Registry contains information that Windows continually references during operation, such as profiles for each user, the applications installed on the computer and the types of documents that each can create, property sheet settings for folders and application icons, what hardware exists on the system, and the ports that are being used.” /18/

In brief, registry files are a sort of settings files where the program stores its settings so that the operating system can operate properly when the program is running. Relevant information about the registry is the same that it is with the installed programs and services; unchangeability from the state when the system was delivered.

Usage of System Resources

There should always be sufficient free disk space available for the operating system and programs to run. The operating system and programs tend to create all sort of temporary files and log files that could be quite large in size. The Windows operating system especially needs a significant amount of free space. Otherwise it could run slower or even get stuck occasionally. The disk may also fill up because of continuously growing report files of MicroSCADA.

Hard disks are the most likely component to fail in a computer. They can be installed in a redundant RAID 1 configuration, where the data is written identically to two or more hard drives. If a hard disk fails, it can be replaced with another disk without interrupting the MicroSCADA service. /7/

The memory usage of the systems is also important information. All the programs that are running are using the physical memory of the system. If the memory usage suddenly rises significantly, it is usually caused by a problem in the program.

CPU usage is like the memory usage of the system. A running program always uses some CPU power but if the CPU usage rises without any reason, there could be a problem in the program.

As mentioned in Chapter 4, handles and threads also represent usage of the system resources. The handles represent open instances of basic operating system object applications interacting with e.g. files, registry keys and shared memory. The threads are like small processes within processes that share resources such as memory and CPU power.

Temperature Information

Temperature rise of the system can cause big problems and in the worst case it can break the components. A continuous high temperature leads to a shorter life span and failure rate of the components. Temperature rise is usually caused by bad ventilation or broken cooling fans. Dust filters may have become so dirty that the cooling of the computer is not functioning as it was designed to. Some components, but not all of them, have sensors that are measuring temperatures of the components. These are for example CPU, motherboard and hard drives. Reading the sensor information requires third party software, because there are no inbuilt programs in Windows operating system that can read the temperatures.

Group Policy Settings

In the Microsoft Windows operating system it is possible to create multiple user accounts with different privileges in the system. The privileges are defined by adding the user for example to the “administrators” user group. Usually the operator user is set to have very limited options besides operating MicroSCADA. All these preferences should have been made and revised before the delivery.

It is important to check that Group Policies have actually been made or if they have been changed afterwards. With the administrator’s privileges the operator of the system can install own programs and make all kinds of unwanted changes to the system.

Network Settings

Problems in the communications are often noticed when they are simply not working. In the case the communication is not working at all, getting the maintenance reports is also impossible. But if the reports have been sent and received it should be checked that all the connections are working and network addresses are correct and unchanged.

If there is a firewall installed in the system, information about open ports is important. Without the right ports open, the communications cannot get through at all. Another interesting thing is the network shares. If there are partitions of the hard drives that are needed to be shared, it should be revised that only they are shared and with the correct preferences.

5.2 Gathering the Data

The System Info Logging Tool in MicroSCADA also gathers a lot of useful information about the operating system and hardware as well. All the system information is located in the same text file named "SYS600_SYSTEMINFO.TXT" right at the root of the zip file. If more information about the system is required, additions to the gathered information can be made by modifying the batch file. Off course it is required to know the commands that are used for getting the wanted information. There are many tutorials available on the Internet that show how the information can be acquired by using the command line.

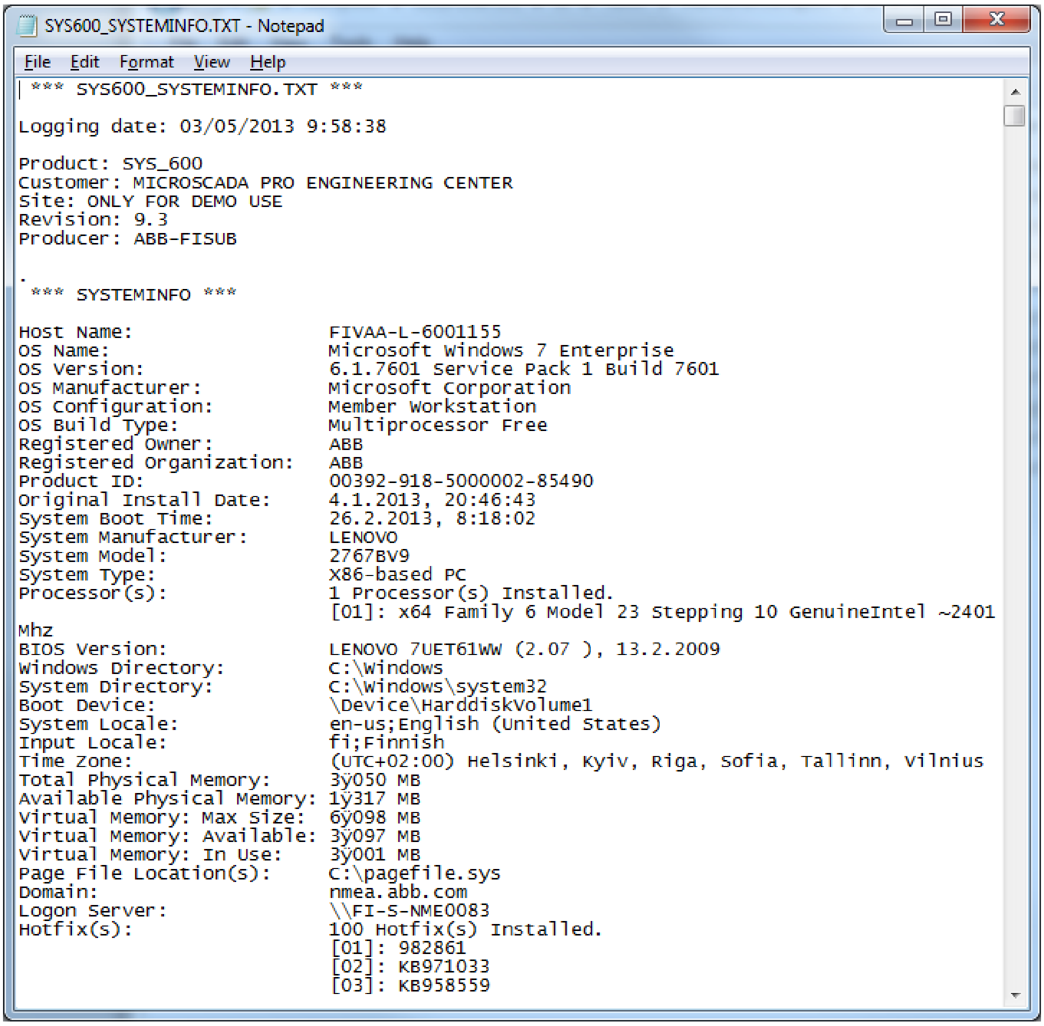
Here is the summary of all the information about the system that the text file contains.

SYS600_SYSTEMINFO.TXT

The file includes all the basic information of the operating system and programs running on it. Luckily it is also in easy to read form, probably because it is generally the basic system information file that could be created with the own system information tool of the operating system.

Right at the beginning of the file there is the logging date which is quite important information because without it one would never really be sure that the files inside the zip file were generated when the program has run.

After the date there comes the usual information about the operating system as can be seen in Figure 8: OS name and version, manufacturer and model of the computer, processor type, OS installation folder, booting device, time zone settings, usage and size of the physical and virtual memory, domain name, logon server, every hotfix installed on the OS and finally information about the network cards.



```
*** SYS600_SYSTEMINFO.TXT ***

Logging date: 03/05/2013 9:58:38

Product: SYS_600
Customer: MICROSCADA PRO ENGINEERING CENTER
Site: ONLY FOR DEMO USE
Revision: 9.3
Producer: ABB-FISUB

.
 *** SYSTEMINFO ***

Host Name:                 FIVAA-L-6001155
OS Name:                   Microsoft Windows 7 Enterprise
OS Version:                6.1.7601 Service Pack 1 Build 7601
OS Manufacturer:           Microsoft Corporation
OS Configuration:          Member Workstation
OS Build Type:             Multiprocessor Free
Registered Owner:          ABB
Registered Organization:   ABB
Product ID:                00392-918-5000002-85490
Original Install Date:     4.1.2013, 20:46:43
System Boot Time:          26.2.2013, 8:18:02
System Manufacturer:       LENOVO
System Model:              2767BV9
System Type:               X86-based PC
Processor(s):              1 Processor(s) Installed.
                           [01]: x64 Family 6 Model 23 Stepping 10 GenuineIntel ~2401
Mhz
BIOS Version:              LENOVO 7UET61WW (2.07 ), 13.2.2009
Windows Directory:         C:\Windows
System Directory:          C:\Windows\system32
Boot Device:               \Device\HarddiskVolume1
System Locale:             en-us;English (United States)
Input Locale:              fi;Finnish
Time Zone:                 (UTC+02:00) Helsinki, Kyiv, Riga, Sofia, Tallinn, Vilnius
Total Physical Memory:     3ÿ050 MB
Available Physical Memory: 1ÿ317 MB
Virtual Memory: Max Size:  6ÿ098 MB
Virtual Memory: Available: 3ÿ097 MB
Virtual Memory: In Use:    3ÿ001 MB
Page File Location(s):     C:\pagefile.sys
Domain:                    nmea.abb.com
Logon Server:              \\FI-S-NME0083
Hotfix(s):                 100 Hotfix(s) Installed.
                           [01]: 982861
                           [02]: KB971033
                           [03]: KB958559
```

Figure 8. SYS600_SYSTEMINFO.TXT

There is also a long list of every installed program on the computer and a lot of information about them such as name, size, process ID and operation count. After the programs there is a list of services available in the OS. The list tells the name, exit code, process ID, start mode, state and status of the service. The start mode can be “Auto” if the service is set to start at system logon, “Manual” if the service is set to start as required or when called from an application or “Disabled” when the service is completely disabled and prevented from running. The state of the service can be either “Running” or “Stopped”. This information about the services could be very useful because many problems occurring are caused by stopped or unwanted services.

At the end of the file there are the running programs of the system. The list tells the name of the program, memory usage of the program and user of the program. This information is similar with the services; much unusual behavior is caused by unwanted programs like third party software installed by the user of the system or computer virus.

5.3 SNMP

SNMP is a simple network management protocol that almost all devices used in the industrial networks support. All important statuses of the devices in the network can be monitored with SNMP. These devices include e.g. computers, GPS devices, network switches, printers and many more. Some settings of the devices can also be changed through SNMP. /12/

SNMP is widely used in the MicroSCADA systems to monitor the devices within the network, but MicroSCADA has no built-in support for SNMP, so a third party program is needed for transferring the information to MicroSCADA. The most used program for this in the MicroSCADA systems is MOXA EDS-SNMP OPC Server Pro.

If SNMP is used in the system, it should be relatively easy to gather different information from the system and add it to the maintenance report.

5.4 Commercial Products

There are a number of commercial products available that are designed to monitor the condition of the server computer. Usually these server management programs also have much of other features, from which some can be useful, and others are not. There is usually a feature that sends e-mail notifications when a parameter, such as memory usage, exceeds a predetermined value. It is also often possible to generate a scheduled report of the condition of the system. These programs could be very useful, but often their price is relatively high. Of course if the customer especially buys a maintenance service with demands of this kind of service, it may be easiest to accomplish with an already existing commercial product. /11/

There are also many good freeware server management programs available, but unfortunately they are almost always only for servers running Linux or UNIX operating systems. Linux and UNIX are the most used operating systems for servers, but MicroSCADA currently runs only on Microsoft Windows operating system, so these programs are out of the question.

Here is a brief list of some of the tested programs and features in them.

PA Server Monitor

This program developed by Power Admin has many inbuilt features and it is fairly easy to use due to the simple graphical interface. Some of its features include remote monitoring of servers across the Internet, disk space monitoring, event logging, performance and server temperature monitor, executing of scripts and many more.

The program has a wide variety of reporting features. Almost all of the parameters can be seen in a form of report. These reports can be generated automatically and sent to the server administrator. The triggering feature allows sending an e-mail notification if a parameter exceeds a predetermined value. The pricing of PA Server Monitor is quite reasonable with a price of $125 per server. This price includes Ultra License and 1 year support and maintenance. /13/

GFI MAX RemoteManagement

Similar to PA Server Monitor, this software is able to do performance monitoring, disk space monitoring, event logging and more. It is also capable to send SMS or e-mail notifications in a case of a problem. Automated reports can be created and they can be sent to the administrator by e-mail. The pricing is calculated differently for every customer, but it is said to be competitive. /5/

SIW - System Information for Windows

SIW is not really server management software like PA Server Monitor or GFI MAX RemoteManagement. It is a tool that analyzes the computer and gathers a wide variety of detailed information about the system properties and settings. SIW can create a report file in many formats, like CSV, HTML, TXT or XML. The system information that SIW can gather includes software information, real-time monitors of CPU and memory usage, network information and also sensor information that are measuring the heating of the system components.

SIW cannot create reports automatically or send reports by e-mail so it would require some external program to handle the sending of e-mails and scheduling the creation of reports. The price is $30 which includes “Technician’s Version” of the software and 1 year updates. /15/

6 DATA TRANSFER

6.1 Sending the Report

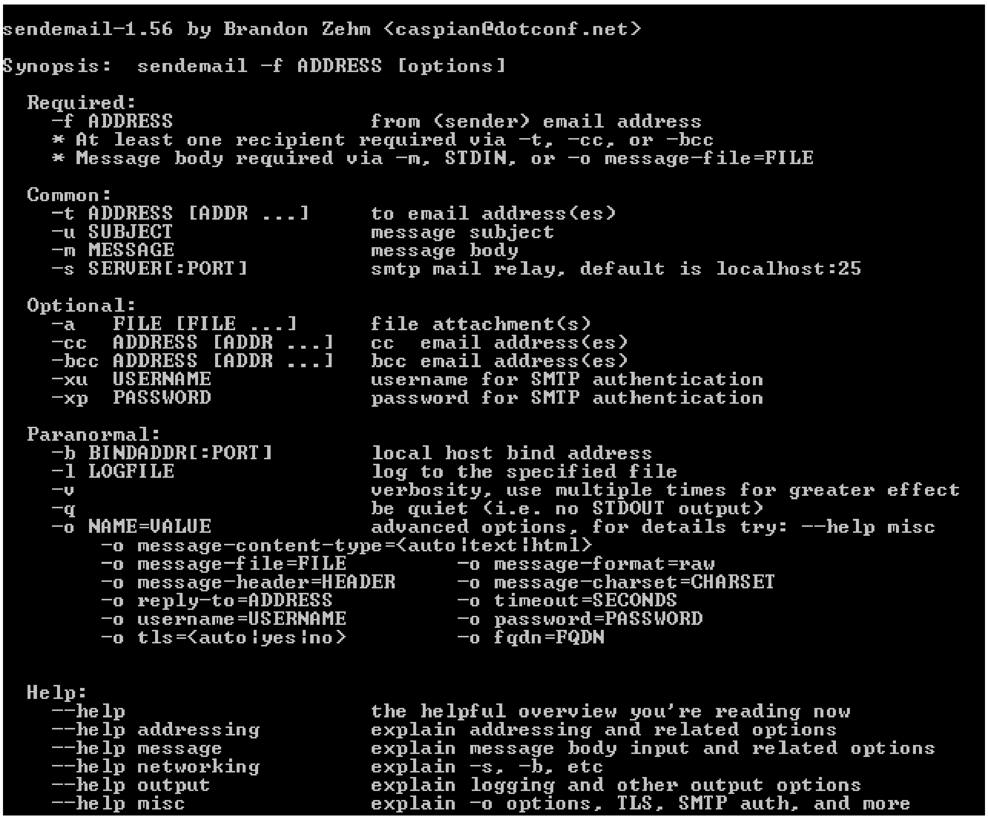
Right from the beginning it was clear that sending the report from the SCADA computer to the recipient is going to be handled by e-mail. The sending of the e-mail should not be very difficult to arrange because it is so widely used method of sending information and there are many good instructions about it on the Internet.

There is a script available for MicroSCADA that can send scheduled e-mails, but in the opinions of the people working in the maintenance, sending the e-mail should be made using a program operating outside MicroSCADA. This is due the reason that if there really is a problem in MicroSCADA, the script for sending the e-mail is likely also out of order. When the program is running independently of MicroSCADA, the e-mails are more likely to be sent. This of course requires that the problem is not occurring in the communications. If so, there will not be any e-mail messages sent by any kind of program.

6.2 Finding the Suitable Program

There are many commercial and freeware software that could be used to send e-mails. Programming it from the scratch should not be very difficult but using already existing solutions this step is much less time consuming. After testing some alternatives the choice is a simple program called SendEmail. It is a freeware program developed by Brandon Zehm. It is licensed under the GNU GPL license which guarantees end users the freedoms to use, study, share, and modify the software. SendEmail is light, simple to use command line SMTP email client.

The software is used from the command line of the operating system. It means that every command is executed in a form of text in the basic command prompt and it can be launched with a simple line in the batch file. A screen capture of the opening screen in the program can be seen in Figure 9. The opening screen shows the basic commands of the program.



```
sendemail-1.56 by Brandon Zehm <caspian@dotconf.net>

Synopsis:  sendemail -f ADDRESS [options]

  Required:
    -f ADDRESS                from (sender) email address
    * At least one recipient required via -t, -cc, or -bcc
    * Message body required via -m, STDIN, or -o message-file=FILE

  Common:
    -t ADDRESS [ADDR ...]     to email address(es)
    -u SUBJECT                message subject
    -m MESSAGE                message body
    -s SERVER[:PORT]          smtp mail relay, default is localhost:25

  Optional:
    -a   FILE [FILE ...]      file attachment(s)
    -cc  ADDRESS [ADDR ...]   cc  email address(es)
    -bcc ADDRESS [ADDR ...]   bcc email address(es)
    -xu  USERNAME             username for SMTP authentication
    -xp  PASSWORD             password for SMTP authentication

  Paranormal:
    -b BINDADDR[:PORT]        local host bind address
    -l LOGFILE                log to the specified file
    -v                        verbosity, use multiple times for greater effect
    -q                        be quiet (i.e. no STDOUT output)
    -o NAME=VALUE             advanced options, for details try: --help misc
        -o message-content-type=<auto¦text¦html>
        -o message-file=FILE         -o message-format=raw
        -o message-header=HEADER     -o message-charset=CHARSET
        -o reply-to=ADDRESS          -o timeout=SECONDS
        -o username=USERNAME         -o password=PASSWORD
        -o tls=<auto¦yes¦no>         -o fqdn=FQDN


  Help:
    --help                    the helpful overview you're reading now
    --help addressing         explain addressing and related options
    --help message            explain message body input and related options
    --help networking         explain -s, -b, etc
    --help output             explain logging and other output options
    --help misc               explain -o options, TLS, SMTP auth, and more
```

Figure 9. SendEmail options

Using the program may seem confusing but it is really simple. Here are instructions how to send an email with an attachment:

1. Write the path where SendEmail.exe is located, for example “C:\sc\temp\sendemail”.
2. Write “-f” and e-mail address that the program is using to send the e-mail.
3. Write the receiver’s e-mail address after the command “-t”.
4. Write the subject of the e-mail after “-u”.
5. Write the message itself after the command “-m”.
6. To attach a file to the e-mail, write “-a” and the path of the file.
7. Write the SMTP server and the port it is using after the command “-s”. If for example Gmail’s SMTP server is used, the command should be looking like this: “-s smtp.gmail.com:587”.
8. Insert the username and password for the SMTP server used. Write the username after the command “-xu” and the password after the command “-xp”.

9. If a mail server with a secured TLS protocol is used, write “-o tls=yes” at the end of the whole command.

Now the command should be looking something like this: “C:\sc\temp\sendemail -f sender@mail.com -t receiver@mail.com -u Subject -m Message -a file.zip -s smtp.mail.com:123 -xu username -xp password -o tls=yes”. This command sends an e-mail with an attachment “file.zip” from an address “sender@mail.com” to an address “receiver@mail.com”. The subject of the e-mail is “Subject” and the message is “Message”. The address of the SMTP server is “smtp.mail.com:123” and it uses TLS protocol. The username and password for the server are “username” and “password”.

6.3 Communication Settings

Using e-mail requires access to an external SMTP server. An existing corporate e-mail server could be used or an SMTP server on the Internet, like Google Gmail used in this thesis. The Windows Server operating system also has an SMTP server included as a part on the Internet Information Services (IIS) feature. IIS can be turned on from the IIS Manager, which is accessed by typing “inetmgr” to the Windows Run dialog box or from the administrative tools in the control panel.

6.4 Security Threats

E-mail is a relatively secure way to send important material, at least if it is sent through a private server. In theory, it is possible that third parties could read or even modify the content during the transit time from sender to recipient, when the e-mail passes through many routers and mail servers. But in reality, the biggest threat is most likely lost or stolen hardware containing offline e-mails. In this case, it is unlikely that anyone even has interest in the e-mails, because they contain information that is only important in the maintenance work. Of course the reports include also information about the system’s hardware and operating system that could be valuable if someone is planning on cyber-attack towards the SCADA system. /4/

Using e-mail naturally requires that the system is somehow connected to the Internet. This creates a lot of security risks, even though the connection is made through an internal industrial network. Security in an industrial network can be compromised in many places along the system. When a SCADA computer is logging data out to some back-office database, it must be on the same physical network as the back-end database systems. This means that there is a path to the SCADA system and also to the end devices through corporate network. If a corporate network is compromised, then any IP based device can be accessed. These connections provide an opportunity to attack the SCADA system with multiple attacks that includes Denial of Service (DoS) attacks, deleting of the system files on the SCADA server, planting a Trojan or a keylogger and using the SCADA server as a launching point to other system components within the network. /9/

Security strategy in a corporate network should of course be developed even though there is no SCADA system, but there are specific steps to protect any SCADA system. These steps are for example properly configured SCADA firewalls, using of smart switches to segment SCADA networks off into their own IP segments, securing the operating system with user policies, using of anti-virus softwares also on the SCADA computers and using SCADA policies with password protection. /9/

As it was explained earlier in Chapter 5.1, the Group Policy Settings of the server computer are set very restricted for every user. However, creating a report using the application example introduced in this thesis requires that the user logged on the computer has privileges to run third party software. This may cause a security risk, because the operator user should not have any privileges to run own programs. Running the programs can be executed securely by enabling “Run only allowed applications” from the Group Policy Settings. When this option is enabled, approved programs can be chosen. Vice versa, launching specific programs can be blocked so that the user will only get an error message when an unwanted program is attempted to launch.

7 INSTRUCTIONS FOR INSTALLATION

In order to create a report introduced in this thesis, some preparations should be made in MicroSCADA and Windows operating system. The report created using these instructions includes all the information on MicroSCADA and the operating system described in Chapters 4 and 5, station alarm counter and a list of active alarms. It is also explained how Windows Task Scheduler should be configured in order to send scheduled reports.

7.1 Copying Programs to /prog/exec/ Folder

There are several programs that should be placed in the /prog/exec/ folder in order to make automated reports. This is the folder that contains all the programs needed by MicroSCADA to work properly.

Here is a list of programs to copy:

- MaintenanceReport.exe
- SendEmail.exe
- MaintenanceReport.bat
- AlarmList.scil

7.2 Creating a Batch File

In order to create the report and sending it by e-mail, several different programs and actions are needed to perform. Perhaps the easiest way to run multiple programs and actions at the time is creating a batch file of the operations.

The batch files are tiny programs or scripts that can be made to simplify routines or repetitive tasks. They are simple unformatted text files that contain commands used in the command prompt of Windows. The text file becomes a batch file when it is saved with a file name extension .bat or .cmd. When the batch file is run, it simply writes all the commands written in the file to the Windows command prompt, just like a user would be writing them manually.

In order to modify the batch file, it is required to understand the commands that are used. Here is a brief list of useful commands that can be used in the batch file:

Table 1. Batch file commands /2/ /17/

>>	Writes the output of the command to a specified location or file.
del	Deletes a file.
dir/s	Displays files in specified director and all subdirectories.
echo	Turns the command-echoing feature on or off, or displays a message.
echo y	Answers “Yes” when the prompt is asking to verify a command.
for	Runs a specified command for each file in a set of files.
if	Performs conditional processing in batch programs.
mkdir	Creates a directory or subdirectory.
pause	Pauses a batch program and displays a message prompting the user to press any key to continue.
ren	Renames a file.
rmdir /s /q	Removes a directory and all subdirectories including any files and delete them without confirmation.
scilc	Runs a program in MicroSCADA that performs a SCIL command or application in C programming language.
tasklist	Displays a list of currently running processes.
wevtutil epl	Exports a log file from Windows to a specific folder.
wmic	Uses The Windows Management Instrumentation Command-line to gather information about the computer.
xcopy	Copies files, directories and subdirectories.
zip -r -q	Compresses a folder or files in to a zip file, using a program included with MicroSCADA. Performs it in quiet mode.

To create a batch file that creates a report in a form that this thesis presents, several commands should be written with a text editor (e.g. Notepad) and saved with a file extension .bat to /prog/exec/ folder of MicroSCADA. The commands used in this thesis are shown in Appendix 1.

The first of these commands runs the modified System Info Logging executable which gathers all the system information. The second command runs the modified MicroSCADA E-mail Alarm application, which creates a list of all the active alarms in MicroSCADA. The application must be run through MicroSCADA program scilc.exe, which allows the code written in SCIL to be run in the C programming language. The third command deletes the previous MaintenanceReport zip file created by the program and answers “Yes” to the verification message of the delete command.

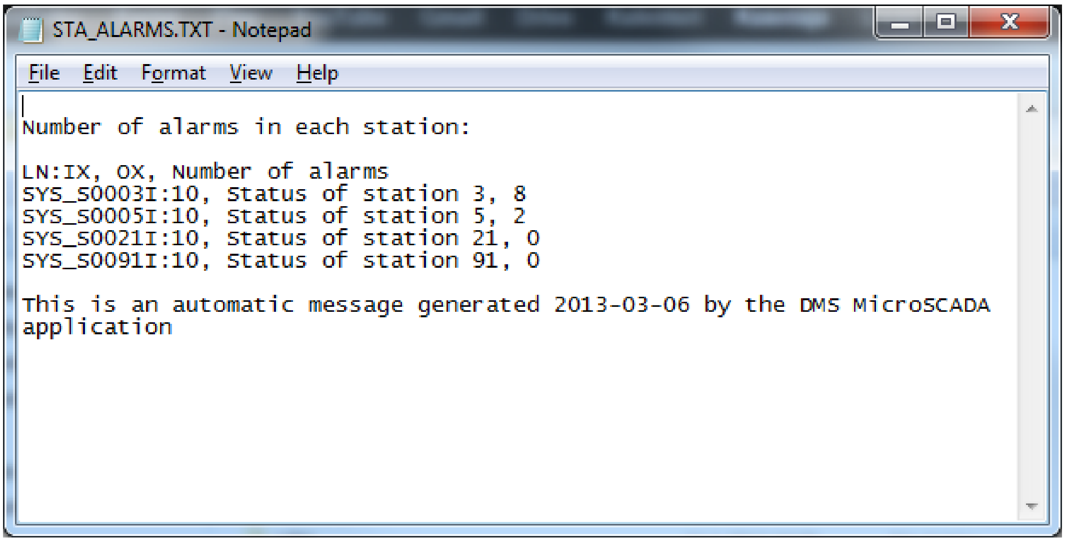
The fourth command renames the zip file from “SendMeToProductSupportCenter_date.zip” to “MaintenanceReport.zip”. Renaming is done, because it is easier to handle the zip file, when it always has the same name. The fifth and sixth commands add the list of active alarms and communication alarms of the stations in to the compressed zip folder already created by the MaintenanceReport.exe program. The “-q” after the command “zip” defines that the operation is made in the quiet mode.

The very last command sends the whole zip folder by e-mail to the desired address. The structure of this command is described in the previous chapter.

Adding more information to the report can be simply done by adding command lines to the batch file. The additions should be placed before the e-mail sending command.

7.3 Configuring Station Alarm Counter

The station alarm counter is a piece of program used for counting communication alarms occurring in each of the stations and creating a report of them. By default it creates a report on the first day of every month and resets the counters. An example of the report can be seen in Figure 10.



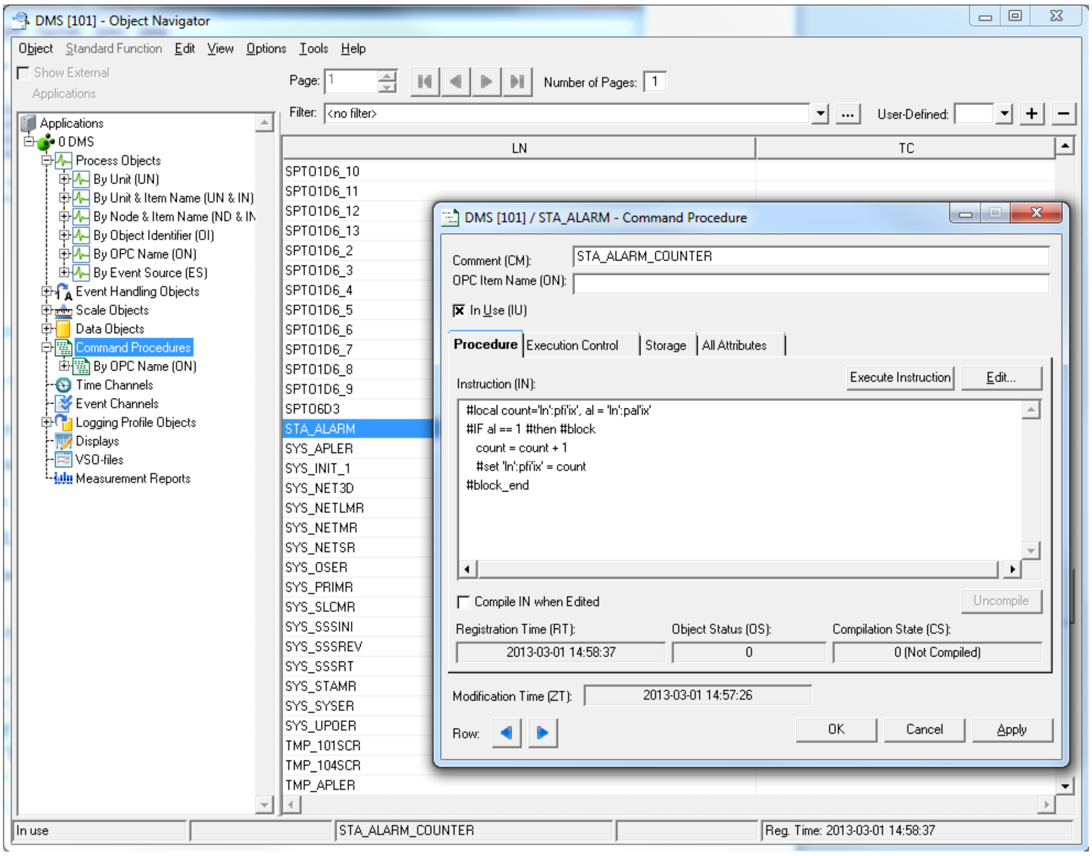
```
Number of alarms in each station:

LN:IX, OX, Number of alarms
SYS_S0003I:10, Status of station 3, 8
SYS_S0005I:10, Status of station 5, 2
SYS_S0021I:10, Status of station 21, 0
SYS_S0091I:10, Status of station 91, 0

This is an automatic message generated 2013-03-06 by the DMS MicroSCADA
application
```

Figure 10. Station alarm report

7.3.1 Creating a Command Procedure for Alarm Counter

First the command procedure “STA_ALARM” in MicroSCADA’s Object Navigator has to be created. With command procedures it is possible to make actions or run a piece of program every time a defined event is happening. The code in SCIL programming language can be placed in the Instruction field of Procedure tab. The code is presented in Appendix 3.

This piece of code adds 1 to variable “count” every time a process object is alarming. Then the value in “count” is moved to “FI” process point of the object. “FI” stands for “Free Integer” and it can be used for own purposes, because it is not used to store any other values. After entering the code it is important to check “In Use (IU)” check box from the Command Procedure window. “STA_ALARM” command procedure should now look like in Figure 11.



Comment (CM): STA_ALARM_COUNTER

OPC Item Name (ON):

☒ In Use (IU)

Instruction (IN):

```
#local count='ln':pfi'ix', al = 'ln':pal'ix'
#IF al == 1 #then #block
  count = count + 1
  #set 'ln':pfi'ix' = count
#block_end
```

☐ Compile IN when Edited

Registration Time (RT): 2013-03-01 14:58:37

Object Status (OS): 0

Compilation State (CS): 0 (Not Compiled)

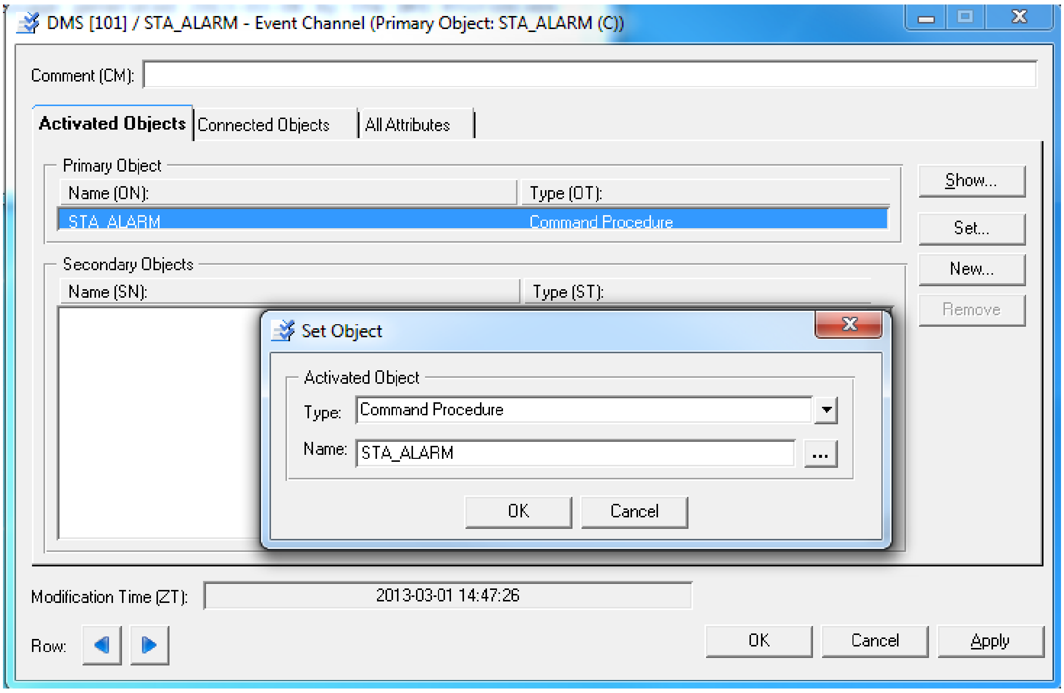
Modification Time (ZT): 2013-03-01 14:57:26

In use | STA_ALARM_COUNTER | Reg. Time: 2013-03-01 14:58:37

Figure 11. Creating a command procedure

7.3.2 Creating an Event Channel

After the command procedure is created, an event channel has to be created that links the desired process objects to the command procedure. This is done as follows: create an event channel “STA_ALARM” in the Object Navigator. From the “Activated Objects” tab, press “Set” button and choose “Command Procedure” as the type from the drop-down list. The name of the command procedure can be written in the “Name” field, or it can be chosen from a list by pressing three dots next to the field. Event channel should now look like in Figure 12 below.



DMS [101] / STA_ALARM - Event Channel (Primary Object: STA_ALARM (C))

Comment (CM):

Primary Object

Name (ON):	Type (OT):
STA_ALARM	Command Procedure

Secondary Objects

Name (SN):	Type (ST):

Set Object

Activated Object

Type: Command Procedure

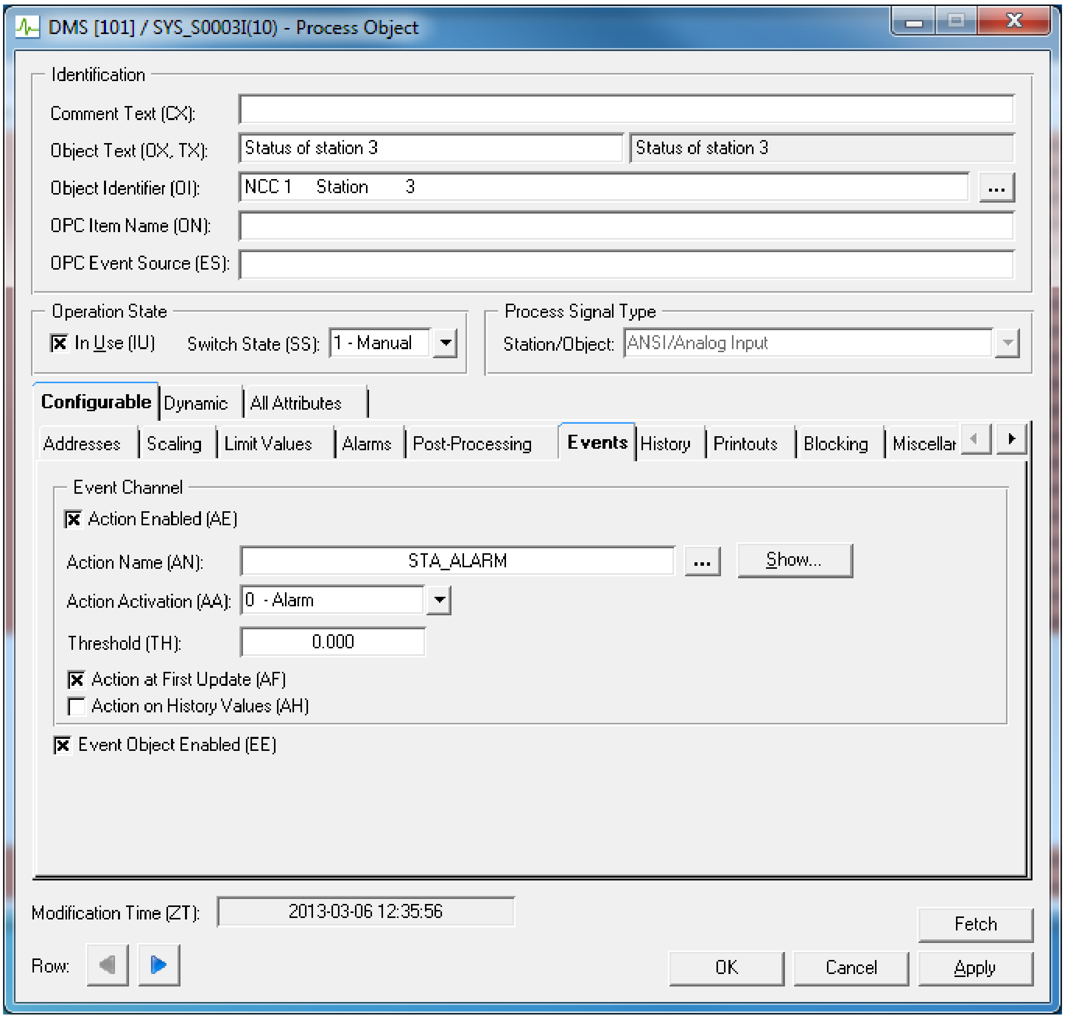
Name: STA_ALARM

Modification Time (ZT): 2013-03-01 14:47:26

Figure 12. Creating an event channel

7.3.3 Connecting Process Objects to Event Channel

The event channel and command procedure should both be now created. The next step is linking desired process objects with the event channel. Communication problems in stations create an alarm in process objects, the logical names (LN) of which are by default named “SYS_S***I”, with an index (IX) of 10. These objects should be configured so that every time a new alarm is born, it launches “STA_ALARM” event that was created before. This is made in “Events” tab of “Configurable” tab. The name of the event, in this case “STA_ALARM”, should be written in the “Action Name (AN)” field or it can also be chosen from the list. “Action at First Update (AF)” should be checked, so that the event is launched every time a new alarm is updated. The event is enabled by checking “Action Enabled (AE)” and “Event Object enabled (EE)” boxes. The configuration of process object is shown in Figure 13.



DMS [101] / SYS_S0003I(10) - Process Object

Identification

Comment Text (CX):

Object Text (OX, TX): Status of station 3 | Status of station 3

Object Identifier (OI): NCC 1 Station 3

OPC Item Name (ON):

OPC Event Source (ES):

Operation State

☒ In Use (IU) Switch State (SS): 1 - Manual

Process Signal Type

Station/Object: ANSI/Analog Input

Event Channel

☒ Action Enabled (AE)

Action Name (AN): STA_ALARM

Action Activation (AA): 0 - Alarm

Threshold (TH): 0.000

☒ Action at First Update (AF)

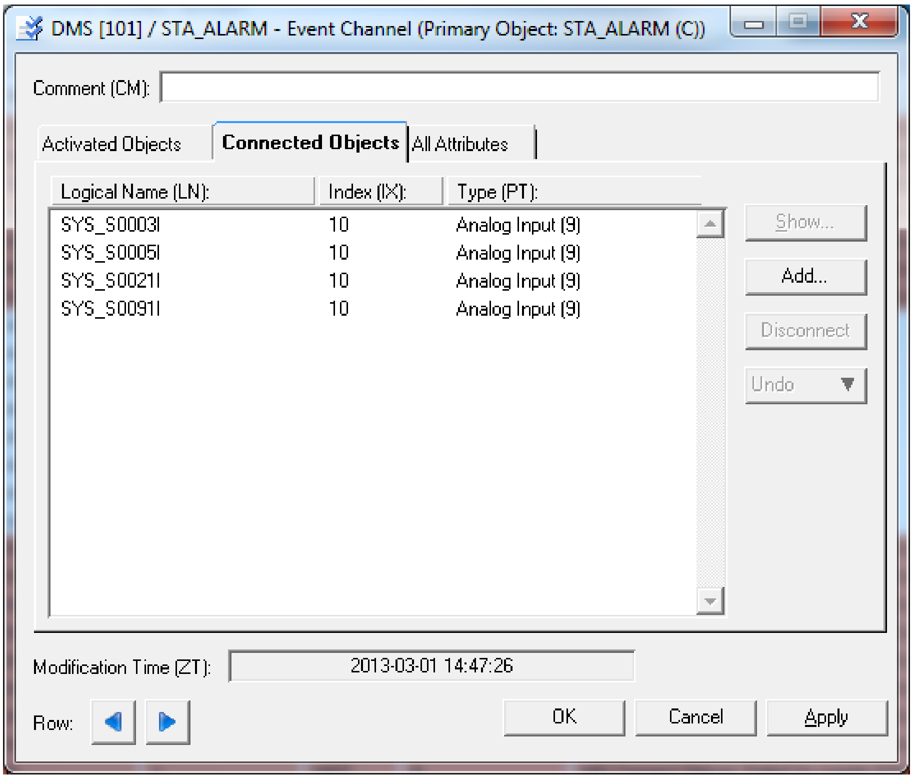
☐ Action on History Values (AH)

☒ Event Object Enabled (EE)

Modification Time (ZT): 2013-03-06 12:35:56

Figure 13. Linking process object to event channel

The linked objects are listed in “Connected Objects” tab of the event channel, as illustrated in Figure 14.



Comment (CM):

Logical Name (LN):	Index (IX):	Type (PT):
SYS_S0003I	10	Analog Input (9)
SYS_S0005I	10	Analog Input (9)
SYS_S0021I	10	Analog Input (9)
SYS_S0091I	10	Analog Input (9)

Modification Time (ZT): 2013-03-01 14:47:26

Figure 14. Connected objects of the event channel

7.3.4 Creating a Command Procedure for Report Creator

Now the process objects are linked to the event and command procedure and every communication alarm activates a program that counts them for each of the stations.

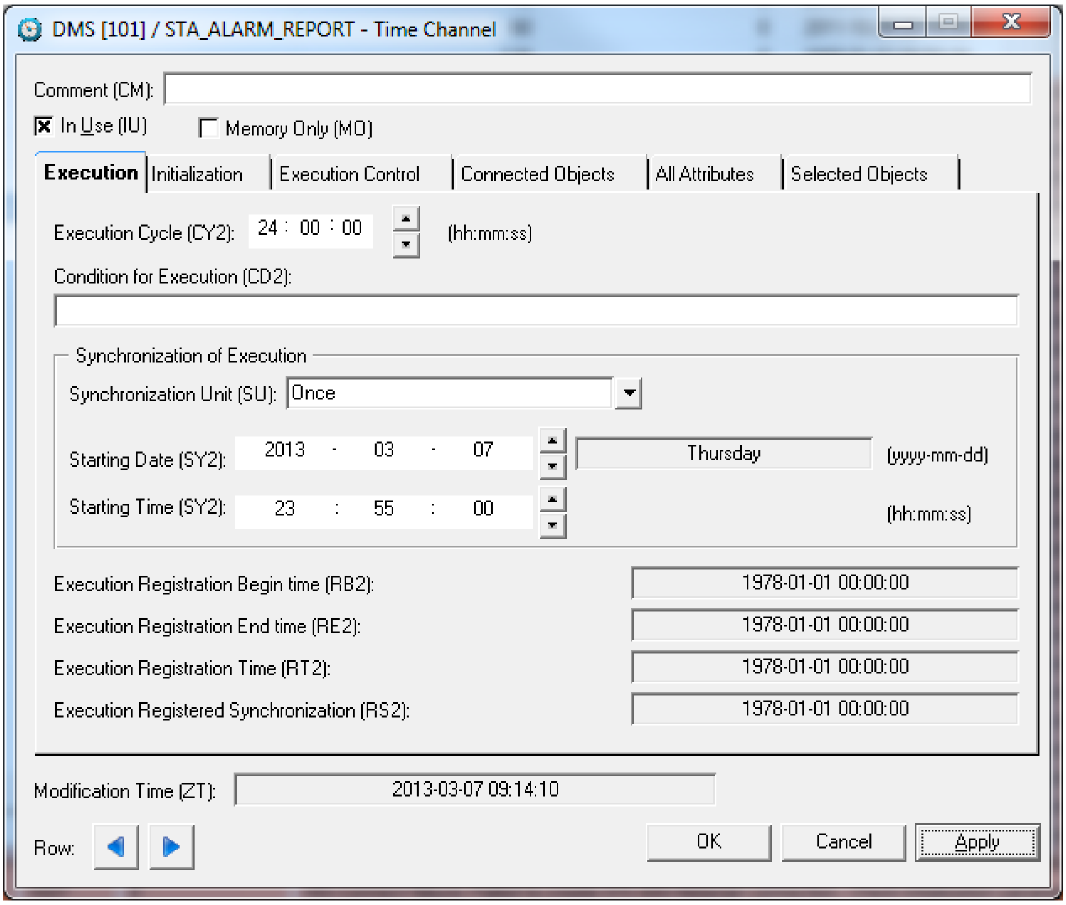
The next step is creating a report of these alarms. A new command procedure “STA_ALARM_REPORT” should be created in the Object Navigator. In the “Instruction (IN)” field, the code presented in Appendix 2 should be inserted.

This code creates a report of communication alarms in the stations and saves it as a text file in the root of the MicroSCADA folder. The report is only created if it is currently first day of the month. This is made because “STATION_ALARM_REPORT” command procedure is run daily, but the report is needed to be created only once a month. The program goes through every process object with the logical name of “SYS_S*I” and index of 10. It writes their logical name, index, object states, object text and number of alarms occurred during one month. After writing the number of alarms, the counter is reset to zero. At the end of the report, the program writes an explanation text that includes the creation date of the report.

7.3.5 Creating a Time Channel

The report is now created and what is left is configuring MicroSCADA so that the report is created automatically. This can be done by using time channel feature in MicroSCADA. With time channels it is possible to execute command procedures in cycles.

A new time channel “STA_ALARM_REPORT” should be created from the Object Navigator. The interval between executions is determined in “Execution Cycle (CY2)” field. The longest cycle time is only 99 hours. This is why the report creating command procedure checked the day of the month at the beginning of the program. 24 hours should be used, so that the execution happens every day. With synchronization settings it is possible to determine the starting time of the cycle, so that the executions happen at the same time every day. An example of the settings can be seen in Figure 15.



DMS [101] / STA_ALARM_REPORT - Time Channel

Comment (CM):

☒ In Use (IU) ☐ Memory Only (MO)

Execution Cycle (CY2): 24 : 00 : 00 (hh:mm:ss)

Condition for Execution (CD2):

Synchronization of Execution

Synchronization Unit (SU): Once

Starting Date (SY2): 2013 - 03 - 07 Thursday (yyyy-mm-dd)

Starting Time (SY2): 23 : 55 : 00 (hh:mm:ss)

Execution Registration Begin time (RB2): 1978-01-01 00:00:00

Execution Registration End time (RE2): 1978-01-01 00:00:00

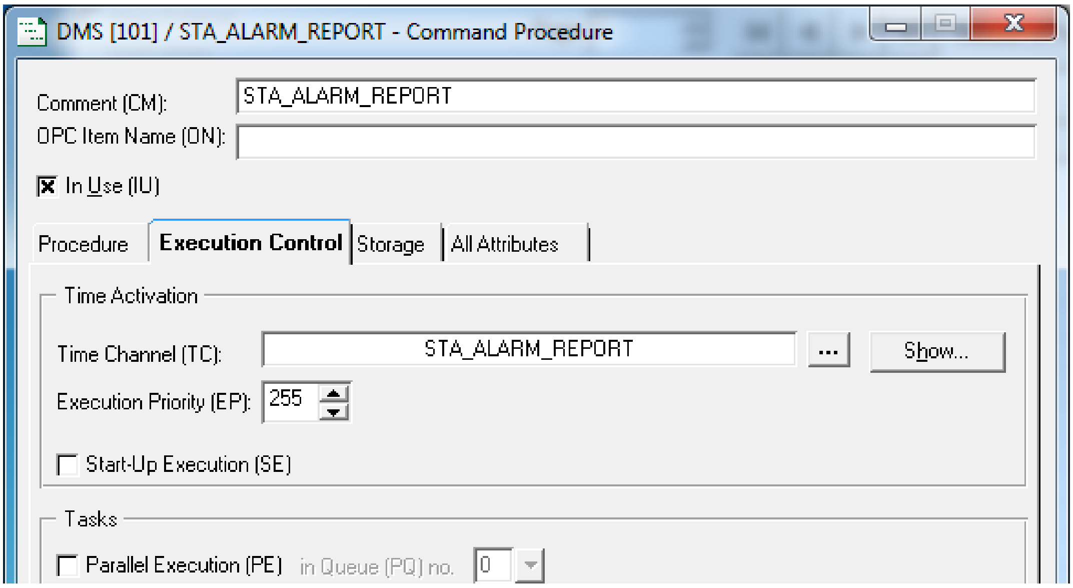
Execution Registration Time (RT2): 1978-01-01 00:00:00

Execution Registered Synchronization (RS2): 1978-01-01 00:00:00

Modification Time (ZT): 2013-03-07 09:14:10

Figure 15. Time Channel settings

Connecting a time channel with the command procedure is configured from "Execution Control" tab of the command procedure. The name of the desired time channel can be written in "Time Channel (TC)" field or it can be selected from the list. This is illustrated in Figure 16.



DMS [101] / STA_ALARM_REPORT - Command Procedure

Comment (CM): STA_ALARM_REPORT

OPC Item Name (ON):

☒ In Use (IU)

Time Activation

Time Channel (TC): STA_ALARM_REPORT

Execution Priority (EP): 255

☐ Start-Up Execution (SE)

Tasks

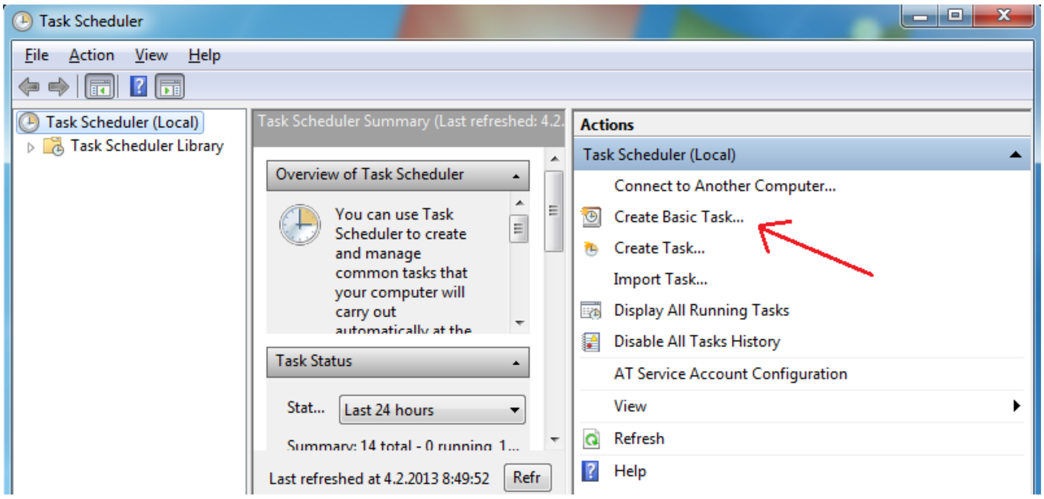
☐ Parallel Execution (PE) in Queue (PQ) no. 0

Figure 16. Connecting Time Channel to Command Procedure

The connected command procedures should now be seen in “Connected Objects” tab of the time channel. The station alarm counter is now completely set up.

7.4 Sending Scheduled E-mail Reports

Sending a scheduled e-mail can be done using the same batch file that launches the System Info Logging tool and Windows Task Scheduler. It is a fairly simple task. The command for SendEmail program described in the previous chapter should be put on the very last line in the batch file. First Windows Task Scheduler has to be opened. Creating a basic task from Actions is as in Figure 17 below.


Task Scheduler
File Action View Help
Task Scheduler (Local)
Task Scheduler Library
Task Scheduler Summary (Last refreshed: 4.2.
Overview of Task Scheduler
Task Status
Stat... Last 24 hours
Summary: 14 total - 0 running 1...
Last refreshed at 4.2.2013 8:49:52 Refr
Actions
Task Scheduler (Local)
Connect to Another Computer...
Create Basic Task...
Create Task...
Import Task...
Display All Running Tasks
Disable All Tasks History
AT Service Account Configuration
View
Refresh
Help

You can use Task Scheduler to create and manage common tasks that your computer will carry out automatically at the

Figure 17. Creating a basic task

A wizard opens a window in which a name to the schedule is given. Giving the name is mandatory. Next the desired trigger is chosen. A calendar schedule can be assigned for the e-mails to be sent by choosing daily, weekly or monthly. After that it is chosen when the schedule is started and when the task should be performed.

"Start a Program" is selected on the next screen and it opens a screen where the program to run is chosen. Enter the path where the batch file is located. After that the task is scheduled and the e-mail will be sent to the chosen address when it was scheduled to.

7.5 The Outcome of the Application

When everything is configured as it was described in this chapter, Windows Task Scheduler launches the MaintenanceReport.bat file at the specified time. The batch file first launches the MaintenenceReport.exe which collects all the information from the MicroSCADA and the operating system. If the station alarm counter is configured, MicroSCADA has created a list of station alarms. After that, the list of active alarms is created and all the information is added to the report file.

When the report file is ready, the batch file uses SendEmail.exe to send the file to the specified e-mail address with determined subject and message.

8 CONCLUSIONS

Predictive maintenance in MicroSCADA systems can be implemented by gathering information from the MicroSCADA program and the operating system and sending this information automatically to the people responsible for the maintenance support. This can be made by using a modified version of System Info Logging tool that is included with the new versions of MicroSCADA and some fairly simple pieces of code. Sending scheduled e-mail reports is easiest to handle with the basic task scheduler of the Windows operating system. This is also recommended, because gathering the information should be made by a program that is not requiring a running MicroSCADA service to work.

Some information that was mentioned useful by the maintenance personnel is not yet included with the report e.g. reading the temperature information from the sensors can be only made by using a third party software. However, if some information is considered interesting, it is easy to modify the created batch file and add own information to the report file. The SCIL command procedures created for this thesis can also be used as a basis to gather different information from MicroSCADA. It is also possible to add different actions to start together with the reporting program, like performing a virus scan, checking hard disk condition or creating a system backup.

REFERENCES

/1/ ABB in Brief Accessed 13.1.2013. http://new.abb.com/about/abb-in-brief

/2/ Command-line reference A-Z Accessed 25.2.2013. http://technet.microsoft.com/en-us/library/bb490890.aspx

/3/ DPS Telecom SCADA Introduction Accessed 13.1.2013. http://www.dpstele.com/dpsnews/techinfo/scada/scada_knowledge_base.php

/4/ Email Security - What Are the Issues? Accessed 18.1.2013. http://www.net-security.org/article.php?id=816

/5/ Features of GFI MAX RemoteManagement Accessed 19.2.2013. http://www.gfimax.com/remote-management/features

/6/ Introduction to Windows Service Accessed 7.2.2013. http://msdn.microsoft.com/en-us/library/d56de412(v=vs.80).aspx

/7/ MicroSCADA Maintenance Manual. ABB.

/8/ MicroSCADA Pro SYS 600 9.3 Application Design. ABB.

/9/ MicroSCADA Pro SYS 600 9.3 Cyber Security Deployment Guideline. ABB.

/10/ MicroSCADA Pro SYS 600 9.3 Programming Language SCIL. ABB.

/11/ Mobley, R. K. 2002. An Introduction to Predictive Maintenance. Second Edition. USA. Butterworth-Heinemann.

/12/ Network Working Group. 2002. An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks Accessed 19.3.2013. http://tools.ietf.org/html/rfc3411

/13/ PA Server Monitor Feature List Accessed 19.2.2013. http://www.poweradmin.com/ServerMonitor/features.aspx

/14/ SCADA: Solutions Centre Accessed 13.1.2013. http://scada.atspace.com/

/15/ System Information for Windows Accessed 22.2.2013. http://www.gtopala.com/

/16/ The Birth of MicroSCADA Accessed 16.1.2013.

http://www05.abb.com/global/scot/scot229.nsf/veritydisplay/5e4989b705a357b9c2256d3a0025ecbe/$File/BirthOfMicroSCADA.pdf

/17/ Using batch files Accessed 25.2.2013. http://www.microsoft.com/resources/documentation/windows/xp/all/proddocs/en-us/batch.mspx?mfr=true

/18/ Windows registry information for advanced users Accessed 31.1.2013. http://support.microsoft.com/kb/256986/en-us

APPENDICES

Appendix 1 Batch File Commands

```
\sc\prog\exec\MaintenanceReport.exe
scilc.exe -do AlarmList.scil
echo y | del \sc\temp\MaintenanceReport.zip
ren                    \sc\temp\SendMeToProductSupportCenter_*.zip
MaintenanceReport.zip
zip -r -q \sc\temp\MaintenanceReport.zip \sc\Alarms.txt
zip -r -q \sc\temp\MaintenanceReport.zip \sc\Sta_Alarms.txt
sendemail  -f  sender@mail.com  -t  receiver@mail.com  -u  Subject  -m
Message -a MaintenanceReport.zip -s smtp.mail.com:123 -xu username
-xp password -o tls=yes
```

Appendix 2 SCIL Code for Alarm Counter

```
#local count='ln':pfi'ix', al = 'ln':pal'ix'
#IF al == 1 #then #block
   count = count + 1
   #set 'ln':pfi'ix' = count
#block_end
```

Appendix 3 SCIL Code for Alarm Counter Report

```
#local ln="", ix=0, report_txt, w, al_obj, i, drive="d"
#local ln_v, ix_v, ox_v, counter, index_v, ox, fi

report_txt = vector(report_txt)

al_obj = APPLICATION_OBJECT_LIST(0, "IX", "OI", "F", "*", "LN ==
""SYS_S*I"" AND IX==10" , "",100)

#if DAY(clock) == 1 #then #block
   #if al_obj.count > 1 #then #block
      index_v = sort(al_obj.LN)
      ln_v = pick(al_obj.ln,index_v)
      ix_v = pick(al_obj.ix,index_v)

      report_txt = append(report_txt, vector("Number of alarms in
each
      station:"))
      report_txt = append(report_txt, vector(""))
      report_txt = append(report_txt, vector("LN:IX, OX, Number of
      alarms"))

      #loop_with i=1..al_obj.count
         ln = ln_v(i)
         ix = ix_v(i)
         ox = 'ln':POX'ix'
         fi = 'ln':PFI'ix'
         report_txt = append(report_txt, vector(ln + ":" + "'ix'"
+ ", "
        + ox + ", " + "'fi'"))
         #set 'ln':pfi'ix' = 0
      #loop_end
   #block_end

   report_txt   =   append(report_txt,   vector("",   "This   is   an
automatic
   message   generated   "+date("full")+"   by   the   "+apl:bna+"
MicroSCADA
   application"))
   drive    =    substr(PARSE_FILE_NAME("PICT"),1,1);check    scada
installation
   drive
   w = write_text(drive + ":\sc\Sta_Alarms.txt", report_txt, 0)
#block_end
```