Implementation of OPC server for SNMP network monitoring on FACTS equipment

Kim Strömberg
School of Innovation, Design and Engineering
Mälardalen University

Supervisor ABB/FACTS: Gabor Stein
Supervisor Mälardalen University: Robert Suurna
Examiner Mälardalen University: Mats Björkman

2013-05-24
Abstract

The goal of this thesis work was to implement a functional OPC server monitoring network devices in ABB/FACTS applications through the SNMP protocol. The work was divided into four different parts, analysis, design, implementation and verification. Different OPC server source codes and different SNMP libraries were analysed. After completing the analysis, Advosol source code and an SNMP library by Oliver Griffet was chosen. The design included determining which SNMP information was relevant. By choosing the relevant information in an early stage, the implementation phase was programming and done in a Visual Studio environment in C#. During the implementation phase the author also got familiarized with InTouch for the HMI window where the SNMP information was shown. A specific SNMP information window was also created in the user interface within InTouch. By implementing the SNMP library within the OPC server the author was able to demonstrate a working application concept at the end of the thesis work.
I lovingly dedicate this thesis to my mother and father for their unconditional love and support throughout my life.
Acknowledgements

This dissertation would not have been possible without the help of my supervisor at ABB/FACTS Gabor Stein, team leader Daniel Gustavsson and et al. at ABB/FACTS/TC for helping me and giving me the opportunity to do my thesis work.

Furthermore I would like to express my thanks to the whole of Netcenter at Mälardalen University for their continuous support throughout my B.Sc. studies. A special thanks to my parents for all their support throughout the years while pursuing my B.Sc. degree with all from math support and guidance to proof-reading this thesis.

Finally, I want to thank Maria for all her support.
## List of abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AVS</td>
<td>Anti-Virus Staging area</td>
</tr>
<tr>
<td>COM</td>
<td>Component Object Model</td>
</tr>
<tr>
<td>DCOM</td>
<td>Distributed Component Object Model</td>
</tr>
<tr>
<td>DMZ</td>
<td>Demilitarized Zone</td>
</tr>
<tr>
<td>FACTS</td>
<td>Flexible AC Transmission Systems</td>
</tr>
<tr>
<td>FTS</td>
<td>File Transfer Server</td>
</tr>
<tr>
<td>GWS</td>
<td>Gateway Station</td>
</tr>
<tr>
<td>HMI</td>
<td>Human Machine Interface</td>
</tr>
<tr>
<td>IETF</td>
<td>Internet Engineering Task Force</td>
</tr>
<tr>
<td>IGBT</td>
<td>Insulated Gate Bipolar Transistor</td>
</tr>
<tr>
<td>LAN</td>
<td>Local Area Network</td>
</tr>
<tr>
<td>Mb</td>
<td>Megabit</td>
</tr>
<tr>
<td>Mbps</td>
<td>Megabits per second</td>
</tr>
<tr>
<td>MB</td>
<td>Mega Byte</td>
</tr>
<tr>
<td>MBps</td>
<td>Mega Bytes per second</td>
</tr>
<tr>
<td>MIB</td>
<td>Management Information Base</td>
</tr>
<tr>
<td>NMS</td>
<td>Network Management Station</td>
</tr>
<tr>
<td>NTP</td>
<td>Network Time Protocol</td>
</tr>
<tr>
<td>NTS</td>
<td>Network Time Server</td>
</tr>
<tr>
<td>OID</td>
<td>Object Identifier</td>
</tr>
<tr>
<td>OLE</td>
<td>Object Linking and Embedding</td>
</tr>
<tr>
<td>OPC</td>
<td>OLE for process control</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------</td>
</tr>
<tr>
<td>OPC-DA</td>
<td>OLE for process control – Data Access</td>
</tr>
<tr>
<td>OWS</td>
<td>Operator Work Station</td>
</tr>
<tr>
<td>SVC</td>
<td>Static Var Compensation</td>
</tr>
<tr>
<td>SNMP</td>
<td>Simple Network Management Protocol</td>
</tr>
<tr>
<td>TC</td>
<td>Technical Control-Design</td>
</tr>
<tr>
<td>UDP</td>
<td>User Datagram Protocol</td>
</tr>
</tbody>
</table>
# Contents

Abstract .......................................................................................................................................................... ii

English .......................................................................................................................................................... ii

Svenska ........................................................................................................................................................ iii

Acknowledgements ........................................................................................................................................ v

List of abbreviations .................................................................................................................................... vi

Chapter 1: Introduction .................................................................................................................................. 1

1.1 Background .............................................................................................................................................. 1

1.1.1 The FACTS technology and ABB ........................................................................................................ 1

1.1.2 Technical Control Design .................................................................................................................... 3

1.1.3 Monitoring ............................................................................................................................................ 5

1.2 Purpose .................................................................................................................................................... 5

1.3 Delimitations .......................................................................................................................................... 5

1.4 Related work ........................................................................................................................................... 6

1.5 Problem formulation ............................................................................................................................... 6

Chapter 2: Theory .......................................................................................................................................... 7

2.1 OPC ......................................................................................................................................................... 7

2.2 SNMP ..................................................................................................................................................... 8

2.2.1 Versions ............................................................................................................................................... 8

2.2.1 MIB & OID ......................................................................................................................................... 9

2.2.2 How SNMP works ............................................................................................................................... 11

Chapter 3: Analysis ...................................................................................................................................... 12

3.1 SNMP .................................................................................................................................................... 12

3.1.1 Support in devices ............................................................................................................................... 12

3.1.2 Relevant SNMP information: ............................................................................................................. 12

3.1.3 Impact on network performance ......................................................................................................... 13
3.1.4 SNMP Manager ........................................................................................................... 13
3.1.5 Licensing ......................................................................................................................... 14
3.2 OPC server ......................................................................................................................... 15

Chapter 4: Method .................................................................................................................... 16
4.1 Analysis ................................................................................................................................. 16
4.2 Design: ................................................................................................................................. 16
4.3 Implementation ..................................................................................................................... 16
4.4 Verification ............................................................................................................................. 17

Chapter 5: Solution ..................................................................................................................... 18
5.1 Conclusion of analysis .......................................................................................................... 18
5.2 OPC server .......................................................................................................................... 19
5.3 SNMP manager .................................................................................................................... 20
5.4 Integration of SNMP in OPC server ................................................................................... 21
5.5 Test page for the HMI (InTouch) ....................................................................................... 23
5.6 Verification of connectivity ................................................................................................. 24
5.7 Displaying SNMP information ............................................................................................ 25

Chapter 6: Results ...................................................................................................................... 26
6.1 Requirements ....................................................................................................................... 26
6.2 Future work ........................................................................................................................... 27

Chapter 7: Conclusion ............................................................................................................... 28

Bibliography .............................................................................................................................. 31
Chapter 1: Introduction

This thesis represents the final part of a Bachelor Program in Computer Science - Network Engineering. It was conducted at ABB/FACTS in Västerås. The ABB/FACTS division is the leading developer of Flexible AC Transmission Systems technology.

1.1 Background

During the last years of developing power systems ABB/FACTS have seen an increased number of network components and complexity of the communication systems. This rapid development has created the need for tools to monitor the network devices in a FACTS plant.

1.1.1 The FACTS technology and ABB

Modern electric power as we know it today was born in the 1880s when three-phase alternating current was introduced. This made it possible to use high voltage and distribute the electric power several kilometres instead of producing electricity in our neighbourhood. To use the power in homes, the voltage needs to be transformed to a low voltage level.

To utilize the power system more efficient a new technology, FACTS, was developed. The FACTS technology was first introduced in 1950 by ABB. The FACTS technology consists of two parts, series compensation and dynamic shunt compensation.
Chapter 1: Introduction

Series compensation was introduced first in 1950. It made it possible to distribute electricity over vast distances. A capacitor is connected in the power transmission line to compensate the line inductance, this makes the distance between power plants and feeding stations seem electrically shorter. Series compensation made it possible to transfer at higher voltage levels than before. Series compensation stabilized the systems and minimized energy losses.

With distributed electricity another problem arose. Industrial factories caused fluctuation voltage levels throughout the day depending on the need at the factories. To compensate for this fluctuation ABB developed in 1970 a thyristor based technology for high voltage, the SVC (Static Var Compensation). When the SVC was introduced there was an immediate stabilization of the electrical voltage and power quality for consumers near the factories. These SVCs work like a shock absorber. If there is a disturbance in the voltage level the SVC compensates this to minimize voltage drops and spikes.¹

ABB continued to develop the SVCs and in 1979 it was introduced to railroad systems, 1996 a mobile SVC and in 1998 SVC light, based on IGBT (Insulated Gate Bipolar Transistor) was introduced.²

During the last 50 years ABB have done more than 600 FACTS projects throughout the world.

---

¹ ABB Power Systems, FACTS, 2013
² ABB AB, 2013
1.1.2 Technical Control Design

A subdivision of ABB/FACTS is TC (Technical Control Design). It is TCs job to implement and design the control systems that are supplied to the customers. These control systems are the interface between customers and the FACTS plants and it is here ABB/FACTS would like a SNMP monitoring functionality.

The control system consists of:

- OWS (Operator Work Station), internal workstation inside the network.
- GWS (Gateway Station), gateway on a DMZ (Demilitarized Zone) for communication with customers LAN (Local Area Network).
- Router/Firewall
- FTS (File Transfer Server), file server in a DMZ for transferring between internal network and customer LAN
- AVS (Anti-Virus Staging area), usually on the same station as FTS for Virus search.
- RWS (Remote Work Station)
- NTS (Network Time System), time server, works as a NTP (Network Time Protocol) server
- Switch
- MAIN, control computer for the control system.
1.1 Background

Chapter 1: Introduction

Fig 1. Diagram of Control System

The GWS, FTS/AVS and RWS are connected to a firewall and are put on DMZ so they are not directly connected to the internal control network. From the firewalls the customers LAN is connected with access rules depending on what they are to access on the internal control network. From the firewalls internal network switches are connected and to those the OWS, 3GH modem, NTS and MAIN computers are connected.
1.3 Delimitations

The expectations are to be able to demonstrate the application on a conceptual level.
1.4 Related work

Since the OPC-DA protocol is a standard in industrial automation several companies have made similar work implementing SNMP support in OPC-DA servers. These include:

- OPC Server for SNMP Devices by MatrikonPC.3
- SNMP OPC Server by Kepware Technologies.4
- EDS-SNMP OPC Server Pro by Moxa.5
- SNMP-OPC Gateway by Obermeier Software.6
- SNMP OPC server by Siemens.7
- SNMP-OPC Server by Chipkin Automation Systems.8
- OPC-Netlistener by Somebytes.9

All of these applications can be bought with licensing and implemented in ABB/FACTS control systems. This would add costs to each project. The thesis will demonstrate how a similar application is built for ABB/FACTS.

1.5 Problem formulation

ABB/FACTS include multiple network devices in their control systems. Today no information about these devices is displayed on the HMI. The task is to implement SNMP functionally in an OPC server that can communicate with the HMI and create a test window in the HMI to display network information.

---

3 Matrikon Inc, 2013
4 Kepware Technologies, 2013
5 Moxa, 2013
6 Obermeier Software, 2013
7 Siemens, 2013
8 Chipkin Automation Systems, 2013
9 Somebytes, 2013
Chapter 2: Theory

2.1 OPC

OPC is a standard defined by the OPC Foundation for communication of real-time data between controllers from different manufacturers. From the beginning OPC Foundations single standard specification was the OPC Specification which is now called Data Access Specification.

Before the OPC standard became widely used each company had to write their own Human Machine Interface software and a proprietary driver for each industrial device. Because of this the first OPC specification was created to simplify driver implementations in software.

The OPC specification was based on OLE, COM and DCOM technologies from Microsoft. It was created by the leading automation suppliers working in cooperation with Microsoft. With this the software suppliers could concentrate on features instead of driver implementations. This also let users choose software by features instead of looking at if the supplier supported their devices.

So the first specification standardized the acquisition of process data. By doing this it was soon realized that other types of data also could benefit from standardization.\textsuperscript{10}

\textsuperscript{10} OPC Foundation, 2013
Today the OPC specifications include:

- OPC Data Access
- OPC Alarms & Events
- OPC Batch
- OPC Data eXchange
- OPC Historical Data Access
- OPC Security
- OPC XML-DA
- OPC Complex Data
- OPC Unified Architecture

### 2.2 SNMP

SNMP (Simple Network Management Protocol) is used to manage network devices, which also include Windows/Unix systems, printers, power supplies and more. By using the SNMP protocol an administrator is able to edit settings, pull information and sense device changes.

SNMP communicate between Agents and Managers, where Agents are software on network devices such as routers, switched or firewalls. The Manager is a server that runs software that can handle management task for the network. These servers are often referred to as a NMS (Network Management Station).\(^\text{11}\)

#### 2.2.1 Versions

There are three different versions of SNMP available.

SNMP Version 1 (SNMPv1) was the first version released by The Internet Engineering Task Force (IETF) whom is responsible for defining the standard protocols that are used for Internet traffic. Then there is SNMP version 2c (SNMPv2c). It was introduced to include more packet types then SNMPv1. Like

\(^\text{11}\) Mauro et al., 2001
the GETBULK packet which make it possible to request a large number of GET in one single packet. Both SNMPv1 and SNMPv2c use communities for security. These communities are like passwords that authenticate the SNMP packets. The community string is sent in every SNMP packet in plain text and the Manager uses it to decide to either accept it or discard it. There are usually different community strings for read and read/write access. This is the only security feature in SNMPv1 and SNMPv2c.

SNMP version 3 (SNMPv3) was introduced to address the weak security of SNMPv1 and SNMPv2c. It is based on SNMPv2c but does not use community strings for authentication. Instead of the community strings it use users with passwords. SNMPv3 also allows for authentication and encryption of the SNMPv3 packet. The level of security depends on how the user is defined. With SNMPv3 there is also an ability to limit how much of the MIB a user can access. Despite the introduction of SNMPv3 with more security SNMPv2c is still more common and not all manufacturers support SNMPv3 yet.\textsuperscript{12}

### 2.2.1 MIB & OID

The Management Information Base (MIB) consists of a database with Object Identifiers (OID) that is organized in a tree structure. These OIDs are the objects that each Agent tracks and specifies what information it is after. Each OID has a numerical representation and a text name. The numerical representation is made up of a series of integers with a dot (.) separator. Each integer represents a level in the hierarchical tree structure.

For system name, 1.3.6.1.2.1.1.5.0 would be specified. The 1.3.5.1.2.1.1 would specify the system sub-tree and the last 5.0 specifies the specific OID and

\textsuperscript{12} Mauro et al., 2001
instance. The text name representation for system name would be
iso.org.dod.internet.mgmt.mib-2.system.sysName.0.

By using 1.3.6.1.2.1.2.2.1.2 instead interface description would be displayed.
This OID could use multiple instances, one for each interface. The text name
representation for the interface description would be
iso.or.dod.internet.mgmt.mib-2.interfaces.ifTable.ifEntry.ifDescr.13

---

13 Mauro et al., 2001
2.2 How SNMP works.

The manager uses a get request to fetch the information from the Agent. This request defines the information type.

A typical SNMP Get command looks as follow:

```
Snmpget -v2c public 127.0.0.1 1.3.6.1.2.1.1.5.0
```

- **Snmpget** – specified what you want to do. (Request SNMP information)
- **-v2c** – specified which SNMP version is used. (v1, v2c, v3)
- **public** – is a community string that is specified on the network device. It is like a password to the device information. Public is usually specified for read access only.
- **127.0.0.1** – IP address of the device.
- **1.3.6.1.2.1.1.5.0** – an OID, in this case system name for a computer.

After the call the device will answer the Manager with the requested information.
Chapter 3: Analysis

3.1 SNMP

3.1.1 Support in devices

There are multiple network devices that ABB/FACTS use and these were tested to see what SNMP information they could provide.

The tested network devices are:

- Sonicwall TZ170 (Firewall)
- Sonicwall TZ210 (Firewall)
- Phoenix Contact 8XT (Switch)
- RuggedCom RS900 (Switch)
- Meinberg Lantime M600 (GPS clock)

Each of the tested devices has the possibility to enable SNMP for monitoring. Within these five different devices there are only two that support some sort of link speed and duplex parameter monitoring. Meinberg Lantime M600 and Phoenix Contact 8XT. On Meinberg Lantime M600 one can retrieve five different values: 0=auto, 1=half10, 2=full10, 3=half100 och 4=full100. The Phoenix contact 8XT can only display Full or Half.

3.1.2 Relevant SNMP information:

By analysing the need in the HMI and having a dialog with ABB/FACTS it has been concluded that a final application should have the ability to fetch the information displayed in the list below.

- Interface status
- Interface speed
- System description
3.1.3 Impact on network performance

Usually SNMP is applied over UDP (User Datagram Protocol) which is unreliable. Being unreliable it makes very little overhead. SNMP datagrams are relatively small but it will still generate a lot of traffic if a lot of requests are made.\(^\text{14}\) Each SNMP datagram has a maximum size of 1.5KB. If monitoring five devices with five different values each. Doing this each 30 seconds, there is a bandwidth need of \((1.5\text{KB} \times 5 \times 5 \times 2 / 60)\) \(1.25\text{KB/s}\). The daily amount of bandwidth required will sum up to \((1.25\text{KB/s} \times 86400 \text{ seconds} / 1024)\) \(105\text{MB}\). If the oldest cable in the control system is a category 5 cable, capable of supporting 100Mbps. Then the total amount of data possible to transfer over the cables in a day would be \((100\text{Mbps} \times 86400 \text{ seconds} / 8)\) \(1080000\text{MB}\).

3.1.4 SNMP Manager

There are multiple C# SNMP libraries that are widely available under different licences. The list below is the main libraries that will be analysed and tested for implementation in the OPC server.

- SNMP library\(^\text{15}\)
- C# Based Open Source SNMP for .Net and Mono\(^\text{16}\)
- An SNMP Library for .Net Framework\(^\text{17}\)

\(^{14}\) Mauro et al., 2001
\(^{15}\) Griffet, 2006
\(^{16}\) Crowe et al., 2013
\(^{17}\) Crowe, 2002
3.1.5 Licencing

To be able to distribute a working application with ABB/FACTS equipment there is a need to use a licensing model that provides the flexibility to be distributed without the need to provide ABB/FACTS source code. The three tested SNMP libraries have different licensing rules.

The SNMP Library uses a GPOL License which is free to use with some restrictions:

“3. License Grant. Subject to the terms and conditions of this License, the Author hereby grants You a worldwide, royalty-free, non-exclusive, perpetual (for the duration of the applicable copyright) license to exercise the rights in the Work as stated below:

a. You may use the standard version of the Source Code or Executable Files in Your own applications.

b. You may distribute the standard version of the Executable Files and Source Code or Derivative Work in aggregate with other (possibly commercial) programs as part of a larger (possibly commercial) software distribution. Any subroutines or modules supplied by You and linked into the Source Code or Executable Files of this Work shall not be considered part of this Work and will not be subject to the terms of this License.”¹⁸

The C# Based Open Source SNMP for .Net and Mono has a GPL License. The GPL License use copyleft which means that the GPL software shall remain free and open even when redistributed with other applications.

“You may convey a work based on the Program, or the modifications to produce it from the Program...

... A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an “aggregate” if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation’s users beyond what the individual works permit. Inclusion of a covered work in an

¹⁸ Codeproject, 2013
aggregate does not cause this License to apply to the other parts of the aggregate.”

An SNMP Library for .Net Framework library doesn’t have any Licensing tags within the code at all.

### 3.2 OPC server

OPC foundation supplies a sample code to registered members which is built up by a C++ wrapper and a C# server implementation. These parts can be used to implement an OPC server and can be modified for custom applications. During initial testing of the OPC foundation samples there were some problems with the C++ wrappers and compiling the code.

Tests were also made on another sample code from Advosol. Advosol uses a program bundle with source code and client programs for testing OPC servers. The code is built on the OPC foundation code but removes the need to edit the C++ wrapper and uses a wizard to create a sample code for custom servers.
Chapter 4: Method

The thesis work is divided into four different parts, analysis, design, implementation and verification. By dividing the work it is easy to know when each part is finished.

4.1 Analysis

First each network device was analysed and their respective manual was read to see if they support SNMP and if they did how much information was available. A list was then produced with the relevant information to be included in the OPC application. Research was also conducted to find out how much overhead SNMP send to conclude if it would have any impact on ABB/FACTS control systems network performance.

4.2 Design:

Each OPC source code was analysed to see which of them would work best for ABB/FACTS need. This was to make sure only C# implementations were needed. Research for C# SNMP libraries was made and the code and licencing was evaluated.

4.3 Implementation

During the implementation the chosen SNMP library was integrated with the OPC source code to a working application. The server was tested with the included test clients. When the application was working and tested a test page in InTouch was created for the HMI, where all the information is displayed.
4.4 Verification

When the implementation was complete and the test pages were done the application went through a verification phase where it was tested against multiple network devices.

After the application was tested and working the OPC SNMP concept was demonstrated to the staff at ABB/FACTS.
Chapter 5: Solution

This part of the thesis consists of the solution and implemented part of the thesis work done at ABB/FACTS.

5.1 Conclusion of analysis

After the analysis of SNMP support on the provided devices SNMP version 2c was chosen. To illustrate the concept basic interface information will be fetched with SNMP and displayed on the HMI.

After trying both OPC Foundations and Advosols source code samples it is decided to use Advosols code since it only requires C# implementation. This was a key part on the project specification.

All of the provided SNMP libraries that are available are working fine. To minimize the programming load the SNMP library by Oliver Griffet was chosen. After evaluating the licensing it is the author interpretation that Griffets code is easier to use for redistributing ABB/FACTS software.

By analysing the need for SNMP in ABB/FACTS equipment it is clear that only a limited number of requests will be made at any given time. Given that fact it is unlikely that SNMP will have any sort of impact on the overall network performance on ABB/FACTS remaining equipment what so ever.
5.2 OPC server

The OPC server is implemented with the source code provided by Advosol. By running their wizard a skeleton is made from where it is possible to construct a custom application.

The OPC server is implemented with the source code provided by Advosol. By running their wizard a skeleton is made from where it is possible to construct a custom application.

![Advosol Wizard](image)

Fig 3. Advosol Wizard.

The Visual Studio project is created and consists of:

- **DANSrvNet4.exe.config**
  - Configurations file which consists of server registry definitions that are used to register and then contact the server.

- **Definitions.cs**
  - Definitions used by the program.

- **IGeneric.cs**
  - Defines the generic server interface. Contains definitions, callback methods and default implementations.

- **ServerAdapt.cs**
  - Contains the methods that are called by the generic server.

- **AssemblyInfo.cs**

- **References**

- **ReadMe.txt**
5.3 SNMP manager

The SNMP library consists of six different dll files that are used as reference with the custom application.

- Mib.dll
- Snmp.dll
- SnmpComp.dll
- SNMDll.dll
- TableReader.dll
- Tools.dll

With this library there are functions defined that makes it possible to define SNMP objects with OIDs and SNMP Agents with their respective IP-addresses.

First define an SNMP object:

- SNMPObject objectoid = new SNMPObject("1.3.6.1.2.1.1.5.0");

Then define a request from an SNMP Agent:

- string SNMPValue = objectoid.getSimpleValue(new SNMPAgent("127.0.0.1");

By doing this the string SNMPValue will consist of the respons from the Agent which in this case will be the System Description defined by the OID.
5.4 Integration of SNMP in OPC server

To implement the SNMP library in the OPC server sample code the while library containing the dlls are imported into the Visual Studio project. This gives access to all the methods used in the dlls.

First data definitions are defined in the ServerAdapt.cs file.

```csharp
// DATA DEFINITIONS
    static ItemDef[] Items = {
        new ItemDef("SNMP.WANSpeed", (int)0, "1.3.6.1.2.1.2.2.1.5.1", "192.168.168.168"),
        new ItemDef("SNMP.OPTSpeed", (int)0, "1.3.6.1.2.1.2.2.1.5.2", "192.168.168.168"),
        new ItemDef("SNMP.LOOSpeed", (int)0, "1.3.6.1.2.1.2.2.1.5.3", "192.168.168.168"),
        new ItemDef("SNMP.LANSpeed", (int)0, "1.3.6.1.2.1.2.2.1.5.4", "192.168.168.168"),
        new ItemDef("SNMP.WANStatus", (int)0, "1.3.6.1.2.1.2.2.1.8.1", "192.168.168.168"),
        new ItemDef("SNMP.LOOStatus", (int)0, "1.3.6.1.2.1.2.2.1.8.2", "192.168.168.168"),
        new ItemDef("SNMP.OPTStatus", (int)0, "1.3.6.1.2.1.2.2.1.8.3", "192.168.168.168"),
        new ItemDef("SNMP.LANStatus", (int)0, "1.3.6.1.2.1.2.2.1.8.4", "192.168.168.168"),
        new ItemDef("SNMP.WANDescr", (string)"None", "1.3.6.1.2.1.2.2.1.2.1", "192.168.168.168"),
        new ItemDef("SNMP.LANDescr", (string)"None", "1.3.6.1.2.1.2.2.1.2.4", "192.168.168.168"),
        new ItemDef("SNMP.LOODescr", (string)"None", "1.3.6.1.2.1.2.2.1.2.2", "192.168.168.168"),
        new ItemDef("SNMP.OPTDescr", (string)"None", "1.3.6.1.2.1.2.2.1.2.3", "192.168.168.168"),
        new ItemDef("SNMP.SysDescr", (string)"None", "1.3.6.1.2.1.1.5.0", "192.168.168.168"),
        new ItemDef("SNMP.SysDescr1", (string)"None", "1.3.6.1.2.1.1.5.0", "127.0.0.1"),
        new ItemDef("SNMP.LANStatus1", (string)"None", "1.3.6.1.2.1.2.2.1.8.2", "127.0.0.1"),
        new ItemDef("SNMP.LANSpeed1", (string)"None", "1.3.6.1.2.1.2.2.1.5.2", "127.0.0.1"),
        new ItemDef("SNMP.SysMemStorage1", (string)"None", "1.3.6.1.2.1.25.2.3.1.5", "127.0.0.1"),
        new ItemDef("SNMP.SysMemUsed1", (string)"None", "1.3.6.1.2.1.25.2.3.1.6", "127.0.0.1"),
        new ItemDef("SNMP.SysMemAll1", (string)"None", "1.3.6.1.2.1.25.2.3.1.4", "127.0.0.1"),
        new ItemDef("SNMP.SysMemType1", (string)"None", "1.3.6.1.2.1.25.2.3.1.3", "127.0.0.1"),
        new ItemDef("SNMP.LANDescr1", (string)"None", "1.3.6.1.2.1.2.2.1.2.2", "127.0.0.1")};
```

These are defined as STATIC because the ItemDef class is used in multiple instances.

The ItemDef class a specified in the Definitions.cs file.
5.4 Integration of SNMP in OPC server

// Simple Item and value definition
public class ItemDef
{
    public string ID;
    public object Value;
    public string OID;
    public string IPAdd;

    public ItemDef(string id, object val, string oid, string ip)
    {
        ID = id;
        Value = val;
        OID = oid;
        IPAdd = ip;
    }
}

Use a forever loop to loop through all data definitions and depending on which type SNMP will retrieve different values.

for (; ; ) // forever thread loop
{
    // increment readable items of type Int, Short, Float or Double
    for (int i = 0; i < Config.Items.Length; ++i)
    {
        System.Type tp = Config.Items[i].Value.GetType();
        if (tp == typeof(int))
        {
            for (int x = 0; x < IPList.Count; ++x)
            {
                SNMPObject s = new SNMPObject(Config.Items[i].OID);
                int myValue = int.Parse(s.getSimpleValue(new SNMPAgent(Config.Items[i].IPAdd)));
                Config.Items[i].Value = myValue;
            }
        }
        else if (tp == typeof(string))
        {
            for (int x = 0; x < IPList.Count; ++x)
            {
                SNMPObject s = new SNMPObject(Config.Items[i].OID);
                string myValue = s.getSimpleValue(new SNMPAgent(Config.Items[i].IPAdd));
                Config.Items[i].Value = myValue;
            }
        }
    }
    SetItemValue(i, Config.Items[i].Value, (short)OPCQuality.GOOD, DateTime.Now);
}
5.5 Test page for the HMI (InTouch)

InTouch Windowmaker is used to create windows that can show the SNMP information in the HMI. For a test page an SonicWall TZ 170 firewall was used. In figure 3 the test page is displayed before OPC connections are made.

Fig. 4. InTouch test page.
5.6 Verification of connectivity

To verify that the OPC server is able to handle SNMP requests it was first tested with the Advosol test client towards the local computer.

Fig. 5. Advosol client.

After being able to connect to localhost the OPC Server was modified to handle SNMP information for the SonicWall TZ170 and tested with Advosols client.

Fig 6. Advosol client
5.7 Displaying SNMP information

InTouch use tag names to define each element on the windows. These correspond with the data definitions that are defined in the OPC server. InTouch then relays through an InTouch component called OPCLink. OPCLink gives InTouch OPC capabilities so it can communicate with the OPC Server.

InTouch sends a tag name to OPCLink that forwards it to the OPC Server with a protocol called SuiteLink. The OPC server then does a SNMP get request to the targeted device and sends back the reply to OPCLink through SuiteLink whom in turn sends it to the InTouch window.

Figure 7 displays a working concept of the implementation of OPC server for SNMP network monitoring on ABB/FACTS equipment.
Chapter 6: Results

6.1 Requirements

The OPC server is fully functional on a conceptual level as requested by ABB/FACTS. It is able to handle the requests and is displaying the information. The program is built in C# and runs in a Windows environment. The OPC server can communicate with external network devices by sending a SNMP request to the device. The OPC server will then relay the answer to an HMI window through the SuiteLink protocol.

During the initial phase of the thesis there were some considerable problems with running and compiling the source codes from OPC Foundation. After analysing the problem there were concerns that DCOM wasn’t functioning correctly. This seems to be caused by a problem with the Windows installation on the virtual machine. To solve this development was moved to a new machine.

Despite getting the compilation to work OPC Foundations code was abandoned in favour for the Advosol implementation. The main concern with OPC Foundations code was that it was necessary to edit and program in a C++ environment to be able to get the wrapper to work correctly. With Advosol this wrapper was already implemented correctly.
6.2 Future work

Before this concept is implemented in systems and delivered to any clients there is a need to do some further developing of the OPC server. The main aspect is to be able to import the Item definitions from an external file. At this moment it is hard coded into the server and it is needed to recompile every time items are added or removed. By implementing a way to read the item definitions from an external file scalability increases. This is because all network devices may not use the same MIB.

It is possible there could be some way to optimize the code to execute faster. Today when exceptions arise it’s likely an OID or IP address that is wrong which makes the program slow because it waits to time out. This is because only one thread is used and loops through all the IP addresses. This issue might be solved by implementing multithreading in the application. If each IP address could run in its own thread only the IP address that needs to time out should be affected.22

Lastly it is very important to do a detailed survey of the licensing rules that each part of the OPC server uses. All assumptions regarding licensing within this thesis is the author’s interpretations.

22 Gold, 2005
Chapter 7: Conclusion

The purpose of this thesis work has been to implement an OPC server with SNMP capabilities and to be able to demonstrate the functionality on a concept level towards ABB/FACTS network devices. The final application meets all the specifications set by ABB/FACTS and it has been demonstrated to the personnel.

During the thesis work at ABB/FACTS the author has been given an insight in the work and approach on how an international company develop control systems. Being able to implement an OPC server has given the author an opportunity to successfully use all the knowledge acquired during his three year at Mälardalen University.

Furthermore by combining the OPC core function and a working SNMP library there is no need to buy a third party application that does the same thing. ABB/FACTS have in the past done some trials on implementing SNMP support in their OPC. Those times they have used various working applications that have high licensing fees.

Previous tested SNMP OPC servers:

Netlistener: 168 Euro per license.
Matrikon SNMP OPC Server: 1600 Euro per license.
Obermeier Software SNMP-OPC Gateway: 598 Euro per license.
Kepware SNMP OPC Server: 995 US Dollar per license.
SAE Automation SAEAUT SNMP OPC SERVER: 790 Euro per license.
Chapter 7: Conclusion

When FACTS ship out their equipment to clients they need to buy one license per NMS computer. This might seem like a low cost but in the long run it will add up to a significant amount.

The cost for the Advosol source code is 1075 USD for the non-exclusive right to use it on a development site. It also includes the right to distribute the server as part of the licensee’s products without any run-time royalties. So by developing your own application there is a potential huge saving. Not including the cost of personnel during further development of the application.

ABB/FACTS ship around 50 stations per year to different locations around the world. If they were to buy the SNMP OPC server Netlistener from Somebytes it would cost 168 Euro / 1439 SEK (exchange rate 8,567SEK/Euro) per license. This would cost 1439 SEK * 50 stations = 71950 SEK a year. So there is a potential saving of 64875 SEK the first year (, with 1075 USD = 7075 SEK).

---

**Accumulated cost in SEK per license**

![Graph showing accumulated cost in SEK per license for Netlistener and Advosol licenses.](image)

**Fig 8. Accumulated cost in SEK per license**
By year 10 the accumulated cost for buying separate licenses are 719500 SEK. A note should also be made that this is calculated that each station only uses one NMS computer. If ABB/FACTS want SNMP redundancies the cost would double.
Bibliography

ABB AB, (2013), Ökar kapaciteten i befintliga kraftledningar, Retrieved 2013-04-23 from http://www.abb.se/cawp/db0003db002698/7f7df0d410e4a85ac1257339003984e2.aspx

ABB Power Systems, FACTS, (2013), There are better ways in FACTS, Sweden


