

Control and Automation of Electricity Transmission and Distribution

Mathias Bartoll

*Department of Computer Science
Mälardalen University
Västerås, Sweden*

mathias@mathiasbartoll.com

Examiner:
Ivica Crnkovic

September 27, 2004

Abstract

In today's power system industry there is a numerous actors who needs access to real-time data and historical information. This requires a system that can distribute information independently of the geographical location. The current solution to the problem is a Web-based SCADA (Supervisory Control and Data Acquisition) system. It uses the benefits that Internet has to offer like; easy access through Web browsers and low development cost. There are also arising problems such as; to guarantee the security and the unpredictable transmission times. This thesis is mainly focused on the Web-based approach using different modern component-based techniques. It presents industry standards and current solutions from different vendors today.

Contents

1	Introduction	1
1.1	Background	1
1.2	Purpose	1
1.3	Organization	2
2	Electricity Transmission and Distribution	2
2.1	Energy Management Systems	3
2.2	Substation Automation System	3
2.3	Master Terminal Unit	3
2.4	Remote Terminal Unit	3
2.5	Intelligent Electronic Device	3
3	SCADA	4
3.1	Functionality	4
3.1.1	Human Machine Interface	4
3.1.2	Alarm and Event Monitoring	5
3.1.3	Reports and Information Sharing	5
3.1.4	Data Logging and Archiving	5
3.1.5	Access Control	5
3.2	Three generations of SCADA Architecture	5
3.2.1	Monolithic Architecture	6
3.2.2	Distributed Architecture	7
3.2.3	Network Architecture	8
3.3	Web-based SCADA system	9
3.3.1	Web-based Architecture	10
3.3.2	Functions	11
3.3.3	Quality of Service	12
3.3.4	Security	14
3.4	International Communication Standards	15
3.4.1	IEC 61850	16
3.4.2	Specify IEC 61850 with UML	18
3.4.3	CIM XML	19
3.5	Using Modern Component-Based Technologies	23
3.5.1	Java and EJB	24
3.5.2	CORBA	26
3.5.3	.NET	27
3.5.4	SCADA in .NET	29
3.6	SCADA Vendors Technology Solutions	31
3.6.1	ABB SattGraph 5000	31
3.6.2	Axeda Supervisor	32
3.6.3	Siemens Simatic WinCC	34
3.7	Future Thoughts	35

List of Figures

1	Monolithic architecture.	6
2	Distributed architecture.	7
3	Networked architecture.	8
4	Architecture of a Web-based SCADA system.	10
5	Three-tier architecture of a Web-based SCADA system.	11
6	Implementation of real-time data monitoring.	12
7	Implementation of alarm display and control.	13
8	CIM model visualized with UML.	20
9	The import/export process of an CIM XML document.	22
10	SpecNet architecture.	24
11	CORBA middleware approach.	27

1 Introduction

Power systems are an important part of the infrastructure supporting our society. These power systems has grown enormously over the last years and become extremely large and complex. The complexity is due to factors such as; the need for more efficient operations, the expanding system scale and the demand of various new services to the customers. The control and distribution of power system data is managed through a SCADA system.

SCADA systems are used for monitor, control and logging of power system states. It collects incoming power system data from field devices to a central computer database. Users from different departments and companies access this information via Internet. This is the approach of today's SCADA systems. The privatization of electric utility companies all over the world has brought competition and numerous changes to SCADA systems. The need of information exchange with external organizations requires quick access and data security. To be able to communicate with other vendors systems, the power industry has gone from proprietary protocols to standard protocols. This new industry accepted communication standard relies on the Internet Protocol (IP). This solution offers a low cost system with Web browsers as the display system for the client. The requested information can be available anywhere in the world through an Internet connection.

There are numerous ways to implement a Web-based SCADA system. However, when these systems become more and more complex the need arise of a modular system. The purpose is to divide the big system into smaller parts or components and give the customer the ability to incrementally build the system. The benefit for the developer is faster release cycles for every component and easier maintenance since every component is independent. Different vendors offer this component-based approach today.

1.1 Background

Electricity transmission and distribution includes a large set of processing for bringing electricity from different type of power stations to its final end users (households, companies). A long the way the electricity passes through several substations. Automation of these is of a crucial manner for efficient and reliable distribution. Despite the long tradition of power systems, it is still a growing area in the automation. The continuously evolvment is because the need for new requirements, standardization which increases the interoperability with other domains and the automation of countries in the third world. The typical architectural solution is based on SCADA solutions.

1.2 Purpose

The aim of this work is to present current solutions and trends for automation of electricity transmission and distribution. The overview includes topological

and architectural aspects. In that subject, SCADA is included with its principles and architectural solutions together with a survey of SCADA vendors and their technological solutions. A analysis of developing SCADA with different modern component-based technologies is presented, where the .NET platform is the primary technology.

1.3 Organization

This thesis is organized in the following order. In section two, there is an introduction to power systems and its main components. The next section discusses SCADA systems. It begins with describing its functionality and the evolvement of technology solutions. With the new open Web-based solutions, standards and component-based technologies are further discussed. We look at three SCADA products from different companies to see which standards these systems support. The SCADA section is ended with a look at the trends of the future. Finally there is a conclusion.

2 Electricity Transmission and Distribution

Electric energy generation, transmission and distribution are a fundamental need of our society. From the user perspective, the basic requirement is to obtain high quality power at a proper cost with no interruptions. Power grids of different topologies are responsible to transport energy and distribute it to end-customers. Such nodes in a network are called substations, they take over the voltage transformation and/or the routing of energy flow by means of the installed switchgear. These substations can, depending on the importance of the station and its degree of automation, be manned or unmanned. Supervisory Control and Data Acquisition and Energy Management Systems (EMS) applications are used to control these power grids. They support energy trading and the functions to remote control substation equipment. Substations are controlled by Substation Automation Systems (SAS). An SAS can be classified as a distributed soft real-time system, they continuously control, monitor and protect the network to avoid unplanned network outages. This covers all the high voltage equipment outside the substation (cables, overhead lines etc.) as well as those inside the substation (circuit breakers, transformers etc.). The SAS also has the ability to provide remote access facilities for SCADA and EMS applications. It is usually composed of Intelligent Electronic Devices (IED) that is connected by a communication network with routers and gateways. An IED is a device which can run executable program code and provides a communication interface. Some real-time critical functions can be executed autonomously on a single IED while other functions are realized in a distributed form over many IEDs.

2.1 Energy Management Systems

An Energy Management System (EMS) is one of the most important types of supervisory systems in a utility's main control center. The functionality comprises the full range of SCADA, network analysis, dispatcher training simulation, offline network planning, generation control and scheduling. The new generation of EMS uses the new industry standard Common Interface Model (CIM) produced by the EPRI's Control Center Application Program Interface (CCAPI). This standard describes EMS data aimed to reduce cost and time needed to add new applications from different suppliers as well as to include interfaces between EMS systems and external systems. The proprietary user interfaces is replaced with Web browsers to view, modify, monitor and control the execution of the EMS functions [26].

2.2 Substation Automation System

Substation Automation System (SAS) can be described as a distributed soft real-time system. Its purpose is to take automated decisions with minimal user intervention. To satisfy the goal it is usually composed of 20 to 100 IEDs which are connected by a high-speed communications network [5, 14].

2.3 Master Terminal Unit

The Master Terminal Unit (MTU) is usually defined as the central host or master of a SCADA system and is located at the operator's central control facility. The MTU initiates virtually all communication with remote sites and interfaces with an operator. Data from remote field devices is sent to the MTU to be processed, stored and/or sent to other systems. It monitors each remote device and can perform predefined actions if the data from such a device triggers an event.

2.4 Remote Terminal Unit

The Remote Terminal Unit (RTU) is a stand-alone data acquisition and control unit. It is located on the remote site and its function is to control process equipment at the remote site, acquire data from the equipment, and transfer the data back to the central SCADA system such measurements as; voltage, current, power, reactive power, as well as circuit breaker positions (open/close).

An intelligent RTU has modularized hardware architecture that supports an open communication protocol - http, and provides a standard interface to Internet/Intranet, which makes it easier to apply a Web-based SCADA system [13].

2.5 Intelligent Electronic Device

An Intelligent Electronic Device (IED) is a device that can run executable source and provides a data communication interface. Devices like a programmable con-

troller, intelligent sensors, intelligent RTUs or a workstation PC classifies as an IED [5].

As the communication bandwidth increases, more intelligence will move into the remote devices, IEDs. SCADA systems connecting directly with SCSEs (Substation Control Systems) and these connections will replace RTUs. This will relieve the SCADA system from some of its work, but old devices and RTUs will yet stay for a long time and require support from the SCADA system.

3 SCADA

SCADA stands for Supervisory Control And Data Acquisition and are essential parts of the energy management system. As the name indicates, SCADA systems focus on the supervisory level used for control, operation and monitoring. The control system consists of a central host or master (MTU); remotes units (RTU's or IEDs); and a collection of software used to monitor and control remotely located field data elements. In order to support SCADA data transmission, a communication network is essential.

The ability of a SCADA system to not only monitor the status of equipment, but to control the remote equipment from a central location, allows more efficient and cost-effective management of remote sites. Routine adjustments to levels, power, pressure, temperature, etc. improves system performance while reducing maintenance costs. Response time to emergencies is significantly reduced.

3.1 Functionality

Main function of a SCADA system is to provide accurate and up-to-date information on an electrical power network and to allow system operators to respond to all power network conditions.

3.1.1 Human Machine Interface

A typical SCADA system provides a Human Machine Interface (HMI) allowing the operator to visualize all the functions as the system is operating. It supports various functions such as; mimics - a graphical representation of the process with dynamically updated values, trends - graphs of variables against selected time periods, reports - allows process variable to be summarized on a periodic basis, alarms - provides presentation of process alarms.

Most of the common HMI software packages use standard data manipulation/presentation tools for reporting and archiving data and integrate well with Microsoft Excel, Access and Word.

With Web-based technology, data collected by the SCADA system is sent to Web servers that dynamically generate HTML pages. These pages are then sent to a LAN system at the operator's site or published on the Internet.

3.1.2 Alarm and Event Monitoring

The system must be able to detect, display and log alarms and events. When there are problems, the SCADA system must notify operators to take corrective action. A Knowledge Based Systems (KBS) is used to identify events and assists operators for the post-fault analysis. Fault diagnostic assistance is also provided through combined use of protection models and rules [6]. Alarms and events should be recorded so engineers or programmers can review the alarms to determine what caused the alarm and prevent them from happening again.

3.1.3 Reports and Information Sharing

Reports can be produced by querying a database with SQL (Structure Query Language) type queries to gather the required information. Sharing data with other users through windows sockets, Web servers and Web services allow almost every computer around the world to access and use the information, assuming they have the correct permissions.

3.1.4 Data Logging and Archiving

Logging can be seen as medium-term storage of data on disk. It is typically performed on a cyclic basis, i.e. once a certain time-period, file size or number of points is reached, the data is overwritten. Archiving is long-term storage of data either on disk or on another permanent storage medium. Logged data can be transferred to an archive once the log is full. Archiving data for later review is helpful for solving problems as well as providing information to the user.

3.1.5 Access Control

Role-based security allocates users to specific groups, determined by the employee's position. As consequence, they will have access to different information. The security issue is further discussed in section 3.3.4 since making SCADA open using WANs (Wide-Area Networks) and mobile networks as communication backbone results in higher demands on that aspect.

3.2 Three generations of SCADA Architecture

SCADA for power systems was developed in the 1960's. Since the early 1980's, there has been two major changes in the overall design, representing a total of three generations of SCADA architecture [3]. The improvements in the architecture have changed from a centralized computing system to a distributed network-based system in the early 1990s [2].

Good SCADA systems today not only control processes but are also used for measuring, forecasting, billing, analyzing (past/present) and planning, meeting a

completely new level of control automation. It must be able to interface with equipment and tools that were not available five years ago and they need to be flexible enough to adapt to tomorrow's changes.

The remarkable cost and performance breakthroughs in the last decade in terms of computers, their networking infrastructure and middleware technologies have created new opportunities for designing wide area controls. New technologies are providing huge increase in performance at lower cost. There are several factors that motivate the evolution of system and power control. The privatization of electric utility companies all over the world has brought competition into the electricity supply industry that leads to numerous changes in the SCADA/EMS. One of the changes requires exchange of information with external organizations such as generators, distributors and brokers. The increase of transactions between buyers and sellers do not only require the tracking of more data but also produces unforeseen loading of the grid. This data exchange requires quick access and data security. The security is also involved when dealing with blackouts and the concern with terrorism. The security must be guaranteed against natural forces and malicious acts. Another driving force is the new generation of smaller units such as wind units and micro turbines. The SCADA system is characterized by geographical spread that requires a distributed network system [4].

3.2.1 Monolithic Architecture

In the beginning when SCADA systems were developed, the concept of computing in general was a single monolithic system that performed all calculations associated with a specific process. Each centralized system stood alone because networks were generally nonexistent in the early 1980s. As an outcome of the lack of networks, the SCADA systems were standalone systems with virtually no connectivity to other systems, see figure 1.

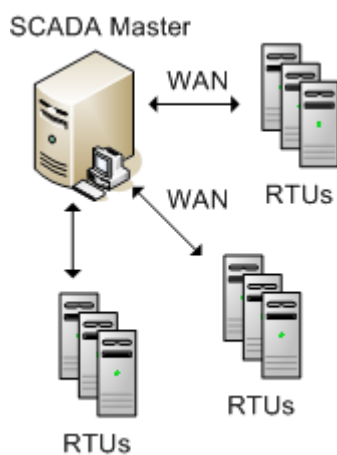


Figure 1: Monolithic architecture.

The vendors of the RTU equipment developed the WANs protocols and these protocols were often protected by proprietary. As consequence, other vendors were not allowed to build equipment that could communicate with these protocols. These protocols were generally very simple, supporting almost no functionality beyond the requirements to scan and control points within the remote service. This meant that it was not feasible to integrate other types of data traffic on the RTU communication network [3].

To guarantee redundancy and a fail-safe system, the use of two identically equipped systems were connected at the bus level. One of them configured as a primary system while the second configured as a standby system. The functionality of the standby system was to monitor the primary system and take over in an event of failure. This type of redundancy meant little or none processing was done by the standby system.

The first generation of SCADA systems based on mainframe systems, a monolithic system performing all computing, was generally limited to the hardware, software and peripheral devices that the vendor provided.

3.2.2 Distributed Architecture

In 1988, the second generation of SCADA systems began to take advantage of the Local-Area Networking (LAN) technology. With this new technology, the system could distribute the processing across multiple stations instead of performing all the processing on a single mainframe as illustrated in figure 2. The stations, consisting of computers, shared real-time information and had different functionalities such as human-machine interface, calculation processes or database servers [3].

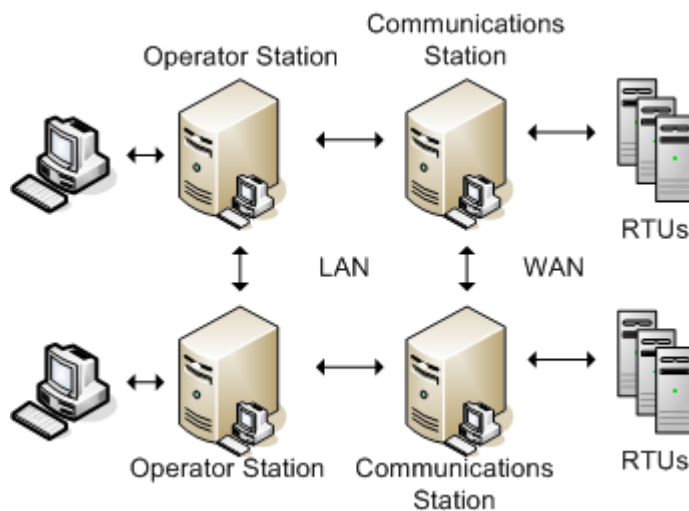


Figure 2: Distributed architecture.

The networks that connected the system were generally based on LAN pro-

protocols, which have a limited cable distance between the stations [3]. Another restriction was that the vendors often created their own network protocols. This lack of standard limited the connection of network devices from other vendors to the SCADA LAN.

With the distribution of processing, the redundancy and reliability of the system as a whole was greatly improved. Instead of having a standby system processing almost nothing, the distributed architecture kept all stations on the LAN in an on-line state all the time. If one HMI station went down another HMI station could be used for operating the system without waiting for the takeover from the primary system to the standby system.

The distribution of functions across multiple systems provided more processing power for the system as a whole compared to the monolithic system. Still the SCADA systems were limited to hardware, software and peripheral devices that the vendors provided. The improvement of the second generation was in the LAN technology. The WAN communication was unchanged and still limited to RTU protocols.

3.2.3 Network Architecture

With this generation the architecture left the vendor controlled proprietary environment and become an open system architecture. The open standards and protocols now made it possible to distribute SCADA functions across a WAN, see figure 3.

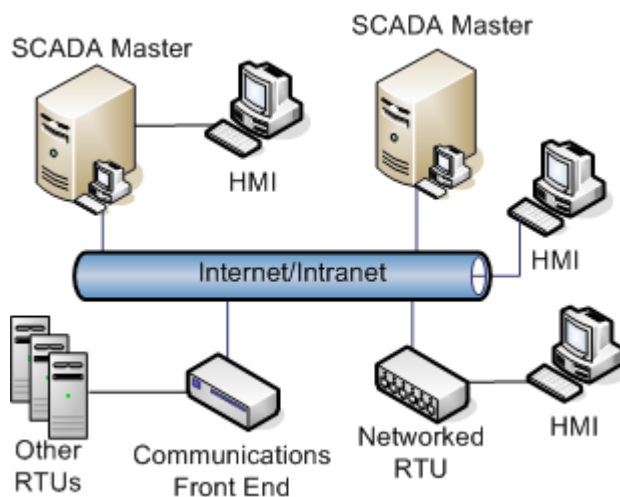


Figure 3: Networked architecture.

The major improvement in the third generation is the use of WAN protocols, such as TCP/IP, allowing the portion of the master station that is responsible for communications with field devices to be separated from the master station [3]. The advantage of locating the communications front-end closer to the field devices such

as RTUs makes it possible to directly reach each RTU in a single hop. The elimination of repeaters and associated overheads greatly simplifies a radio network and reduces its cost.

The open standards eliminated many of the limitations from the previous two generations. The user could now connect different third-party peripheral devices. With the new open systems the SCADA vendors could now concentrate on the SCADA master station software and let other actors create the hardware.

Introducing WAN greatly improved the reliability of the system. The second generation's limitation to LAN could in an event of a total loss of the facility housing; result in that the entire system would be lost as well. With the ability to distribute the processing across physically separate locations, the system can survive despite loss of a single location.

3.3 Web-based SCADA system

The privatization of electric utility brings new requirements to SCADA systems. Information exchange with external organizations such as distributors and brokers are neither time critical nor needed on a dedicated basis, but it require quick access and data security. The World Wide Web (WWW) has become a convenient way to access information on the net because the Web browser integrates different network services into a common easily accessible user interface [15]. With WWW, it is possible to send a lot of data from one side of the world to the other in almost no time. Technological advances in networking have also made it possible to develop a low cost WWW display system for accessing information over Internet. The Web technology enables the same GUI to be used and accessed locally within the LAN or Internet without incurring additional development or maintenance cost.

Web-based SCADA or Internet SCADA makes use of IP, which enables data to be routed over the Internet. The ability of IP to address a large number of devices in groups or sub-networks makes the network operation easier and therefore reducing the occurrence of human errors. Another advantage of IP is the simplification of SCADA host application development. Multiple protocols can now be addressed by a common interface, delegating to the network itself the responsibility to resolve the different locations, protocols, interfaces and baud rates.

Before implementing a Web-based SCADA system, two major issues must be taken into consideration. The first one is the aspect of time delay, which can lead to irregular data transmission and data loss. In the worst case, this can make the whole system unstable. Delays can occur due to buffer delays during packet assembly, conflicts on the asynchronous network, re-transmission due to errors, network traffic and routing path. The typical delay in TCP/IP in Ethernet is 7-200 ms [14]. The performance of the application is limited by the speed of data communication of the user connection. The other issue is the problem of security. When malicious hackers can grant access to a system, the consequences can be catastrophic.

Section 3.3.3 covers the delay concern and section 3.3.4 reveals some security methods.

3.3.1 Web-based Architecture

The architecture of a Web-based SCADA system is shown in figure 4. The Web browser (client) gets SCADA related Web pages from the Web server, which receives requested information from the connected intelligent RTUs [1]. This architectural design requires that the field device in this example, the RTU, have TCP/IP network support.

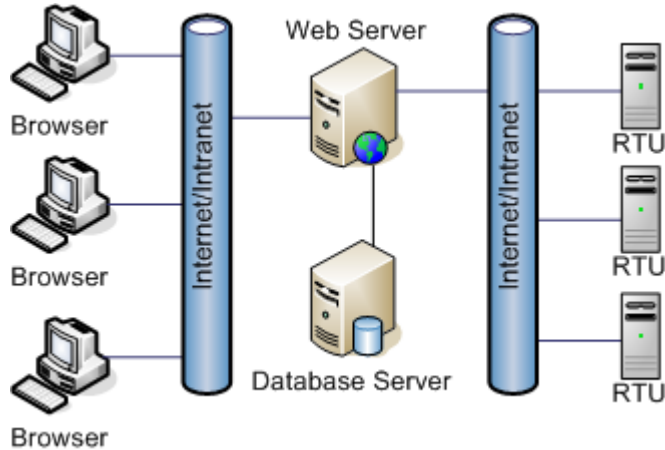


Figure 4: Architecture of a Web-based SCADA system.

The Web-based applications inherit basic features from traditional client/server applications. The client/server structure provides the scalability and robustness required to support mission-critical applications throughout the enterprise comprising thousands of users [15]. This architecture is object-oriented, server-based and database driven and is ideally suited for Web-based workflow solutions.

A Web-based SCADA system inherits three-tier client / server architecture, which is flexible and scalable enough to handle larger applications. There are a numerous advantages with this architecture [1]. The functionality of each service is clearly defined and implemented through a set of components, which brings clear and consistent development goals. The management gets easier because services divide the functionality into different distinct task. Changes in the implementation of one component will not affect other components. Finally, the functionality of a specific service is encapsulated. This isolation of functionality makes it easy to trace any error in the implementation to the corresponding component.

The three-tier architecture is illustrated in figure 5. RTUs and Web browsers are located in the top layer, the client services layer that provides an interface to other applications or operators. The Web/application servers located in the applications services layer realizes all application programs and a database server located in the data services layer provides storage and low-level manipulation of data in the database.

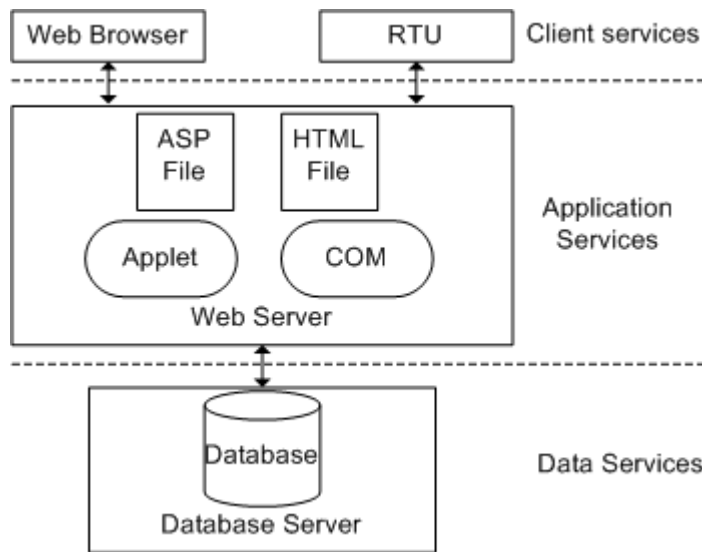


Figure 5: Three-tier architecture of a Web-based SCADA system.

3.3.2 Functions

A Web-based SCADA system provides the same critical functions as a traditional SCADA system. The difference is that they are implemented through several sets of Web site components instead of integrated software programs. As additional service, functionality like alarming and reporting can use emails in an Internet SCADA systems. When an alarm occurs, alarm emails are sent out to email-enabled cell phones and PC's. An Auto-dialer has the capability of sending SMS messages and voice warning to required maintenance and management units according to pre-defined alarm conditions.

The following two sample implementations from [1] will demonstrate how real-time data monitor and alarm display and control is implemented through a set of Web site components. The environment consists of three computers. A Web server running a Windows 2000 server with Internet Information Services (IIS) and a SQL server. The second computer runs a Web browser and the third computer imitate an intelligent RTU by running a Java program.

The first example is the functionality of real-time data monitor that sends periodically updated data to operators, illustrated in figure 6. The gathered data is stored in the database server, which is mapped to the source file in the Web server. When the target data in the database is updated, the source file is automatically refreshed with the new data. An applet is connected to the source file to get and display the latest data in an HTML file. The HMI file, in which the applet is included, establishes a human machine interface and completes some initiations. The data monitor function starts from the operators view when the Web browser sends an HTTP GET command to the Web server, asking for the hmi file. The Web browser receives the file, display it contents and the applet is downloaded to run to

show the target data.

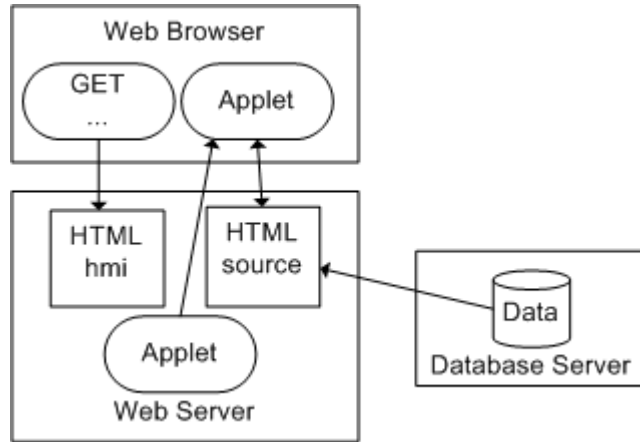


Figure 6: Implementation of real-time data monitoring.

The second example is the alarm data display and control function, shown in figure 7. It samples field data periodically, processes the gathered data and automatically makes related operations according to the information. The intelligent RTU periodically collects field data and sends it to a Web server. It communicates with an open interface, which is an ASP file in the Web server. The data sent from the field device is using the Java package HTTP Client that collects data as parameters in an HTTP POST command. The ASP file receives the command, parse the data and save them into the database. The COM component is responsible for automatically verify the data in the database by running some rules. If it finds something out of the ordinary, it will update an HTML file that shows periodically information to operators through an applet similar to the HMI file in example 1. The component also sends corresponding commands to the RTU as well.

3.3.3 Quality of Service

Changing to TCP/IP networking enables the management of SCADA networks to be integrated into a system common to the corporate data networks. TCP is an Internet protocol used to ensure reliable End-to-End communication. Despite its advantages it has the drawback of the non-deterministic nature of Ethernet and TCP/IP communications. The transmission times can vary and under extreme conditions the performance may not be predictable. These problems are solved by the Quality of Service (QoS) standards that guarantee delivery of connection-oriented traffic over TCP/IP networks [16]. QoS standards are central to the success of TCP/IP. They are designed to enable a network to provide transport services appropriate to the needs of each application. SCADA data traffic for example; it requires very low and consistent transmission delays but limited bandwidth. Administrative data and e-mails require higher bandwidth but are far less critical of

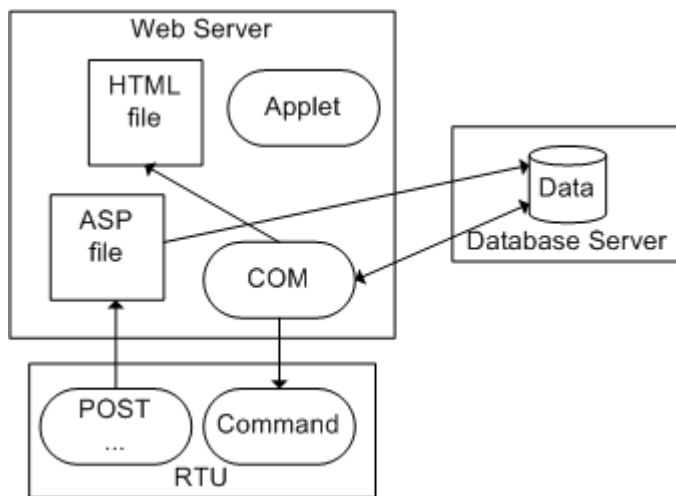


Figure 7: Implementation of alarm display and control.

time delays and variations.

The choice of providing QoS on TCP/IP networks rely on two schemes; Differentiated Services RFC 2474 (Diffserve) and Integrated Services RFC 1633 (Intserve) [16]. In the Diffserve technique it is the network that controls the QoS. Data transfer uses three classes: expedited, assured and best efforts forwarding. Data frames are marked as near as possible the network edge with the best suitable priority. The marking is applied by a device called label switch router. These routers can after the marking treat the frames appropriately. Diffserve uses Multi-Protocol Label Switching (MPLS) to deliver, which is a system supported by a number of major network equipment suppliers. MPLS can be used for setting up Virtual Private Networks (VPNs) that enable separation of differing data traffic types on a single physical network. This separation can be required for security management.

In the other technique, Intserver, it is the end system or the application that controls the QoS using Resource Reservation Protocol (RSVP). It proposes three levels of service: guaranteed, controlled and best efforts. The choice of leaving the end system computers to set the priorities can in practice lead to many claimants for the highest priority.

One thing to remember is that QoS cannot be guaranteed on Ethernet LAN, which provides the last distance to the SCADA measuring points and master systems. This is not a problem if the Ethernet LAN offers 10 or 100 Mbit/s together with the low bandwidth requirement of SCADA.

TCP/IP networking with QoS has the technical capability to support other power system operational requirements as well as business administration.

3.3.4 Security

"Are we vulnerable?" asked Joseph Weiss, executive consultant for KEMA Consulting, which is based in Fairfax, Va. "Of course, we are. We designed ourselves that way" [17]. The problem is that remote devices and SCADA systems were never designed with security in mind. Weiss continued; "When companies designed control systems worldwide, there were always two unwritten assumptions; everyone assumed the system would be isolated, not connected to anything else. We also assumed that the only people who would use the control system were people who were supposed to use it. That was a good assumption for another day." [17].

With the Internet technology, the companies now let the Internet carry SCADA messages instead of installing expensive private telecommunication. Field devices like RTUs, pumps and breakers now even have their own plug-in connections.

When Weiss investigated the industrial control systems, he concluded that it consists of two operating systems [17]. The first uses Windows or Unix for the operator console. That system provides role-based security, determined by an employee's position. For example, a plant manager and an operator would have access to different information. This system is relatively secure despite occasional hacks and viruses. The second operating system is the control system. Its functionality is to receive and sort data, respond to commands, and the like. These systems were originally designed to operate in isolation and usually have a password control. There are some differences from a conventional network. A real-time control system prioritize its operations. Each task such as receiving data to processing and device actuation is time critical. However, it will stop the process if something with higher priority appears. In contrary to a regular PC or network that will run a calculation until it is finished.

Field devices are built to manage a specific operation. Because of that, they use inexpensive, low-cost microprocessors. This cause trouble when implementing encrypted authentication. The processors before the 486 microprocessor, dating from 1989, cannot run such instructions without unacceptable delays. The TCP/IP implementations must be very simple in such field devices and the result is none support of https (Hyper Text Transfer Protocol Secure). Even CRC-check (Cyclic Redundancy Check) sums of packets are often not calculated. This is one big problem with the simple field-level TCP/IP applications. To solve this security issue is to place a dedicated encryption device between the SCADA field device and the modem that links it to the Internet. It would not only scramble the data, but also do it in a way that authenticates the sender as a trusted source.

TCP/IP itself does not contain any mechanism for security. However, it can be implemented at higher level in order to protect data. Instead of using standard http-protocol, it is easy to use an https-protocol. The https uses SSL (Secure Socket layer) which provides methods for authentication and encryption. This method is usually used when transaction security is to be ensured [18].

Securing the SCADA LAN from the Internet is managed by using a firewall.

The firewall device can deny access to server from unauthorized client and prevent most illegal attacks.

Eric Byres, research manager at the Internet Engineering Laboratory of the British Columbia Institute of Technology in Burnaby said: "In IT, data integrity and asset protection are number one. In the industrial world, plant safety is primary. Our whole starting point is different and that impacts everything from audits to passwords. We need to take what IT has given us and modify it to work for us." [17]. Byres stated an example: "Standard IT policy is to lock down a console after someone makes three bad password attempts. That is great on your desktop. But what if someone made the mistakes because he's panicking that a recovery boiler is going through the roof?" [17].

3.4 International Communication Standards

In today's competitive power business, a basic requirement is easy exchange of information between all devices in a power control and management system. It also requires free access for all interested parties. The power system is controlled by the control and management systems, which includes several devices and sensors for data acquisition. These controlling systems produce a large amount of data. The gathered data is turned into information, which is a properly pre-processed set of all relevant data in the context of the process that the system is build for. The full semantic meaning has to be supported and provided everywhere it is needed by the communication. The provided information is then used for decision making. It can be either system operations or business management done by humans or safety related decisions done automatic. The request to include all data or information respectively for decision-making calls for a common language for it to be communicated, an international communication standard [8].

The article [7] presents an example that shows how important standard-based communication systems are. A large electric utility has more than 200 different protocols running on intelligent devices within its distribution network. A well-known vendor supports more than 100 different RTU protocols. A wide variety is not always a good feature. The utilities spend an escalating amount for real-time information exchange; the costs spent on system integration and data maintenance is exploding. The larger expense in time and money comes from maintaining the API's to existing in-house applications. The purpose of standardization is to effectively and efficiently perform seamless device data integration and sharing information. Another key issue is the support of re-usability. The re-usability is a crucial factor in reducing the costs of the overall system design, engineering, operation and maintenance.

The IEC (International Electrotechnical Commission) is a worldwide organization for standardization comprising all national electro technical committees (IEC National Committees). The object of the IEC is to promote international co-operation on all questions concerning standardization in the electrical and electronic fields [9].

Since 1995, about 60 experts from 14 countries have been tackling today's problem with a large variety of protocols in three IEC working groups. They responded to all these challenges and created the single, global and future-proof standard for substation communications, and this is IEC 61850.

3.4.1 IEC 61850

World-wide, electric utility deregulation is expanding and creating demands to integrate, consolidate and disseminate real-time information quickly and accurately within and with substations. Development and sharing information among all industry participants on critical resources is one of the most crucial issues for reliable power systems.

IEC 61850 is the international standard for substation automation systems. It defines the communication between devices in the substation and the related system requirements. It supports all substation automation functions and their engineering. The general definition of IEC 61850 can be applied in all areas where there is a need to exchange real-time data and metadata.

The objective with the IEC 61850 is the interoperability between IEDs that originate from various suppliers, to enable the unrestricted exchange and usage of data to perform their individual dedicated functionality. Interoperability is the ability of two or more intelligent electronic devices from the same or different vendors to exchange information and use that information for correct cooperation. Improving device data integration will reduce the cost for engineering, operation, monitoring and maintenance and increasing the agility of the whole life cycle of a substation.

The standard set of documents is divided into ten parts, which focus on four major issues [10].

- Functional modelling: a functional model of the substation automation application domain (part 5).
- Data model: a data model for substation automation system defining the syntax (naming scheme) and semantics of the application level data that can be exchanged in SAS (part 7).
- Communication service modelling and protocol stacks: a communications service modelling defines different ways of accessing data of IEDs. The communications services and data models are mapped to concrete data communications protocols stack (part 7, 8 and 9).
- SAS engineering and testing: an XML based Substation Configuration Language (SCL, part 6) is defined to describe all information exchanged in a substation communication network. The last part (part 10) describes recommended testing for IEC 61850 compliance.

Communication protocols in IEC 61850 divide data into logical groupings such as protection functions, supervisory control and metering and measurement. There are 13 different groups with the intent that all data that could originate in the substation can be assigned to one of these groups. Each group is further subdivided into 86 logical nodes representing application specific meaning and is intended to provide separate sub-categories of data. There are 355 different classes of data that are used to construct logical nodes. These data classes are divided into 7 categories such as; system information, metered data and status information. Accessing data in the network is analogous to accessing data across a conventional IT network using a tool like Windows Explorer, i.e. browse the network until the data source is located, then drill-down into the data source until the data is located. By communicating and browsing the contents of the device you can see what data it holds.

The standard communication protocols TCP/IP and OSI over Ethernet 802.3 or X.21serial are used. It is object-oriented modelling compliant, supporting inheritance, encapsulation, hierarchical, models and other object-oriented features [11].

IEC 61850 supports both client-server and peer-to-peer communication. It is the peer-to-peer communication ability that is used to exchange Generic Object Oriented System Events (GOOSE) messages between IEDs. GOOSE is used to model the transmission of high priority information. The model is based on cyclic and high-priority transmission of status information. Upon detecting an event, the IEDs use a multicast transmission to notify those devices that have registered to receive the data. The performance requirements states that no more than 4ms is allowed to elapse from the time an event occurs to the time of message transmission. The amount of traffic generated after an event is dependent on the number of IEDs, network topology and the type of event. In these Ethernet networks, collisions are quite possible, so GOOSE messages are re-transmitted multiple times by each IED [12].

With the IEC 61850 standard all IED vendors are required to provide a descriptor file for their IEDs in eXtensible Markup Language (XML) format. XML provides data and also instructions on how the data should be interpreted. A specific device can upon a request send their configuration in XML. With the XML and the substation configuration language defined by IEC 61850 the data will be available for any vendor. This allows dynamic configuration of the GOOSE communications with the publisher/subscriber model [12].

IEC 61850 meets the requirements for an integrated information management providing the user with consistent knowledge of the system online, realized by the self-description and metadata. Information models are standardized for vendors where objects are accessed by names rather than vendor specific point numbers. This standard is today supported by the major vendors such as ABB, Siemens and Axeda.

3.4.2 Specify IEC 61850 with UML

The Unified Modeling Language (UML) is an object-oriented modeling language used for system specification, visualization and documentation. Modeling is the designing of software applications before coding. Modeling is an essential part of both large and small projects. Using a model assures that different characteristics such as; functionality, security and reliability are fulfilled before implementation; otherwise changes are difficult and expensive to make.

UML is a way of describing software with diagrams and is a language that both users and programmers can understand. It has emerged as the software industry's dominant language and is already an Object Management Group (OMG) standard. It represents a collection of best engineering practices that have been proved successful in the modeling of large and complex systems.

Contrary to IEC 61850, the protocol standards in the past were rather simple. They included a small amount of primitive data types and a small number of types of application communications services. The mapping of the application data items to underlying communications stacks was also relatively straightforward. This simplicity made it possible to specify and maintain data models of protocol standards based on informal and textual notations. In [5] we can see that this approach is almost impossible on today's more complex protocols. The complexity has risen based on several factors. The exchanged information consists of more complex pieces. Communication standards make use of existing layers like Ethernet and TCP/IP. Communications services are realized through reuse of standard IT technology like, remote procedure calls and transaction support. As a result, it leads to a standard with many implicit and explicit dependencies. Using UML models makes it easier to identify inconsistencies that can appear in a big complex standard.

To use an UML approach offers many advantages. In [5] there is an example illustrated with one concrete logical node called XCBR, which represents a circuit breaker. The XCBR is first presented as defined in the standard and what the procedure is to find relevant defined parts of its specification. The logical node is illustrated in the form of tables spread throughout different documents. The text and table description is hard to follow and there is a great risk that human errors occur.

The UML model of the same logical node is developed into a class model. A class wraps attributes (data) and behaviours (methods or functions) into a single distinct entity. The model consists of several classes and they are linked together through different relationships. One relationship example is the inheritance link indicating one class is a superclass of the other. The UML model is created and maintained with a CASE tool, which provides a wealth of features such as; support propagation of changes, pop-up menus to select available types, etc. The checking of the logical correctness of the type definition is managed through the easiness to display for a specific type of all the attributes, operations, associations and their cardinalities. The possibility to display the model elements of interest in the class

diagram keeps the partial display manageable despite a complex diagram. The views of the UML diagrams, the model browser and the model element documentation may also appear simultaneously on the screen. Providing this facilitate on text documents is almost impossible.

The complete UML models can be used for automatically code and documentation generation. The advanced documentation plug-ins is able to represent the model in several formats, such as XML and RDF schemas as well as text and table format.

Another cost and timesaving benefit is the easier maintenance. It allows for consistent modifications from one version of the standard to another. Changing a type name is made and documented at one place within the model. Compare this to the changes necessary at several places in the documents based on the text notation.

The UML model allows specified mappings to other existing UML models in a formal way. Devices and applications, which implement the IEC 61850 specification, are not restricted to the substation automation domain. They have to communicate with network control centres and applications in the entire electric utility environment. An example of a mapping can be the UML model for the IEC 61850 to a common information model.

The motivation of using UML model is to improve the consistency validation and maintenance of the standards. It gives everyone from business analyst to designer to programmer a common vocabulary to talk about software design.

3.4.3 CIM XML

Models are essential to the operation of modern power systems. Simulation of the power systems is necessary for both planning and operations, which depends on appropriate models. Operational models are typically more comprehensive than those used for planning. They support analysis of incoming SCADA data as well as simulation of expected and unexpected operational scenarios. The desegregation of utility functions and the introduction of power markets in many national power systems have increased the model maintenance burden. This means that that many overlapping models must be created and coordinated with each other hosted in different geographical locations. To exchange this system modeling information with one other supports system planning functions such as maintenance scheduling, transmission planning and operations planning. For the analysis to be realistic, major portions of neighbouring systems must be modeled in addition to an organization's own service territory. The information needed for real-time power system operation requires far more details about the field equipment and its connectivity. These models must include the substations bus segments, switches and measurements details. Node/breaker model is the referred name of such a model [29, 30].

A vendor neutral choice that meets the demand of node/breaker level of detail is Common Information Model (CIM). CIM defines a utility industry standard object-model for the development and integration of applications used for electric power

systems engineering, planning, management, operation and commerce. The CIM does not require any particular instrumentation or repository format. It is only an information model - unifying the data, using an object-oriented format, made available from any number of sources. Due to the size of the complete CIM, the object classes contained in the CIM are grouped into a number of logical packages, each of which represents a certain part of the overall power system being modeled. Collections of these packages are progressed as separate international standards.

The CIM is comprised of a specification and a schema. The specification defines the details for integration with other management models. It includes expressions for common elements that must be clearly presented to management applications such as; object classes, properties, methods and associations. The specification defines syntax, rules, meta schema, meta schema elements, and the rules for each element. It also defines a CIM syntax language based on Interface Definition Language (IDL) called Managed Object Format (MOF). What it not describes is specific CIM implementations, APIs, or communication protocols - those topics are outside the scope of the specification. The schema provides the actual model descriptions. It supplies a set of classes with properties and associations that provide a well-understood conceptual framework within which it is possible to organize the available information about the managed environment.

UML is used to diagram CIM models, a fragment of the CIM wires model is shown in figure 8 [29]. The base class of the CIM is the PowerSystemResource class defined to represent a generic power system component. A variety of subclasses are derived from this abstract class representing various power system equipments (for example lines, capacitors, transformers, breakers and substations).

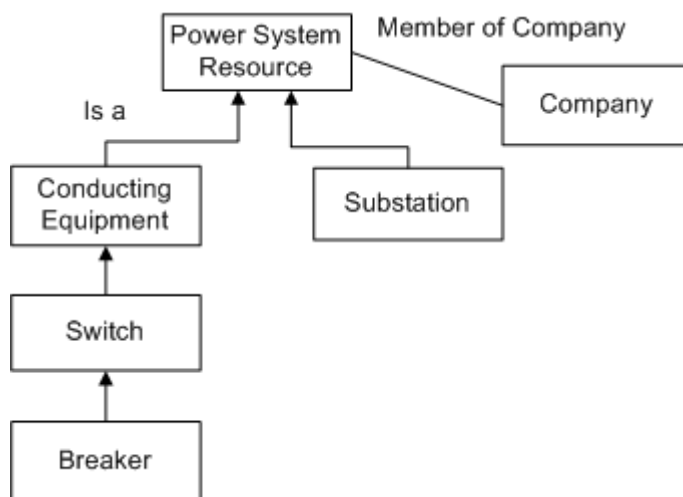


Figure 8: CIM model visualized with UML.

Since CIM is an abstract model, it is neither a modeling database specification

nor an exchange format [29].

XML (eXtensible Markup Language) is a World Wide Web Consortium (W3C)-endorsed standard for document markup. It defines a generic syntax used to mark up data with simple, human-readable tags. It provides a standard format for computer documents. XML is a metamarkup language for text documents. Data is included in XML documents as strings of text. The data is surrounded by text markup that describes the data. XML's basic unit of data and markup is called an element. The XML specification defines the exact syntax this markup must follow: how elements are delimited by tags, what a tag looks like, what names are acceptable for elements, where attributes are placed, and so forth [31].

XML is a metamarkup language, which means it does not have a fixed set of tags and elements that are supposed to work for everybody in all areas of interest for all time. It allows developers and writers to define the elements they need as they need them. For reasons of interoperability, individuals or organizations may agree to use only certain tags. These tag sets are called XML applications [31]. An increasingly number of industry groups codifying data exchange formats in the XML language resulting in standards like CML for the chemical industry and HL7 for the health care industry to name a few [29].

The following simple example illustrates how a valid well-formed XML document can be defined.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE note SYSTEM"InternalNote.dtd">
<note>
  <to>Bill</to>
  <from>Steve</from>
  <heading>Reminder</heading>
  <body>Don't forget the meeting on Tuesday!</body>
</note>
```

The note tag has a header and a message body. The tags in the example above (like <to> and <from>) are not defined in any XML standard. These tags are invented by the author of the XML document. The author and reader must agree on the meaning of things to be able to understand the document. The CIM XML language introduces a power system oriented vocabulary that includes <transformer> and <breaker>. These tags are drawn from the CIM schema.

A Document Type Declaration (DTD) is used to validate the document. The DTD lists all the elements, attributes, and entities the document uses and the contexts in which it use them. The DTD may list items the document does not use as well. However, the DTD syntax is quite limited. A given DTD or schema only describes the data in particular message type; it does not describe how the instance data for that message type relates to instance data for other message types. W3Cs standard, Resource Description Framework (RDF), on the other hand, describes an entire model including datamodel, syntax and schema.

RDF model provides a statement including resource, property and a value. A resource is anything that can be uniquely identified by a Uniform Resource Identifier (URI). A property is any characteristics of a resource that can be described as a value. These values can be a string, resource etc.

RDF is used as a schema to describe the CIM. RDF uses XML as a common syntax for the exchange and processing of metadata. Therefore, while XML messages are an excellent way to pass chunks of information, RDF provides applications with an understanding of how these chunks of information fit into an overall framework.

CIM XML is an application of the RDF. It consists of a profile, sub-setting the RDF syntax and an RDF schema derived from the CIM for power systems, figure 9 shows the process to export/import CIM/XML documents. With an agreed CIM RDF schema, the XML document can be converted to a power system model for export. The resulting CIM XML model exchange document can be parsed and the information imported into a foreign system. The CIM XML document can then be viewed through a browser using an XSL stylesheet to format the contents for human readability. Separate XML tools are used to validate the format of the file and the conformance with XML and the RDF Syntax [29].

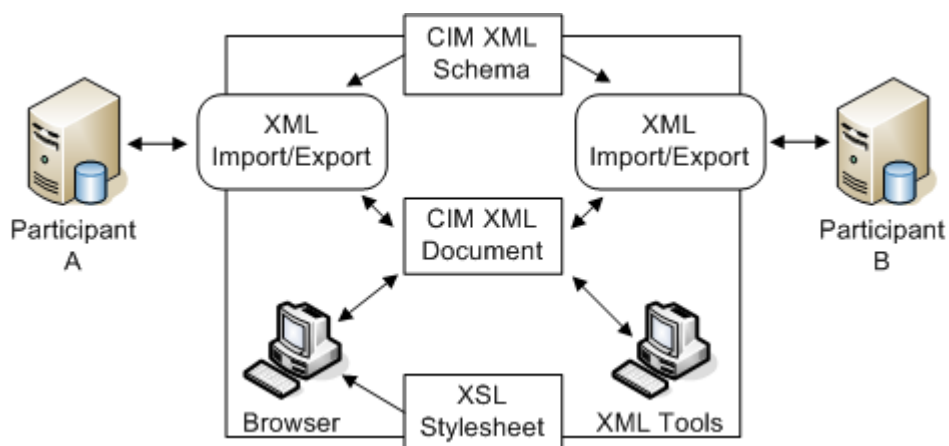


Figure 9: The import/export process of an CIM XML document.

An example of a CIM XML document describing a substation called "East" is shown below [29]. It is owned by a company named "BPA" and contains a circuit breaker made by "Admirable Electric". Each resource has a unique identifier (ID) associated with it. The substation includes a description of a breaker and a reference to the company, which is described as a separate element. The tags in this document are within the cim namespace, which means that they are defined in the CIM RDF schema.

```

<cim:Substation ID="ID1" cim:PowerSystemResourceName
="East">
  
```

```

<cim:MemberOfCompany resource="#ID3">
  <cim:Contain>
    <cim:Breaker ID="ID2" cim:PowerSystemResource
      Name="11023"
      cim:Manufacturer="Admirable Electric" cim:Normal
      Open="true"/>
  </cim:Contain>
</cim:Substation>
<cim:Company ID="ID3" CompanyName="BPA">
  <cim:CompanyDescription> This is a government
  organization
  </cim:CompanyDescription>
</cim:Company>

```

CIM was approved as an international standard (IEC 61970-301 CIM base) in 2002. This means that the generation and transmission portion of the CIM model is now internationally accepted. It provides assurance to vendors and utilities that the CIM is sufficiently complete and stable to be included in their products and system integration solutions.

3.5 Using Modern Component-Based Technologies

Web applications on the Internet have advanced from being a static HTML page with graphics, sound and other basic features to dynamic content based on server-side extensions such as CGI (Common Gateway Interface), ASP (Active Server Page) and database integration. These Web applications are developed in ordinary programming languages and make use of the existing object-oriented technologies and services for the platform. During the evolution, the Web browsers have also evolved to support new technologies, which include client-side active components such as ActiveX controls, Java Applets, integrating with XML and scriptlets. With these technologies, it is possible to implement a complex software program in an ordinary SCADA system as a Web-based application. An application hosted by a Web server consisting of components like COM component, applet, ASP file, HTML file and so on. Operations to the application are carried out through the HMI, which in this case is the Web browser.

To be able to deal with a more complex environment that contains of big IT systems there will be a need to divide the big systems into smaller parts or components. Benefiting from the well-proven modularized architecture of a Web-application makes SCADA system functions like; data-acquisition, alarm display and control and supervisory control very independent and efficient components. This division leads to faster release cycles for every component and the ability for the customer to choose components from different vendors. The component based system makes it easier for the customer to build the system incrementally since they can buy the components when the need arise.

3.5.1 Java and EJB

Java is a technology designed with networking in mind. This makes writing networking programs easy with straightforward commands for Java applications and applets to send and receive data and to communicate across the Internet.

Java is platform independent and portable. It executes in a run time environment called virtual machine that executes bytecode (platform independent code) generated by a Java compiler. The Java virtual machine (VM) can be incorporated or embedded in Web browsers or the kernel of the operating system. This gives the advantage to run a Java-based SCADA system across different platforms without any modification [15].

The following example from [18] demonstrates existing SCADA software from Siemens Spectrum called SpecNET. This system integrates some of the SCADA functions like the one-line diagram generation and real-time display. The architecture of the system, illustrated in figure 10, is based on different modules. The SCADA server running on a Unix workstation with the Solaris operating system. The SpecNET server runs on a PC with Windows NT and an IIS to support Web publishing and gateway function. The SpecNET server is written entirely in Java and are based on the client/server architecture to fit the different platform requirements. The client side is a PC with Java enabled Web browser. These users access the SCADA system through the SpecNET server. JavaCon (Java connectivity to SCADA) is used as a data communication bridge between the SCADA system and SpecNET server. The SpecNET server receives messages from the SCADA system through JavaCon and provides the necessary support for client HMI.

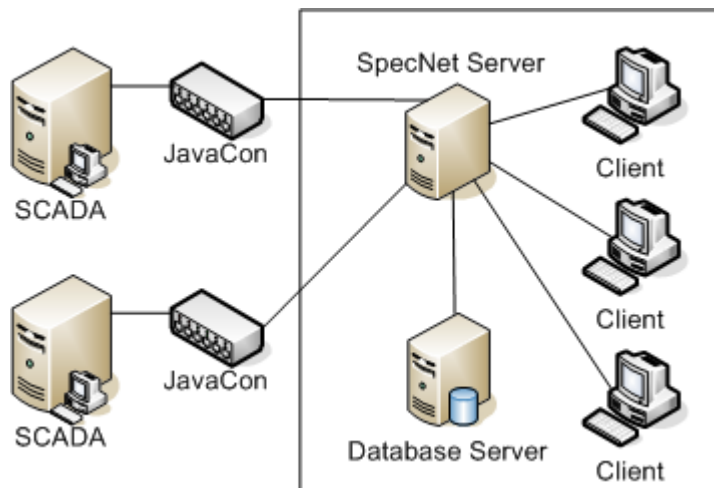


Figure 10: SpecNet architecture.

JavaCon is used as an interface to the SCADA server through a JNI (Java Native Interface) connection. JNI is used when the SpecNET server want to communicate with the SCADA server. The JNI provides Java code that runs inside Java VM

enabling it to operate with applications and libraries written in other programming languages. The function module in JavaCon is coded using SCADA API, allowing it to talk with the SCADA server kernel program like a native component to get data. It also enables it to receive real-time data from the database.

The communication between SpecNET server, SCADA server and clients are built upon two methods, namely RMI (Remote Method Invocation) and socket communication. The Java RMI system allows an object running in one Java VM to invoke methods on an object running in another Java VM. SpecNET RMI implementation comprises two separate programs: a server and a client.

The server application creates some remote objects, makes references to them accessible, and waits for clients to invoke methods on these remote objects. The client application gets a remote reference to one or more remote objects in the server and then invokes methods on them. When a remote client requesting public information the server grant access immediately. For confidential information the client request are redirected to the user authentication object asking for username and password and allocating the user to the different objects depending on the user level.

A socket connection will be set up between the SpecNET server and the client site when the user wants to see the substation auto generation one-line diagram. Before the socket connection, the SpecNet server runs the JNI communicating with the SCADA server to obtain the one-line diagram raw connection information. Multithreading is used to process multiple client requests. Each client gets a separate thread for the data transmission to the SpecNET server.

The databases for storing static and dynamic data are using the JDBC (Java Database Connectivity) and ODBC (Open Database Connectivity) driver. It provides a convenient data accessing mechanism. Database operation on the server site is enabled through RMI and JDBC access. JDBC is a Java API for executing SQL queries. It lets the Java program to send SQL queries to the appropriate database.

Another possible deployment of SpecNET is to use EJB (Enterprise JavaBeans) technology, which provides low-level services such as; support for transactions, concurrency, persistence, security and life cycle management for each service. With EJB, the SpecNET system achieves better performance.

The architecture of EJB has an EJB server to manage resources needed to support the EJB container. The EJB container provides a secure, scalable and transactional environment in which enterprise beans operate. These enterprise beans are server-side Java objects that communicate with the SCADA server or database. A Java Name Directory Interface (JNDI) is a naming service where clients can lookup and obtain references to a bean. After that procedure, the client talks with the beans through home and remote interfaces. The EJB framework offers two types of enterprise beans, namely session and entity. Entity beans differ from session beans in several ways. Entity beans are persistent, allow shared access, have primary keys, and may participate in relationships with other entity beans. They typically represents specific data or data in persistence storage like share information. Real-time

information display can for example be designed as an entity bean. A session bean is a non-persistent object that operates exclusively on behalf on the client session that creates it. A typically example of a session bean is client authentication.

3.5.2 CORBA

Common Object Request Broker Architecture (CORBA) is an object-oriented open-standard based middleware for distributed objects defined by the OMG. Unlike RMI, CORBA is not tied to one language, and as such, can integrate with legacy systems of the past written in older languages, as well as future languages that include support for CORBA. CORBA is not tied to a single platform, a property shared by RMI.

The basic CORBA paradigm is that of a request for services of a distributed object. The services that an object provides are given by its interface. Interfaces are defined in IDL. Distributed objects are identified by object references, which are typed by IDL interfaces. CORBA allows objects to make requests of remote objects (invoking methods), and allows data to be passed between two remote systems. Remote method invocation, on the other hand, allows Java objects to be passed and returned as parameters. This allows new classes to be passed across virtual machines for execution (mobile code). CORBA only allows primitive data types, and structures to be passed - not actual code.

Under communication between CORBA clients and CORBA services, method calls are passed to Object Request Brokers (ORBs), illustrated in figure 11. The ORB is the distributed service that implements the request to the remote object. It locates the remote object on the network, communicates the request to the object, waits for the results and when available communicates those results back to the client. By hiding all details of data transmission from the client, it facilitates the communication between clients and server. Therefore, the user can concentrate on the implementation of the objects services and their usage in client programs, avoiding all peculiarities of communication. To target an object, the client only has to specify the object reference, which is created when the object is created. The General Inter-ORB Protocol (GIOP) provides interoperability (a set of message formats) between ORB implementations from different vendors. The GIOP can be mapped onto different telecommunication protocols for transport protocol independence. Thus, the Internet Inter-ORB Protocol (IIOP) is a mapping of GIOP onto the TCP/IP protocol stack. IIOP defines the transfer syntax and a set of message formats for ORB interoperation just as an IDL interface defines the protocol between an object and its clients.

Real-time CORBA (RT-CORBA) provides capabilities to ensure predictable behaviour end-to-end for requests that traverse from one object to another. It defines standard interfaces and QoS policies that allow applications to configure and control processor, communication and memory resources. Finding applications in distributed systems, which needs predictable response time and prioritized invocations. Although RT-CORBA is oriented towards applications with hard real-time

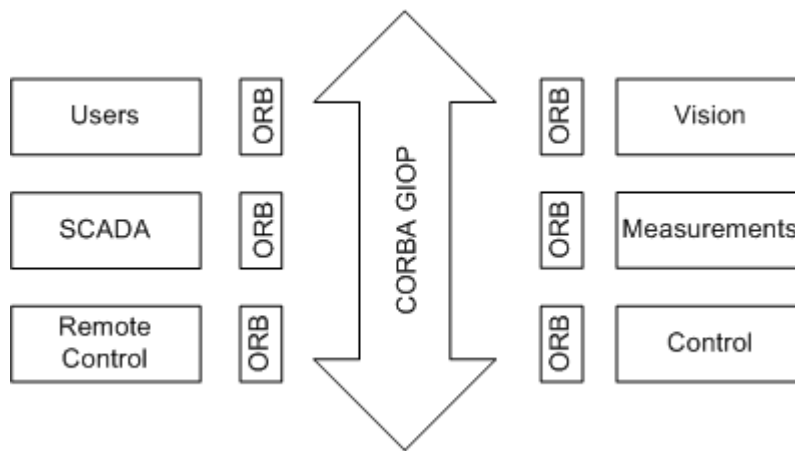


Figure 11: CORBA middleware approach.

requirements, it also supports applications with soft-real time requirements [27].

Using a middleware approach that lies between the application and network hides the complexity of communication and data transfer. To invoke a remote procedure call appears no different from a local one from the developer point of view. Middleware can be defined as "software (glue) that connects two otherwise separate applications" [32].

Distributed systems, as SCADA, are essentially heterogeneous (different architecture, hardware, networking, operating systems and software). They comprise of physically distributed components that act as a unity to achieve a common goal. The challenges to develop such a system are to achieve: heterogeneity, scalability, concurrency, openness, transparency and mobility [32]. Using CORBA for this task provides interaction of sub-systems and modules within a SCADA system. It supports interoperability, reusability and portability of systems components. There are several numbers of CORBA implementations and mappings to various programming languages that are available from different vendors. The real-time and fault-tolerant specifications are also a benefit. The use of a middleware can significantly reduce development time, increase interoperability, portability and reusability of distributed systems [32].

3.5.3 .NET

The .NET platform is a new development framework providing a new Application Programming Interface (API), new functionality and new tools for writing Windows and Web applications. It is not only a development environment; it is also an entire suite of servers and services that work together to deliver solutions to solve today's business problems [19]. One of the main ideas with .NET is to make the connectivity and interoperability between businesses easier.

Inside the .NET platform the .NET Framework reside. It includes the Common

Language Runtime (CLR) and a comprehensive class library for building Web and Windows applications. The CLR is the mechanism through which .NET code is executed. It provides a lot of added value to the programs it supports, because it controls how a .NET program executes and sits between the program and the operating system [19]. It can implement security, versioning support, automatic memory management through garbage collection, and provide transparent access to system services. The CLR makes it unproblematic to design components and applications whose objects interact across languages. Objects written in different programming languages can communicate with each other, and their behaviours can be integrated. For example, you can define a class and then use a different language to call a method on your original class or get a class from your original class. You can also pass an instance of a class to a method of a class written in a different language. The interesting aspect of CLR is that compilers can now be written for any other computer including those computers not running a Microsoft operating system.

The Base Class Library (BCL) provides standard functionality such as input/output, string manipulation, security management, network communications, thread management, text management, and user interface design features. It is a huge collection of managed code classes that integrate with the CLR. These class libraries provide support for tasks such as; data access, XML Web services, security, file IO, XML manipulation, class reflection, messaging, ASP.NET, and Microsoft Windows services.

The .NET Framework includes support for two distributed programming models: .NET Remoting and Web services. Both techniques support developing distributed applications and application integration. Web services provide a simple API using Simple Object Access Protocol (SOAP) messages to invoke methods. The clients do not have to know anything about the object model, platform or programming language used to build them and the Web service don't have to know anything about the clients. The only thing the two parts have to agree on is the format on the SOAP messages sent between them. This is defined by the Web service using Web Services Description Language (WSDL) and XML Schema [20]. Compared to Web services, which provides a simple programming model, the .NET Remoting offers a much more complex functionality. Remoting includes support for passing objects by value or by reference, callbacks, multiple-object activation and lifecycle management policies. In order to use .NET Remoting, a client needs to be aware of all these details; in short the client needs to be built using .NET. Since Web services can be implemented on any platform in any programming language we will focus on that technique when building a SCADA system.

A Web service represents black-box functionality like components that can be reused without worrying about how it is implemented. It allows you to provide access to functionality without having to build a complete application. A developer building the consuming client application can use one or more Web services to create a new application or complement an existing one. Because of the abstraction provided by the standard-based interfaces, it doesn't matter whether the application

services are written in Java and the browser in C++, or if the application services are deployed on a Unix system while the browser is deployed on Windows. The interoperability is one of the key benefits gained from implementing Web services.

A Web service relies on industry standards. One of the most important standards is SOAP, which is a vendor neutral integration protocol that standardizes network communications between software applications. SOAP is a general purpose protocol for sending messages from one software application to another. The key benefit with SOAP is that it can be used to invoke remote procedures between systems in a standard way that is platform, programming language and geographic independent.

A second standard, WSDL, is an XML-based language that describes the Web service. It includes details such as where to find the service (its URI), what methods and properties it supports, the data types, and the protocols used to communicate with the service. WSDL describes the Web service method interface thoroughly enough for it to be used to create proxy methods that enable other classes to invoke its members as if they were local methods [22].

To be able to publish and discover public Web services we need a directory standard called Universal Description, Discovery and Integration (UDDI). It contains information about the technical interfaces of a business services. Through a set of SOAP-based XML API calls we can discover these services, which can be invoked and used. UDDI has two parts: a registry of all a Web service's metadata, and a set of WSDL port type definitions for manipulating and searching the registry.

Web service consumers employ a discovery process to learn that a Web service exists and where to find the current WSDL document. Web service consumers set this discovery process on a target server by providing a URL to a discovery tool. The discovery tool attempts to locate DISCO documents on the target server and informs the consumer of the location of any available WSDL documents. The DISCO document is an XML document that contains pointers to such things as the WSDL file for the Web service [19].

For application data to be moved around the network by the transport layer, it must be packaged in a format that all parties can understand. Web services rely on the System.Xml.Serialization.XmlSerializer class to marshal data to and from SOAP messages at runtime [20]. For metadata, they generate WSDL and XSD definitions that describe what their messages contain. The reliance on pure WSDL and XSD makes Web services metadata portable; it expresses data structures in a way that other Web service on different platforms and with different programming models can understand.

3.5.4 SCADA in .NET

To get the communication to work, an interface needs to be implemented between the SCADA system and the system interested in accessing the information. Web Services is an integration solution allowing the owner of the SCADA system to

publish a real-time interface without regard for the external system [21]. This is accomplished with the WSDL that describes all the details of where to find the service and what methods and properties that is available.

To create a Web service the class must inherit from `System.Web.Services.WebService`. This class is placed in a source file and saved with an `.asmx` extension. The methods that you want to expose as Web services will be marked with the `WebMethod` attribute. Once this is done, these methods can be invoked by sending HTTP requests using SOAP. Each Web service should have a unique namespace that allows potential clients to separate it from other namespaces. By default, each `.NET` Web service is given the same namespace: `http://tempuri.org`. The namespace should be changed as soon as possible before publishing the service on the Web. Using a temporary name does not affect the overall functionality, but will affect consistency and violate Web service naming convention. Although most namespace names out there look like URLs, you don't need to use URLs. A unique name suffices.

Consuming a Web service is very straightforward too. You can very easily create a proxy class for your Web service using either command-line utility (`wsdl.exe`) or the "Add Web Reference" option in `VS.NET`. The Web service proxy hides the entire network and marshalling plumbing from the application code, so using the Web service looks just like using any other local object. To invoke a Web service requires a wrapper class like the proxy that lives on the client and exposes the same methods as the Web service.

`.NET` Framework uses disconnected data architecture, which means that data is first read from a database into a dataset and the Web service uses the dataset as a data source. The Web service can work with the dataset in much the same way as with the real data. When an update operation is performed on the dataset, these changes are written through to the underlying database. The main advantage in disconnected data architecture is that it takes less system resources than connected data architecture.

The security is a major issue when designing a system that will deliver critical data. Web services rely on HTTP, and integrate with the standard Internet security infrastructure. The best way to implement a Web service is to host it inside an IIS. The best part of hosting inside IIS is that you can use its strong security features without changing the client's code or the server's code. This means you can secure a remoting system just by changing the hosting environment to IIS and passing identifications (user name, password, and optionally, the domain) by setting a client configuration option.

IIS has rich support for various authentication mechanisms such as: Windows Integrated Security (NTLM), Basic authentication, Passport authentication and certificate-based services. NTLM provides the same robust security system as Windows NT. The drawback with NTLM is the loss of authentication support over firewalls and proxy servers. So for Internet based scenarios involving firewalls the choice is on Basic or Passport authentication.

After the user is authenticated, your next concern might be to hide sensitive

data that's being marshalled between remoting tiers. IIS also provides a strong solution for encryption, namely SSL. SSL is an industry standard for encrypting data and IIS fully supports SSL by using server side certificates. Clients then only need to specify https (instead of http) as the protocol in the server's URL [20].

Other ways to secure the system is to only pass data elements over the demarcation line, not important hints about the structure of the database or the system. The Web service interface should be developed by care. If not a two-way access with control capabilities is required, the source should be created as a read-only data source.

By using Web services, utility engineers and system operators will be able to provide access to different automation system data sources in the same way and with equal technology as the IT world is using. One computer system can describe its capabilities and available services to another, which helps guarantee loosely coupled, system-to-system integration. De-coupling is to keep the knowledge of the producer-consumer association strictly by the consumer leading to a publish-subscribe API. The owner of the SCADA system can manage upgrades and changes without any affect on third party systems. The original functionality will be provided by the new SCADA system and any additional functionality will be described via the WSDL file to the third party system.

3.6 SCADA Vendors Technology Solutions

3.6.1 ABB SattGraph 5000

SattGraph 5000 [28] is an object-oriented graphical HMI system based on client/server principles. With this approach, the system is fully scalable and flexible in the sense that is easy to expand to a more complex level without having to change system or to make a major investment. There are several levels of complexity available to ensure a system that complies as closely as possible with the needs of the customer. All versions of SattGraph 5000 can be connected to, and communicate with, other I/O devices.

ABB Automation is a member of OPC (OLE for Process Control) Foundation, an organization established in 1996 by companies in the field of industrial automation. The foundation is dedicated to ensure interoperability in automation by creating and maintaining open specifications that standardize the communication of acquired process data, alarm and events records, historical data, and batch data to multi-vendor enterprise systems and between production devices. Production devices like sensors, RTUs, HMIs, trending subsystems, alarm subsystems and more. Based on fundamental standards and technology of the general computing market, the OPC Foundation adapts and creates specifications that fill industry-specific needs.

SattGraph 5000 is equipped with an OPC client, which enables it to read, write and exchange data with other OPC servers on the market. This open software architecture means that it can be installed in any existing OPC based automation system.

The current system can freely communicate with SattGraph 5000 components via COM and OLE.

Some technical specifications and functionalities.

- Software used is Windows NT4.0.
- ODBC interface is used to communicate with databases such as Access, SQL Server or Oracle and allows access via your own application.
- Standard Windows packages can be integrated, like Microsoft Excel for handling statistics and presenting data graphically or Microsoft Word for producing formula-based management reports.
- Recorded data can be viewed in real-time or historical time. Historical reports can be generated for a day, a month or a year.
- The security environment has a number of predefined user privileges. Application specific privileges can easily be added. For preventing unauthorized users from accessing the system, the operator must log on before he can enter the system.
- A backup manager can backup data manually or run automatically daily, weekly or monthly. The backup does not interfere with the normal running of the system.

3.6.2 Axeda Supervisor

It is a complete Internet-based solution for control and information. Axeda Supervisor [24] supplies the software tools needed to build a complete automation solution. For example, businesses can develop discrete and process control systems using WizPLC, and then distribute and visualize the information for authorized users with Wizcon. These two tools together with three other are gathered within the Axeda Supervisor software suite.

Wizcon for Windows and Internet provides all the SCADA/HMI functionality for operator display on Windows NT, Windows 2000 or Windows XP, as well as the ability to publish the same information to any Java-compatible Web browser. This software provides historical and real-time information from the production sites to administrative services and beyond. Using the Wizcon development studio, a user can define the real-time database used by the applications. It contains the tag and alarm definitions, as well as the communications drivers. Tags refer to control values monitored by the system. Each tag is identified by a unique name and can be one of several data types; integer, real number, string, etc. Tags are associated with external device components, like registers or I/O points in the IED, and can be defined online. Once tags are created in the Wizcon definition module, they can then be used in other modules for display purposes or for calculating control functions. An alarm is a test on a tag value that generates a text message when

the condition is true. Alarm application messages are used to notify operators of exception conditions in the plant. Each Wizcon SCADA station can communicate with up to 32 networks of field devices simultaneously, with the ability to use different communication drivers for each field device.

Wizcon is then used to build the HMI interface, including images, charts and event summaries. These applications can either run on SCADA operator workstations under Windows 32-bit platform or be published on the Web server using the Wizcon automatic HTML page conversion utilities and Java applets. The real-time and historical data on the Wizcon server is then accessed by the Web-applications thru authorized users.

Wizcon's network is based on client / server architecture using 32-bit, preemptive multitasking and multi-threading technology to achieve good performance and ensure data reliability and integrity. This architecture design allows users to transparently define network data objects, without defining network tags or alarms in the local configuration database. The tags users define in one station are instantly available for use in any other station across the network. This configuration cuts the number of licenses required and preserves database integrity.

The Wizcon WizSQL Connection module uses the Microsoft ODBC interface. This allows the user to build event-driven SQL or ODBC queries for reading from and writing to databases like MS SQL Server, Sybase and Oracle. Wizcon also supports the OPC client and server interface, which allows its applications to instantly connect with available OPC servers, as well as supply data to OPC clients.

Information is secured by a number of user and group levels according to the user's position and role in the organization (engineer, operator, technician and so on). The authorization requires a username and password to grant access. Internet security devices such as firewalls and other type of encryption solutions can also provide enhanced security.

WinPLC is an open, standard-based solution for Windows NT. It can be used on the plant floor instead of standard IEDs or it can run side by side with the devices. WizPLC communicates with the terminal equipment in the plant through a fieldbus and updates Wizcon whenever there is a change in the tag values. This is possible since WinPLC and Wizcon shares the same database.

WizScheduler for Internet is a Wizcon add-in that enables users to plan, schedule and execute a variety of tasks based on the date and time via a simple Internet browser. Users can easily create daily or yearly remote and task-oriented schedules through an Internet browser. WizScheduler is both task- and time-oriented.

WizAAM (Advanced Alarm Module) is a decision module that allows you take appropriate action. It is used to provide information of events in real-time to the appropriate employee at any time via any communications medium, such as a Web browser, mobile phone, email, pager, SMS, printer or fax.

WizVCR allows users to replay a scene that occurred in the past, either locally across a network or via a Web interface. A control panel provides play speed control and choice of timeperiod.

3.6.3 Siemens Simatic WinCC

Simatic WinCC [25] offers all appropriate SCADA functions under Windows 2000/XP. In version 6.0 the scalability is consistent from a single-user solution with 128 tags to a client/server solution with an integrated historian and operator stations on the Web. The number of tags can be upgraded depending on the size of the project (128, 256, 8k and 64k tags).

The WinCC server allows multiple coordinated HMI stations to be operated together with networked automation systems. One server can supply up to 32 network connected (TCP/IP) clients with archive data, process data, messages and reports. The WinCC Web Navigator offers the user to visualize and operate the plant via Internet in the same way as the local operator station. The Web browser must have support for Active X.

WinCC uses standard interfaces to link the automation level to the IT world and to ensure information exchange between both sides, this tool is called IndustrialDataBridge. These interfaces include OPC in the field of automation and SQL databases in the IT world. There is the ability to integrate systems made by different manufactures using a large number of interfaces such as; OPC, SQL, OLE-DB and office formats. Microsoft SQL Server 2000 is integrated in the basic WinCC system - including its real-time response, performance and industrial standard.

WinCC offers a suite of different tools. One is already mentioned above, IndustrialDataBridge, which makes data exchange between different manufactures automation systems possible. Another called, Dat@Monitor Web Edition, is for displaying and evaluating current process status conditions and historical data on any office PC using Internet. The Web Navigator server supplies the Dat@Monitor client with current and historical process.

The Connectivity Pack includes the OPC HDA 1.0 (Historical Data Access) and OPC A&E 1,0 (Alarm & Events) servers for accessing historical data from the WinCC archive system or for transferring messages. The direct access to the archived data in MS SQL server database is possible due to OLE-DB.

Security is obtained by authorization. Different user access levels regulate the access rights that different employees have. In addition, the system also supports common security mechanisms like routers, proxy servers, firewalls, SSL encryption and VPN technologies.

With the newly released service pack 2, WinCC now supports the interoperable format of data namely XML. XML allows intelligent data to be shown that describes the content as well as the display. With OPC XML v1.0, it is possible for the automation area to use a manufacture-independent communication protocol. Another new feature is that WinCC SCADA servers not only run under Windows 2000 servers, but also under Windows 2003 servers. This multi-purpose operating system can take on different centralized or distributed tasks and comes with enhanced security, reliability, availability and scalability.

3.7 Future Thoughts

The need for compatibility in data exchange and the increasing cost-pressure force utilities to move towards integrated network solutions where Ethernet and IP are the two dominating technologies. These technologies have been established for many years in the IT infrastructure. The success is because IP can virtually be used over any physical media and the fact that the IEC have accepted IP as a network and access protocol. The utilities fibre-optic backbones also lead to a major growth of IP-based solutions. The rapid improvement in computing systems performance and the life cycle has been shorter than previously. The life cycle of SCADA systems are on the other hand comparatively long because it is difficult to replace a computing system in a timely manner. The gap between the two life cycles continues to grow, but prediction is that most field devices will be Ethernet/IP enabled in a near future [23].

The business role of SCADA systems has changed from just has been a management tool for electrical grids. Today it is a knowledge management tool, learning and gathering technical as well as commercial information and serving a range group of users. SCADA vendors are now looking to improve openness to the integrated business practices of the power industry. Energy companies become more focused on their supply, chains and systems that were once an exclusive domain now need to be integrated with other enterprise-level systems. This trend makes sure that all users within a utility have real-time access to the information, which they really need and have access rights to.

Most of the vendors today develop their system based on standard interfaces. Examples of these types of interfaces include OPC in the field of automation and SQL databases in the IT world. Non-proprietary communication in the field of automation has become very important. It ensures interoperability between different vendors.

XML is emerging as the standard language for data exchange in many business sectors and is starting to gain attention in the field of building automation. By supporting XML for building automation objects, manufacturers give their customers the flexibility to configure the system on their own, use a configuration package from another manufacturer, or use a third-party software package that supports XML as a file format. As we have seen, Siemens Simatic WinCC uses XML in their latest version and more vendors will probably follow.

While looking at the operating system choice, Microsoft is the big actor with the Windows versions NT/2000/XP/2003. Microsoft was in the beginning the first choice for SCADA developers and it is going to be very difficult to gather enough development resources away from Microsoft to compete in the financially limited market. A Web-based solution with Linux and Apache makes a lot of sense. However, customers must make it profitable for manufacturers to migrate toward Linux and Apache, either in place of or in addition to standard Microsoft offerings.

4 Conclusion

This thesis has described a power system with focus on the SCADA system in a Web-based environment. Migrating to TCP/IP and Ethernet networking gives the SCADA system more flexibility and scalability than any of its previous counterparts. SCADA information can be available anywhere geographically through an Internet connection allowing users to access real-time and historical data through Web browsers. This is achieved at a low cost of hardware, software, installation, administration and management. The Web-based approach together with new defined standards from IEC enables the electricity utilities to select equipment from a very wide range of vendors. Reducing the costs spent on system integration and data maintenance and gives support for re-usability. One of the overruling concerns in power system engineering is the safety of the system. When these industrial control systems used to monitor and operate now are linked to networks and the public Internet it definitely makes them harder to protect. The steps taken are stronger IT policies and encryption but this is still the most critical part of a Web-based SCADA systems.

Deregulation and the changing relationships between energy producer, seller, deliverer, buyer and user are creating a need for integration that has never been considered before. Component-based technology is one approach to shortened delivery times, easier integration and frequent upgrades to new releases of different products. The modularized architecture leads to faster release cycles for every component and the ability for the customer to choose components from different vendors. The customer can buy the components when the need arise. There are several technologies to use each of them having their benefits and limitations. The .NET example used with Web services shows how open standards like SOAP and XML will have a significant impact on existing and future utility automation systems.

SCADA system developed by different vendors in this thesis is all using object-oriented technology with a modularized system. Solutions based on standards that are required to divide the systems into exchangeable products. The trend shows that proprietary protocols are vanishing and open standards are the key for integrated low cost SCADA systems.

References

- [1] D. Li, Y. Serizawa, M. Kiuchi, *Concept Design for a Web-based Supervisory Control and Data-Acquisition (SCADA) System*, IEEE 2002, pp.32-36.
- [2] T. E. Dy-Liacco, *Modern Control Centers and Computer Networking*, IEEE Computer Application in Power, Vol.7 No.4, 1994, pp.17-22.
- [3] R. H. McClanahan, *SCADA and IP, Is Network Convergence Really Here?*, IEEE Industry Application Magazine, 2003, pp.29-36.

- [4] K. Tomsovic, D. Bakken, V. Venkatasubramanian, A. Bose, *Designing the Next Generation of Real-Time Control, Communication and Computations for Large Power Systems*, IEEE 2003.
- [5] O. Preiss, T. Kositc, C. Frei, *Data Communiactions Standards: A Case for the UML*, ABB Switzerland, Corporate Research.
- [6] J. A. Hossack, G. M. Burt, J. R. McDonald, T. Cumming, J. Stokoe, *Progressive Power System Data Interpretation and Information Dissemination*, IEEE/PES, 2001, pp.907-912.
- [7] S. Karlheinz, *Standard IEC 61850 For Substation Automation And Other Power System Applications*, 2002.
- [8] L. Andersson, K. P. Brand, *The Benefits of the Coming Standard IEC61850 for Communication in Substations*, Switzerland, 2000.
- [9] IEC, *Communication Networks and Systems in Substations - Part 1: Introduction and Overview*, 2003.
- [10] O. Preiss, A. Wegmann, *Towards a Composition Model Problem Based on IEC618500*, 2003, pp.227-236.
- [11] K. Schwarz, *Comparison of IEC 60870-5-101/-103/-104, DNP3, and IEC 60870-6-TASE.2 with IEC 61850*, 2002.
- [12] D. Proudfoot, *UCA and 61850 for Dummies*, 2002.
- [13] L. Gräsberg, L-O. Österlund, *SCADA ESM DMS - A Part of the Corporate IT System*, 2001, pp.141-147.
- [14] A-R. Khatib, X. Dong, B. Qiu, Y. Liu, *Thoughts on Future Internet Based Power System Information Network Architecture*, Power Engineering Society Summer Meeting, 2000, pp.155-160.
- [15] B. Qiu, H. B. Gooi, *Web-Based SCADA Display Systems (WSDS) for Access via Internet*, Power Systems, IEEE Transactions, Volume13 (2), May 2000, pp.681-686.
- [16] K-H. Mak, B. Holland, *Migrating Electrical Power Network SCADA System to TCP/IP and Ethernet Networking*, Power Engineering Journal, December 2002. pp.305-311.
- [17] A. S. Brown, *SCADA vs. the Hackers*, Mechanical Engineering, December 2002.
- [18] B. Qiu, H. B. Gooi, Y. Liu, E. K. Chan, *Internet-Based SCADA Display System*, IEEE Computer Applications in Power, 2002.

- [19] T. Thai, H. Lam, *.NET Framework Essentials, 2nd Edition*, O'Reilly, ISBN: 0-596-00302-1, Sebastopol, 2002.
- [20] D. Priya, T. Ewald, *ASP.NET Web Services or .NET Remoting: How to Choose*, 2002, <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnbda/html/bdadotnetarch16.asp>
- [21] T. Huneycutt, *XML Web Services in the Utility Automation/IT World*, Gridlogix, Inc., 2004.
- [22] J. Templeman, D. Vittler, *Visual Studio .NET - The .Net Framework Black Book*, The Coriolis Group, ISBN: 1-57610-995-X, Scottsdale, Arizona, 2002.
- [23] ABB Switzerland Ltd, *SCADA over IP-based LAN-WAN connections*, Utility Automation Systems, 2003.
- [24] Axeda Systems Inc, *Axeda Supervisor - Product Summary*, 2004.
- [25] Siemens, *SIMATIC WinCC Version 6.0 - New Perspectives on Process Visualization*, Germany, 2003.
- [26] W. Xingping, Z. Yang, W. Xiwei, *A New Generation EMS*, Power Systems Technology, 2002, pp.190-194.
- [27] A.D. Selvakumar, P.M. Sobha, R. Pitchiah, *Real-time CORBA on MICO-MT - Design, Implementation, Performance and Application*, India.
- [28] ABB, *SattGraph 5000 Object-oriented SCADA system*, Industrial Automation Software.
- [29] A. deVos, S.E. Widergern, J. Zhu, *XML for CIM Model Exchange*, Power Industry Computer Applications, 2001, pp.31-37.
- [30] A. deVos, C.T. Rowbotham, *Knowledge Representation for Power System Modelling*, Power Industry Computer Applications, 2001, pp.50-56.
- [31] E. T. Ray, *Learning XML*, ISBN: 0-59600-046-4, 1st edition, 2001.
- [32] E. M. Burmakin, B. A. Krassi, *Distributed Automation and Control Systems*, 2002, pp.25-28.