[1.]Matej GURÁŇ, [2.]Aleš JANOTA, [3.]Peter HOLEČKO

AUTOMATION OF SCADA SYSTEM DEVELOPMENT

[1-3.]Department of Control and Information Systems, Faculty of Electrical Engineering, University of Žilina, Žilina, SLOVAKIA

Abstract: Rapid development of large-scale SCADA systems for industrial applications puts a lot of stress on developers. Each SCADA system consists of a large number of objects unnecessary for communication and data manipulation. Low efficiency and high rate of human made mistakes rise call for automated solutions. This paper introduces application of an Object Generator that automates a way of pre-processing and importing of basic objects to the SCADA system. The Object Generator differs from other solutions by direct communication using APIs in runtime. It works with the Siemens products WinCC as a representative of the SCADA system and the TIA Portal as an integrated platform for control system development. The authors discuss motivation (within the context of maritime applications) and describe advantages and disadvantages of the presented concept. The main advantage of the promoted application is a decrease of human-made-mistakes in the process of linking process data to internals of the SCADA system.
Keywords: SCADA, development, object, security, WinCC

INTRODUCTION

The global maritime security market was valued at USD 17.13 billion in 2017, and is expected to reach USD 25.75 billion by 2023, by witnessing a CAGR of 7.03% during the forecast period, 2018 to 2023. The global maritime safety market is segmented based on technologies and systems into screening and scanning, access control, detectors, geographic information system, surveillance and tracking, weather monitoring, SCADA, communication, and others [5]. During the last decades remote command and control has been made feasible due to the development of networking technology and the advent of Industrial Control Systems (ICS).

ICS are command and control networks and systems designed to support industrial processes. The largest subgroup of ICS is Supervisory Control and Data Acquisition (SCADA) systems [3]. SCADA systems, because of their high-level automation mechanisms and data interpretation capabilities, have managed to reduce waste of time and provide cost savings in logistics, maritime transport and control operations [4]. SCADA systems are routinely seen on ships. These are often referred there to as Distributed Control Systems (DCS), having similar functions to SCADA systems (the field data gathering or control units are usually located within a more confined area). Behind the SCADA concept there is a history of over 50 years of development: from the first idea of supervisor control of systems to the most complex data acquisition and supervisory control systems nowadays [13].

Protecting critical infrastructure from cyberattacks poses unique challenges. The environments can be harsh and systems often use specialized protocols including BACNet, DNP3, IEC-60870-5-104, IEC 60870-6 (ICCP), IEC 61850, MMS, Modbus, OPC, Profinet, S7 (Siemens) and many others that represent the industry's most extensive support of SCADA. After three generations of SCADA – standalone SCADA, distributed SCADA, and networked SCADA, the next phase in the evolution of SCADA and logical platform for an upgrade seems to be Internet of Things (IoT) which revolutionizes SCADA by offering standardisation and openness. Some applications can already be seen, e.g. for real-time water quality monitoring [12]. Interconnection of the networks of both information and cyber-physical systems utilising SCADA, computer-based and wireless systems, including the information, services, social and business functions, defines the cyber environment. Such an environment is not limited to ships' systems but extends to shore based activities of both the ships operators and the port' cities [1]. This paper introduces application of an Object Generator that automates a way of pre-processing and importing basic objects to the SCADA system.

The Object Generator differs from other solutions by direct communication using APIs in runtime. It works with the Siemens SCADA system WinCC (V7.4) and the TIA Portal as an integrated platform for control system development. The authors discuss motivation and describe advantages and disadvantages of the presented concept.

WINCC SCADA CONCEPT

SCADA systems are rather complex and need an entire set of information and communication technologies. These technologies are needed for communication with both lower and higher layers of the architecture. Programming of Programmable Logical Controllers (PLCs) and intelligent devices that take direct control over managed appliances and technologies represent lower layers; higher layers include Enterprise Resource Planning (ERP) and Manufacturing Execution Systems (MES) that use data from SCADA to utilize management of resources and to improve workflow of the site.

Figure 1 illustrates a general scheme of the SCADA system. The key part of SCADA is a runtime engine that takes responsibility for processing of acquired data. Data originates from communication driver that communicates directly using protocols over physical buses, or it can use software interfaces (e.g. ODBC, or OPC). Data processed by runtime engine is stored in databases for manipulation and archiving purposes.

Nowadays, many companies also implement Application Programming Interfaces (APIs) into their solutions. These APIs make possible to not only import data on your own but also even manipulate with objects within SCADA system.

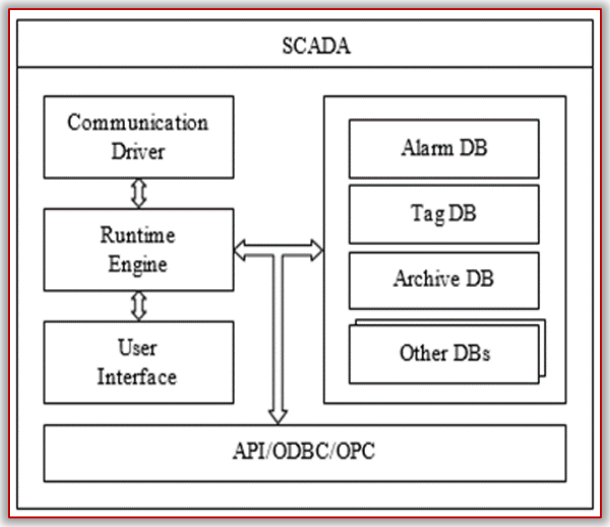

Figure 1. SCADA system internals

With the SCADA system SIMATIC WinCC V7.4, we get an innovative, scalable process-visualization system with numerous high-performance functions for monitoring automated processes, with complete functionality for all industries and features optimum openness.

WinCC consists of two key parts - Configuration Software (CS) and Runtime Software (RS) as depicted in Figure 2. CS takes responsibility for system settings and also provides the graphical interface for a user. On the other side, RS is invisible to the user but represents a key part of the whole system. RS communicates with low level appliances using protocols and application drivers and takes care of data acquisition and its proper manipulation. It also takes responsibility for dedicated databases that are used to store gathered data. Other support subsystems may depend on this system as well.

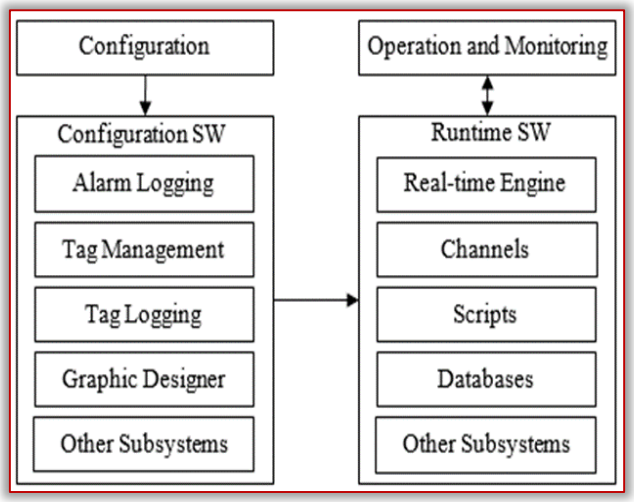

Figure 2. General WinCC Architecture

— Tag Management

Tag Management is a part of CS and takes charge of tag manipulation (adding, removing, modifying) and setting its attributes. The term tag is nothing else but the variable used to store acquired values from the process. One of the parameters held by the tag is an address, which is used to communicate with external PLCs or appliances, as shown in Figure 3.

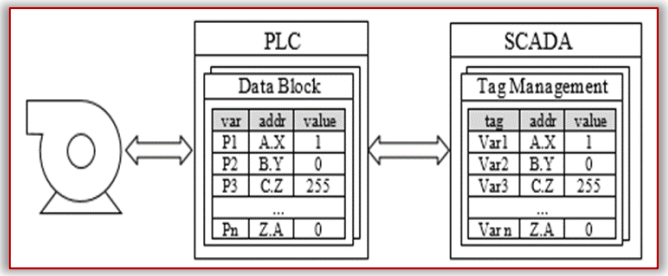

Figure 3. Communication between process control and SCADA

These addresses are points of our interest since developers of SCADA systems must manage them (add, edit, group, etc.) on their own. The number of tags results from technology size; for medium up to huge technological units it could be thousands. Considering this together with the fact that human developers make mistakes this is the main reason for automation.

— Alarm Logging

In order to capture any change of the monitored value, the Alarm Logging subsystem is used. Its primary objective is to help an operator with decision making process and to provide journal for later analysis. In some cases, acknowledgement of message is needed for the purpose of higher priority tasks. The alarms in working environment are generally organized in a table sorted by time, with the latest data found at the top. Alarms in WinCC are divided in two separate groups:

Bit messages – a message is triggered when a bit in tag is changed.
Analog messages – a message is triggered if limit values are exceeded (either over or bellow predefined boundaries). The value is gathered from the PLC in a defined period, and limits are set on the side of the SCADA system.

Figure 4 shows the example of an alarm table.

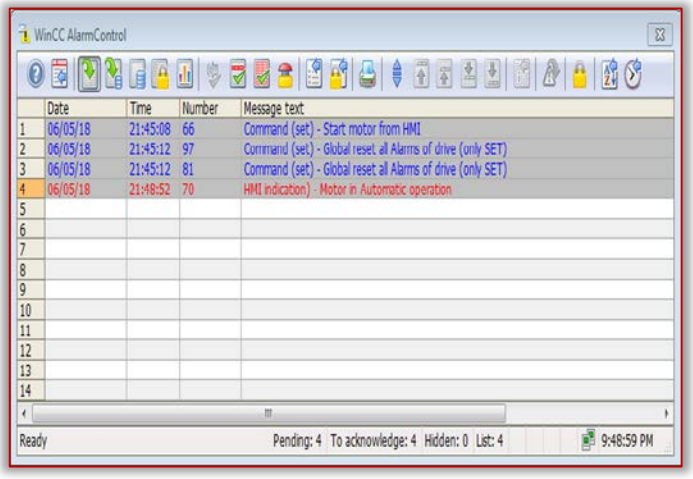

Figure 4. Alarm table example

— Tag Logging

Tag Logging is used for archiving the process value gathered and stored in a tag. Figure 5 shows architecture of Alarm and Tag logging and their relation with runtime software. The archiving subsystem consists of two separate groups – process and compression archives [8]. The former stores each value separately in the order it was read. This approach has some disadvantages related to memory space consumption. It is also essential to

emphasize the notable fact that the value of information decreases over time until it becomes negligible at some point in time. For this purpose, we can use the latter type of archive that uses mathematical and statistical functions to reduce size of useful data. The most used functions are mean, median or other functions (e.g. minimum, maximum, modus etc.).

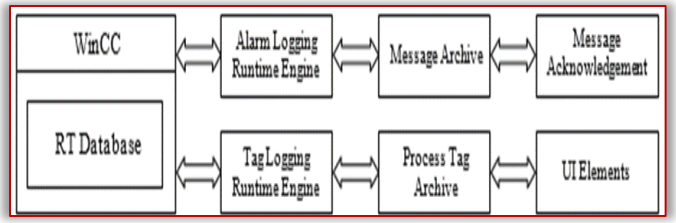

Figure 5. Alarm and Tag Logging architecture

— TIA PORTAL

Intensifying pressure on unification of fragmented development environments of Siemens Company converged to creation of an integrated platform called Totally Integrated Automation – TIA Portal [11]. In addition to programming of PLCs and Human Machine Interfaces (HMIs) in near future it will also be used for design of large scale SCADA applications.

Since the Object Generator works with PLC programs let us briefly discuss this aspect. The user defined program for PLCs is divided into blocks of the following types:

Organization Block (OB) – its main role is to create an interface between the operation system and the user defined program;
Function (FC) – the block without persistent storage, which means no results are stored from previous function call. It is used for encapsulation of routines;
Function Block (FB) – unlike FC, it has allocated memory used for sharing of results among calls, or between other blocks.
Data Block (DB) – it is used as persistent storage area for FB or in global context for all blocks;
System Function/System Function Block (SFC/SFB) – it groups the most used functions provided by the operation system of PLC.

— Data Blocks

A PLC uses data blocks (DBs) as the main part of communication with the SCADA system. The reason is that the process values are directly mapped in variables of data blocks. Data Blocks are divided into two main groups: global and instance ones. Global DBs are used to provide data persistency to all blocks (OB, FC, FB) and they are created by programmers themselves in required numbers.

The instance DB is created automatically at the moment when a new FB is created. Its structure (variables and attributes) is copied directly out of FB it is related to. Each variable of DB consists of:

Name – is used for the purpose of identification;
Data type – defines the size of the allocated memory and format of stored data;
Pre-defined value – is a starting value, in case of no initialization;
Offset – is a variable address restricted to the address space of DB. It is used for direct communication with upper layer systems, such as SCADA, HMI, ERP;
Comment – describes variable usage;
Retain – defines, if variable value should be backed-up in case of restart;
Others – there are other attributes to set, like Writable from OPC/HMI, Setpoint, Monitor value and many others.

— Interfaces description

Both, TIA Portal and WinCC, distribute their APIs as a standalone package, needed to be installed manually. When we talk about the TIA Portal, the API is called OPENNESS (Figure 6) and according to official documentation [7] it offers automation of engineering tasks related to PLC, HMI or project itself.

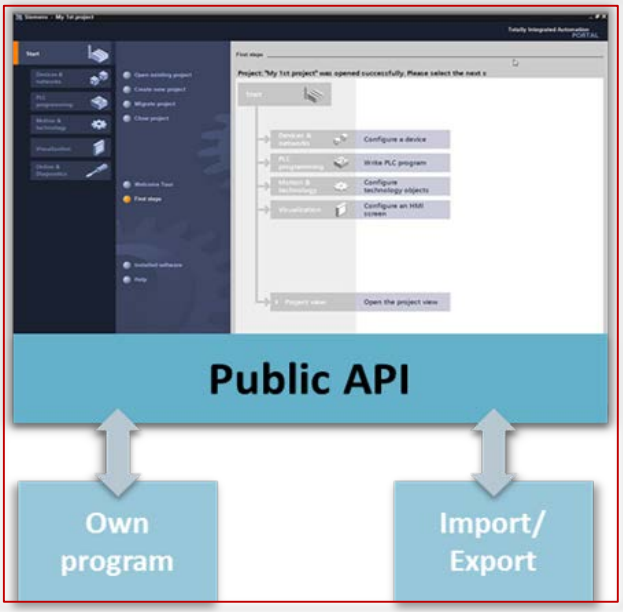

Figure 6. Engineering tasks automation using OPENNESS [10]

The current trend of application development under Windows platform focuses on usage of the C# language. The Siemens Company tries to keep pace with this trend and uses the C# as a language of choice in their TIA Portal application. Because of that, the API works only with the C# language and its implementation in user projects requires usage of Dynamic Linked Libraries (DLLs), which makes possible to interact with the API. Siemens also provides comprehensive documentation for this project.

On the other side, WinCC is developed in the C++ and is dated to 90's, which makes it much older than the TIA Portal. The API distributed for WinCC is called ODK, which stands for Open Development Kit and allows programmers to access data and functions of the WinCC configuration and the WinCC runtime system [9].

Mentioned facts about different languages of APIs make it rather difficult to develop user applications under one programming language. The key reasons are:

C++ does not offer automatic memory management and code written in it is also referred as unmanaged or native;
C# has built in Garbage Collector that takes care of memory handling. We refer to code written in C# as a managed code. Another important C# feature is Common Language Runtime (CLR) execution environment that allows language interoperability [6].

To overcome these differences between languages the authors were forced to use wrappers for C++ (Figure 7).

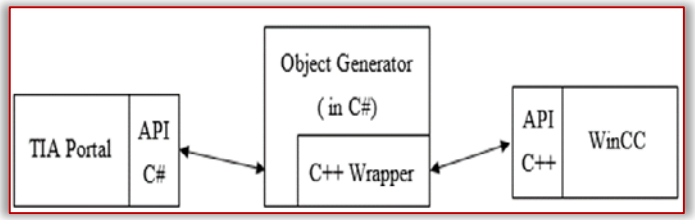

Figure 7. Usage of C++ Wrapper

REALIZATION OF OBJECT GENERATOR

The whole application is written in C#, which makes it easier to communicate with OPENNESS. On the other side we were facing complications referred in Chapter 3.2.

The realization took two steps [2]. The first step was to connect to the TIA Portal and gather all needed information about the project (Device list, Project structure, DB list and FB list). The second step was based on connection to WinCC and automated import of processed objects from the first step. Let us describe each stage of this development and realization.

— Connection to the TIA Portal

The graphical user interface of the Object Generator, shown in Figure 8, allows users to interact with the TIA Portal in several ways. We can open the 'clean' TIA Portal instance and connect to it, or we can manually list the project we want to open and then send it to the TIA Portal. These methods are related when a new instance of the TIA Portal is needed, but the situation changes when a user has already opened a project.

The Object Generator also treats this scenario and provides a simple way to list all available projects across all TIA Portal instances. When a user has chosen the right project to work with, another step consists in processing of gathered project data. The first step is to get all devices and their project structure; the result can be seen in Figure 9. These Data Blocks as mentioned before are not only used as a storage area within the PLC, but also for communication with the SCADA system. After a user chose devices and appropriate DBs he/she wants to work with, the processing initializes.

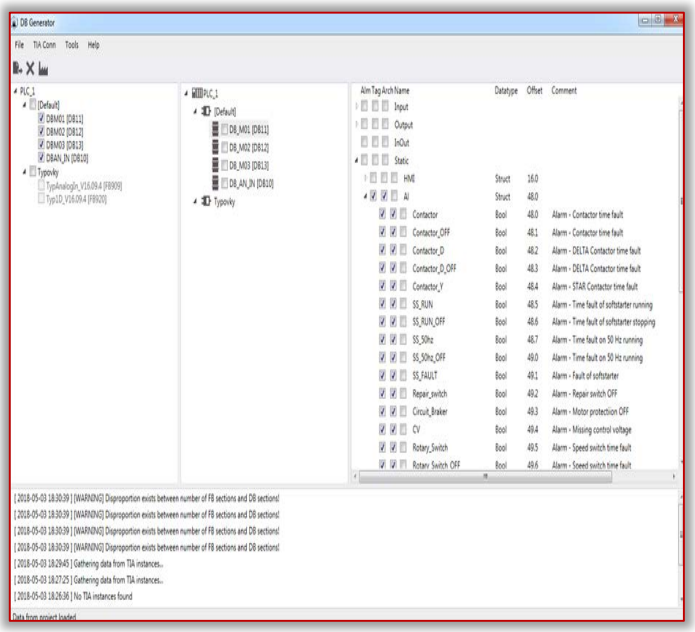

Figure 8. Graphical user interface of the Object Generator

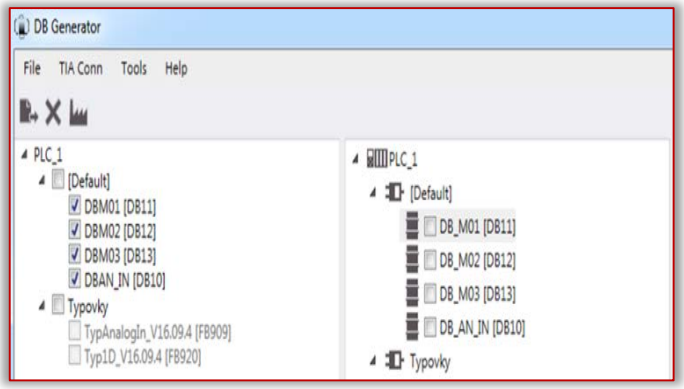

Figure 9. Devices and their blocks structure

In front of each variable there are three checkboxes that allow marking the variable as a tag, alarmed variable or archiving. Once the user checks all wanted variables, the last thing he/she must do is to set attributes for WinCC project.

The only possible way to interchange data with OPENNESS is by using XML structures. Therefore, processing is based on XML parsing and manipulation which extracts attributes such as names of variables within DB, for each variable its datatype, address and comment. Afterwards, the whole structure is presented to user in the form of a tree view (Figure 10).

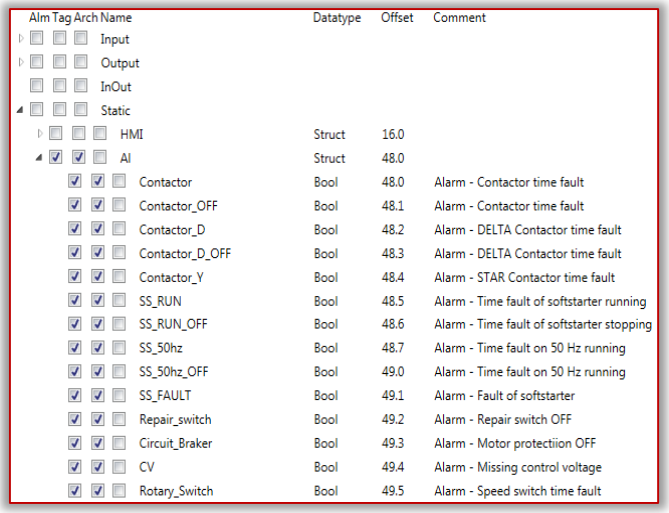

Figure 10. Tree view of variables and option boxes

— Connection to WinCC

In order to initialize import of tags to WinCC user needs to set several options. The first one is the path to WinCC project and in case of archived tags the interval of acquisition. For this purpose, the Object Generator provides a simple dialog window, shown in Figure 11.

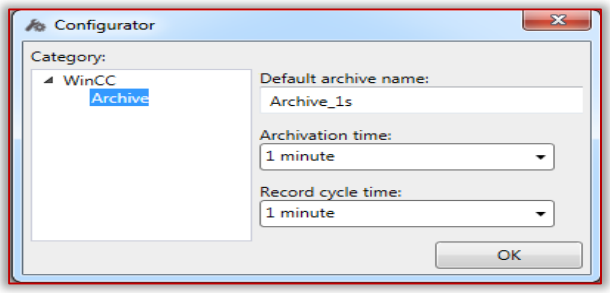

Figure 11. Configuration utility windows

— Other features

The Object Generator also provides a simple way of how to save all processed data and changes in the form of the XML file. The big advantage of this approach can be found in advanced possibilities for a skilled developer. In case of some minor changes in the PLC project, the SCADA developer could manually edit the configuration file with no need to get through mentioned procedures.

During ordinary operation of the Object Generator, many events and actions happen. To keep track of what was done and for case of troubleshooting the logger is available. User can find it at the bottom of the application window (see bottom of Figure 8).

CONCLUSION

The main advantage of the promoted application is a decrease of human-made-mistakes in the process of linking process data to internals of the SCADA system. This also enables reduction of errors, which could potentially be abused and exploited for a cyber-attack. Another benefit results from runtime import and export of raw data without any need to manually handle some arbitrary spreadsheet files. The saving feature helps a user to backup and save unfinished work for later.

The chance for improvements could be found in optimizing time consuming tasks such as XML processing and rendering of processed data. The whole architecture could also be refined and re-factorized with better feel for object-oriented design; using of design patterns, unit tests and better documentation within code.

Despite these potential future improvements, mentioned drawbacks and pitfalls made during development of this application offers great help to SCADA developers and let doors opened for further improvements.

Acknowledgments

This work was supported by the project ITMS: 26210120021, co-funded from EU sources and the European Regional Development Fund.

References

[1] Cassi, E. et al.: The Cyber Risk and Its Management in the Marine Industry. In Technology and science for the ships of the future. Proc. of NAV 2018: 19th International Conference on Ship & Maritime Research, 950-959, 2018.

[2] Guráň, M.: Creation of the Object Generator of a SCADA application based on a PLC program (in Slovak). MSc. Thesis No. 28260220182009, DCIS, Faculty of Electrical Engineering, UNIZA, 69 pages, 2018.

[3] Communication network dependencies for ICS/SCADA Systems. European Union Agency for Network and Information Security (ENISA), December 2017. https://www.enisa.europa.eu/topics/critical-information-infrastructures-and-services/scada

[4] Kalogeraki, E.-M.; Papastergiou, S.; Mouratidi, H. and Polemi, N.: A Novel Risk Assessment Methodology for SCADA Maritime Logistics Environments. Appl. Sci., 8 (1477), 1-32, 2018.

[5] Maritime Security Market- Segmented by Type (Screening and Scanning, Communications, Surveillance and Tracking, Detectors), Threat and Vulnerabilities (Deep Water Security, Perimeter Security), End-User (Military, Government Agencies, Coast Guards) and Region - Growth, Trends and Forecasts (2018-2023). Report, Mar 2018.

[6] Microsoft® Common Language Runtime (CLR) overview. April 2018. https://docs.microsoft.com/en-us/dotnet/standard/clr

[7] Siemens AG. Automating projects with scripts. 496 pages, May 2017.

[8] Siemens AG. SIMATIC HMI WinCC V7.4 - Getting Started. Entry ID: A5E37531782-AA, 236 pages, February 2016.

[9] Siemens AG. SIMATIC HMI WinCC V7.4 - Working with WinCC. Entry ID: A5E37536341-AA, 2584 pages, February 2016.

[10] Siemens AG. STEP 7 Professional V14 SP1. Entry ID: 109747136, 200 pages, April 2017.

[11] Siemens AG. TIA Portal OPENNESS: Introduction and Demo Application. Entry ID: 108716692, V1.2, 35 pages, May 2017.

[12] Saravanan, K.; Anusuya, E.; Kumar, R. and Son, L. H.: Real-time water quality monitoring using Internet of Things in SCADA. Environmental Monitoring and Assessment, 190 (556), 2-16, 2018.

[13] Ujvarosi, A.: Evolution of SCADA systems. Bulletin of the Transilvania University of Braşov, 9(58), No. 1, 63-68, 2016.

ISSN: 2067-3809