Implementing a Distributed Measurement System – a Case Study

V. D. Zaharia, D. M. Iudean, B. Ţebrean, F. Drăgan, T. E. Crişan

Faculty of Electric Engineering, Technical University of Cluj-Napoca, Romania

***Abstract** – The issue of organizing distributed monitoring and control devices, both from a physical topology point of view as well as from a logical connectivity perspective, represents a perpetual research topic in the field of applied Electric Engineering. The complexity of the issue is due to the multitude of equipment and devices coexisting on the market and due to their various interconnectivity possibilities. Interconnecting such devices, equipped with various communication interfaces and protocols, is achieved by implementing unified data transfer platforms, such as the OPC (Object Linking and Embedding for Process Control) platform.*

***Key words** – distributed, monitoring, control, Programmable Logic Controller, Object Linking and Embedding for Process Control (OPC).*

1. INTRODUCTION

The paper presents a case study based on the design and implementation of a distributed monitoring and control network, consisting of devices enabled with various communication protocols. Collecting the data from the distributed devices is done through the OPC Client-Server platform. The collected data are presented by means of customized HMIs (Human Machine Interfaces), and can also be stored and used for further studies and analysis.

The presented monitoring and control network consists of four FX3U Mitsubishi Programmable Logic Controllers (PLCs), a PC terminal as well as of devices for monitoring the electric energy quality parameters and consumption. Each of the PLCs is equipped with one or more input/output extension modules, special function modules as well as communication interfaces used in specific monitoring and control applications.

Due to the fact that a single PLC can be equipped with more communication interfaces which connect it simultaneously to more communication busses, the implemented monitoring and control network represents eight PLCs, which can be regarded individually. [2]

The designed and implemented network is meant to recreate a real situation which might occur in an industrial environment.

2. THE PROPOSED SYSTEM

In order to simulate a real-life situation regarding a complex monitoring and control networks being deployed, the proposed system consists of two main communication busses: the RS-485 bus and the Ethernet bus.

The RS-485 communication bus provides the possibility of connecting multiple devices without the need of auxiliary equipment (modems), while covering large distances with high transfer speeds (up to 100 kbs at 1200m). [1] These properties make RS-485 one of the most popular communication busses for scientific and process monitoring and control applications.

Networks based on the Ethernet bus consist of communication nodes and interconnection equipment. The Ethernet bus provides simultaneous connectivity options for multiple devices at high transfer speeds, over wide areas, with three complexity levels depending on the network topology: “point-to-point”, “bus” and “star”. [6][3]

In order to interface the two communication busses, a MOXA NPort Express DE–211 adaptor has been configured.

The devices connected to these busses are equipped with various communication interfaces, compatible with communication protocols used in industrial monitoring and control applications: CCLink, CANOpen, Modbus, ASi, Profibus, Mitsubishi FXNet.

The reasons for which various communication protocols coexist in the same monitoring and control network are numerous, one of them being the proprietary protocols some device manufacturers try to impose. Another reason for the variety mentioned above is the fact that these type of networks are flexible and scalable, as they follow the monitored processes' evolution, therefore the coexistence of communication protocols specific to different periods. Another reason for implementing different protocols in the same network is the variables' different degree of importance. Therefore it would be inefficient to implement high-speed large volume communication protocols in order to transmit slow-varying parameters or with a low degree of importance.

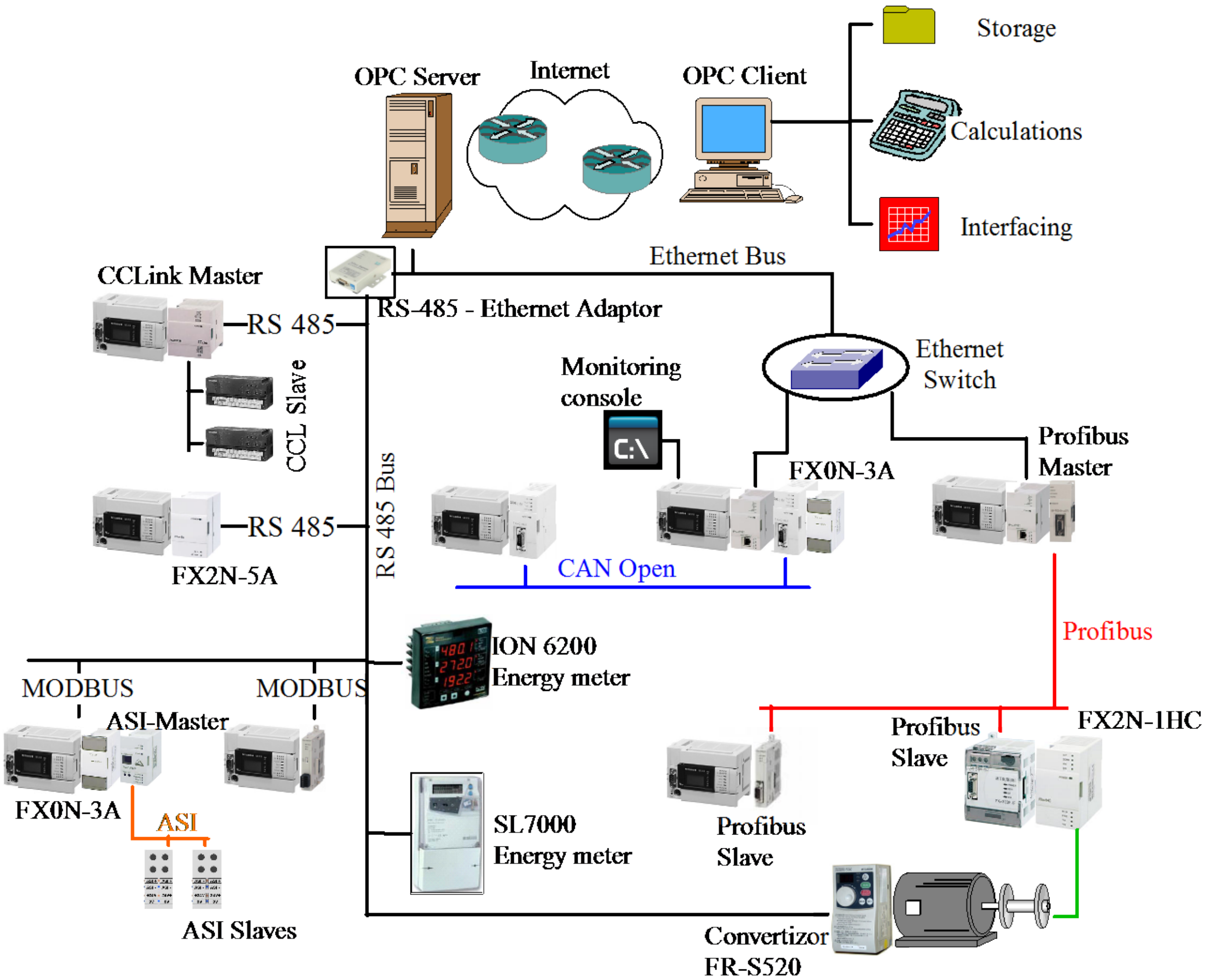

Fig. 1 The distributed measurement system

The challenge in such a diversified environment from the communication point of view is to manage the bidirectional data transfer to and from the distributed devices. In order to achieve complete bidirectional data transfer from the lowest levels of the distributed system up to the user interface level, the implementation of a unified communication platform is required.

Such a standard-based data-connectivity method can be supplied by using the OLE (Object Linking and Embedding) for Process Control (OPC) Specifications. The OPC Standard is used to overcome one of the automation industry's biggest challenges – enabling various data access by establishing communication between devices, controllers and applications, regardless of their proprietary interfaces and protocols. [4]

3. IMPLEMENTING THE OPC PLATFORM

The OPC architecture consists of two main elements: the OPC Server and the OPC Client applications. These two elements represent an intermediate layer in the communication process between a data source and a data sink. [4]

An OPC Server is a software application, a "standardized" driver, specifically written to comply with one or more OPC specifications. OPC Servers are connectors that may be thought of as translators between the OPC world and a Data Source's native communication protocol or interface. The OPC Client and Server applications relationship being bi-directional, this means OPC Servers can both read-from and write-to a Data Source. The OPC Client/OPC Server relationship is also a Master/Slave one which means one OPC Server will only transfer data to/from a Data Source if an OPC Client commands it to.

An OPC client software is written to communicate with OPC “connectors”. It uses messaging defined by an appropriate OPC Foundation specification. OPC Clients represent a data-sink. They initiate and control communications with OPC Servers based on what the application they are embedded in requests of them. OPC Clients translate a given application’s communication requests into an OPC equivalent request and send it to the appropriate OPC Server for processing. In turn, when OPC data returns from the OPC Server, the OPC Client translates it back into the application’s native format, so the application can properly work with the data.

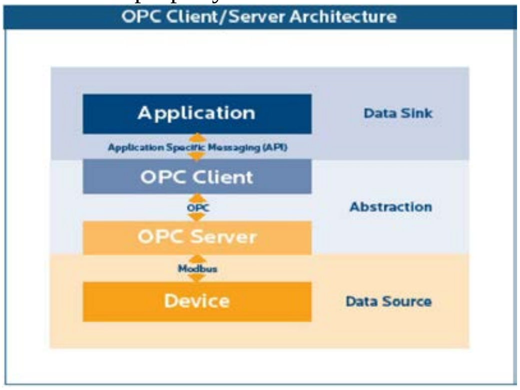

Fig. 2 The OPC Client/Server Architecture [4]

In order to connect the data acquisition system to a remote terminal for data presentation and logging purposes, an OPC Client/Server platform has been configured. [5]

The OPC Server allows the user to define communication channels with the distributed devices. Each channel is configured according to the remote devices’ communication interfaces and protocols. The next step of the OPC Server’s configuration is adding devices in their respective communication channels, which have been defined according to their specifications.

Finally, tags are defined for each device connected to the channel. The tags connect the OPC Server to memory registers in the remote devices. Each tag is corresponds to a parameter monitored by the remote device. In the OPC Architecture tags have three attributes: **value**, **quality** and **timestamp**.

Each device defined in the OPC Server application can parent one or more tag groups, which can be organized by the user in order to meet the particular needs generated by the structure of the system being supervised.

When configuring the OPC Client application, a series of steps have been followed. First, a connection to an active OPC Server must be added. Next, groups are added to the server connections, and finally items are added to the defined groups. Each item in the OPC Client is bound with a tag predefined in the Server application.

OPC Server and Client applications have been configured according to the structure presented above, in order to comply with the designed network’s structure.

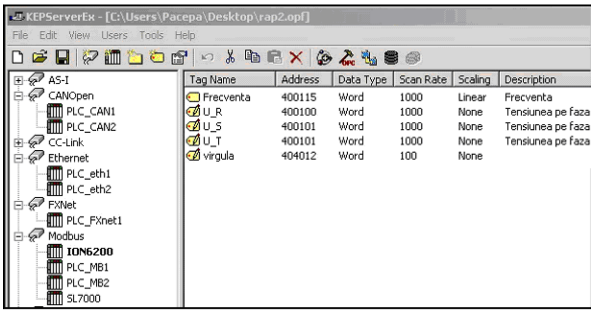

Fig. 3 The OPC Server configuration

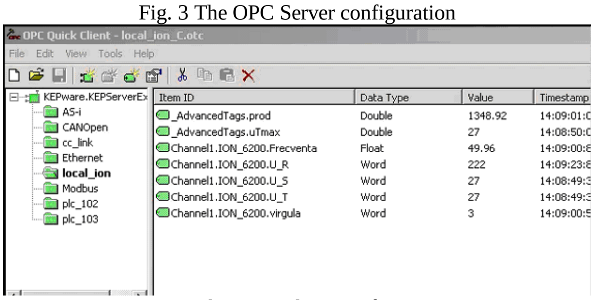

Fig. 4 The OPC Client configuration

The variables managed by the OPC Client-Server platform can be presented to the end user in various ways by means of customized HMI applications or by publishing them on-line. The data can also be stored in a database for further use.

In order to present the data acquired by the ION 6200 power meter, a custom HMI application has been devised, which presents the values for the electrical parameters monitored by the device.

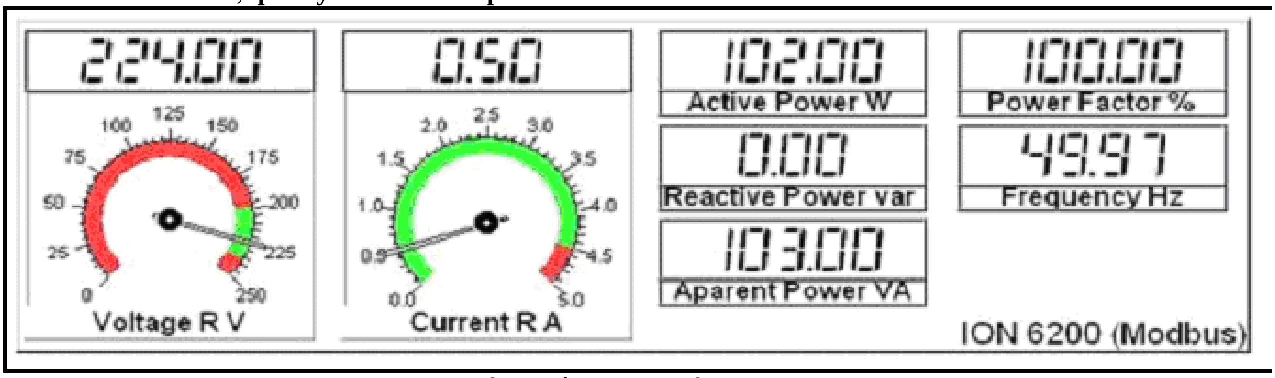

Fig. 4 HMI application for monitoring the ION 6200 power meter

Another HMI application developed for the network presents the states of the input and output ports of one of the PLCs, as well as the values for the voltage and current monitored by the mentioned device. The values are presented in numeric format and their evolution history is presented in a graphical form.

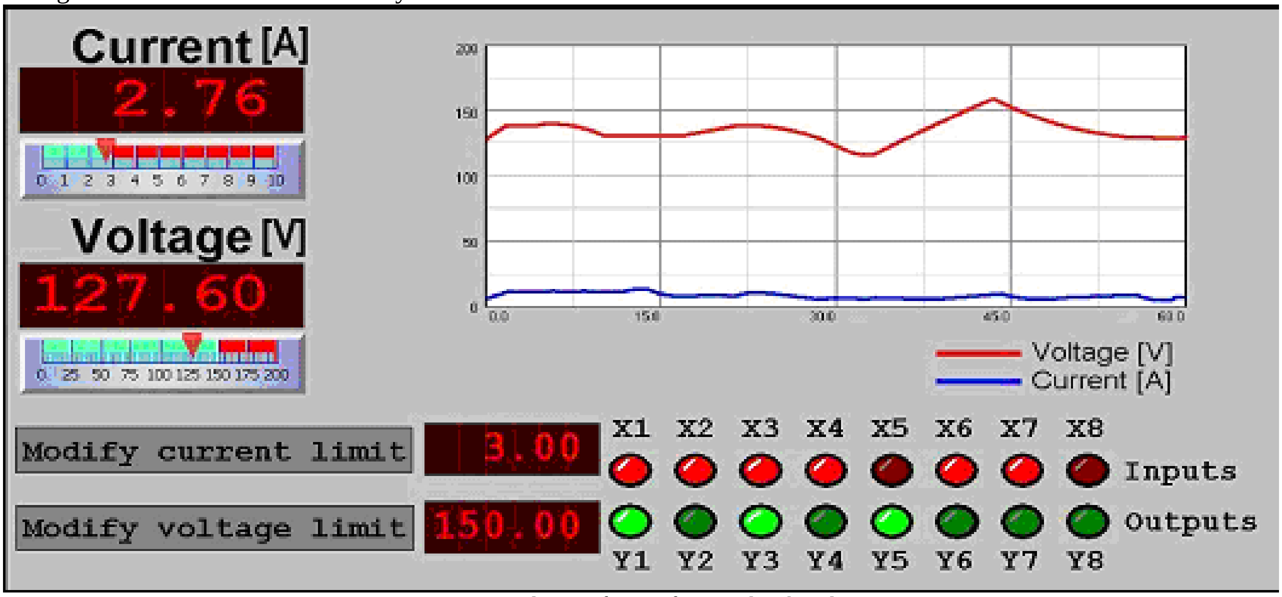

Fig. 5 HMI application for interfacing a distributed PLC

The acquired data can also be published on-line and due to security reasons it can be configured to be read-only.

4. CONCLUSIONS

The paper presents a complex monitoring and control network based on two communication busses and consisting of PLCs and other monitoring and control devices enabled with various communication protocols used on a large scale in industrial applications.

Due to the variety of protocols implemented in the network, the OPC Server-Client platform has been implemented and configured, allowing the data transfer between the network's levels, regardless of the communication protocols involved.

For data presentation purposes, customized HMI applications have been implemented, allowing bidirectional access between the network's layers.

REFERENCES

1. Axelson, J. – Serial Port Complete: Programming and Circuits for RS–232 and RS–485 Links and Networks, Lakeview Research, Madison, WI, 1998
2. Copândean R., Holonec R., Drăgan F. – Automate programabile. Modalităţi de programare şi aplicaţii, U.T. Press, Cluj–Napoca, Romania, 2010.
3. Drăgan, F. – Protocoale de comunicaţii, UT Press, 2008.
4. Kominek D., (2009) OPC: The Ins and Outs to What It's About, Matrikon Inc
5. OPC Foundation (2003) OPC Data Access Custom Interface Standard Version 3.00
6. Schwab, C. – Looking behind the automation protocols, The Industrial Ethernet Handbook, 2008.

Valentin Dan Zaharia
Faculty of Electrical Engineering, Technical University of Cluj-Napoca, 26-28, G. Bariţiu st., Cluj-Napoca, Romania
valentin.zaharia@ethm.utcluj.ro