Proceedings of the International Multiconference on
Computer Science and Information Technology pp. 791–796

ISBN 978-83-60810-27-9
ISSN 1896-7094

Supervisory control and real-time constraints

Wojciech Grega

Department of Automatics, AGH University of Science and Technology, 30-059 Kraków, Al. Mickiewicza 30, Poland
wgr@ia.agh.edu.pl

***Abstract*—OPC (OLE for Process Control) protocol was developed as a solution which fulfills requirements of open data integration architecture for industrial control. OPC standard is not primarily intended for feedback control or communication with high-bandwidth hard real-time requirements. Adding OPC to a process could influence the dynamics of the control loop and could cause problems in controller design and implementation. The experiments presented in this paper have shown that OPC if properly configured, is capable of providing a loop time shorter than the time constants of many industrial processes.**

I. Introduction

THERE are three major trends observed in contemporary industrial control systems:

- distributing and decentralizing structures of automation,
- increasing integration of communication through all the levels of the control systems supported by application of wired and wireless networks,
- growing demand for application of IT standards.

Most of the contemporary industrial automation systems adopts multilevel, vertical control architecture [1], [2]. The hierarchical, multilevel approach to industrial process automation is a well accepted method able to cope with complexity of the process and multiple criteria of operation. The structure from Fig. 1 represents a functional decomposition, as it is based on defining functionally different control objectives for vertically dependent layers.

Logically, a typical complex, industrial control system is structured into three levels (Fig. 1), which are: the direct (device) control level, supervisory level and process optimization and management level. Basic objective of the direct (device) control level is to maintain the process states in a safe operations mode according to the prescribed set values. Device controller level provides interface to the hardware, either as separate modules or as microprocessors incorporated in the equipment to be controlled. A number of embedded control nodes and Programmable Logical Controllers (PLC) are used as the front-ends to take the control tasks.

The supervisory level comprises workstations and industrial PCs providing, the high-level program support, database support, graphic man-machine interface, alarming and general management of computing resources. The partial objective of this level is to keep set values of device controllers or process inputs on prescribed values in order to meet demands of product quality or other economical constraints.

The supervisory level often interacts with optimization level. The objective of this level is to calculate the best values of process inputs. Its structure can be very complex and include such modules as identification module, simulation module or optimization module [2].

Usually direct control level operates with an intervention frequency significantly higher (i.e. sampling period is shorter) than the intervention frequency of the supervising level. In direct control sampling intervals can be in the range of miliseconds, whereas sampling interval of the process optimization level can be in the range of seconds or even minutes. This effect is justified by slow-varying disturbances shifting optimal high- level set-points, when compared to the device level feedback control dynamics.

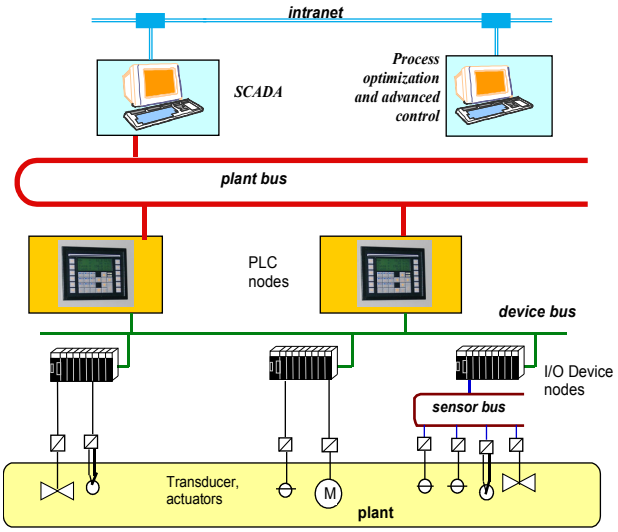

Fig.1 Multilevel structure of the industrial automation system

The ability to easily integrate information from control systems and „plant floor" measurements with supervisory and optimisation levels is a critical issue. It is very difficult to share data between various devices and software manufactured by different vendors without a common industrial communication protocol and integration standard to facilitate interoperability [3], [4].

This work was supported by the European Regional Development Fund Grant no. UDA-POIG.01.03.01-12-171/08/00.

The key is an open and effective communication and integration architecture concentrating on data access, not on the types of data. Ethernet comes as a solution to the data transmission problems mentioned above, while the OPC (OLE for Process Control) protocol was developed as a solution which fulfills requirements of open data integration architecture [5], [6].

In determining the capability for improving the plant operation through control methods implementation, it is important to understand where the delays occur and how these individual delay components contribute to the end-to-end communication. As a first step in this delay analysis, the impact of the network is assessed.

It is well known that the introduction of data transmission networks into the feedback loop in many cases violates conventional control theories assumptions such as non-delayed or evenly spaced sampling sensing and actuation. The computer network is characterized by its maximal throughput. This parameter limits the amount of data that can be sent within a time unit. Network-induced delays may vary depending on the network load and medium access protocol. Generally, networked control often introduces some additional temporal non-determinism. For distributed control systems variable queuing delays, transmission delays, transport layer ACK delays and the lost of data, leads to the deterioration of the quality of control [7]. Various methodologies were proposed for compensation of such the effects [8].

When using networks for control, it is often important to assess determinism as a QoS parameter. For example, evaluating whether end-to-end message communication times can be predicted exactly or approximately, and whether these times is bounded. Total end-to-end delay between network nodes is the sum of pre-processing time (microprocessor), waiting time (network protocol – MAC), transmission time (data rate & length), post-processing time (microprocessor).

Very little was said about the impact of OPC data processing on the dynamics of a control loop [9]. The overhead associated with OPC is included in pre (post) processing times and might be significant and variable for some configurations of this interface. Most of this delay is due to the software implementation of the OPC protocol, as OPC was not intended for hard real-time applications.

J. T. Parrott et al. from the University of Michigan [16] had investigated more deeply the delays induced by the application layer in three types of communication:

- UDP, User Datagram Protocol, a simple transport layer protocol,
- OPC, OLE for Process Control, an open application level communication protocol,
- VPN, virtual private networks, which create secure "tunnels" to transfer data between networks.

The Authors concluded that application layers delays are more important than network delays.

The main focus of this paper is in the usage of OPC in real-time control loop. It has already been mentioned that OPC has been designed to help with the interoperability problems inherent in a market with different propriety devices, protocols and industrial network standards. The question that needs to be asked is, which costs does this interoperability incur on the system in terms of performance? In order to achieve these goals a number of experiments was planned and performed. The results are given in the following sections.

II. Networks

Current communication systems for automation implements different protocols. This is a substantial disadvantage, leading to necessity of using vendor-specific hardware and software components, which increase installation and maintenance costs. Moreover, presently used fieldbus technologies make vertical communication across all levels of the automation systems difficult. Gateways need to be used to establish connections between different kinds of fieldbus systems used in the lower level, and Ethernet used in the upper levels.

Ethernet provides unified data formats and reduces the complexity of installation and maintenance, which, together with the substantial increase of the transmission rates and communication reliability over the last years, results in its popularity in the area of industrial communications.

Ethernet, as defined in IEEE 802.3, is non-deterministic and thus, is unsuitable for hard real-time applications. The media access control protocol, CSMA/CD with its backoff algorithm, prevents the network from supporting hard real-time communication due to its random delays and potential transmission failures. In real-time systems, delays and irregularities in data transmission can very severely affect the system operation. Therefore, various techniques and communication protocol modifications are employed, in order to eliminate or minimise the undesired effects.

To employ Ethernet in industrial environment, its deterministic operation must first be assured, which can be accomplished in several ways. Coexistence of real-time and non-real time traffic on the same network infrastructure remains the main problem. This conflict can be resolved in several ways, by [10]:

- embedding fieldbus or application protocol on TCP/IP – the fieldbus protocol is tunneled over Ethernet, and full openness for "office" traffic is maintained,
- using special Data Link layer for real-time devices – special protocol is used on the second OSI Layer, implemented in every device. The real-time cycle is divided into slots, one of which is opened for regular TCP/IP traffic, but the bandwidth available is heavily limited down,
- using application protocol on TCP/IP, direct MAC addressing with prioritization for real-time, and hardware switching for fast real-time,
- maintaining real-time on TCP/IP is achieved by prioritized messaging and time synchronization – the

synchronized devices assign higher priority and timestamp real-time messages,

All the specific techniques allow a considerable improvement in terms of determinism. The desire to incorporate a real-time element into this popular single-network solution has led to the development of different real-time Industrial Ethernet solutions, called Real-time Ethernet, as PROFINET, EtherCAT, Ethernet/IP [11], [12] and many more. The conditions for the industrial use of Ethernet are described by international standard IEC (*International Electrotechnical Commission*) IEC 61 784-2 *Real Time Ethernet,* Fig.2.

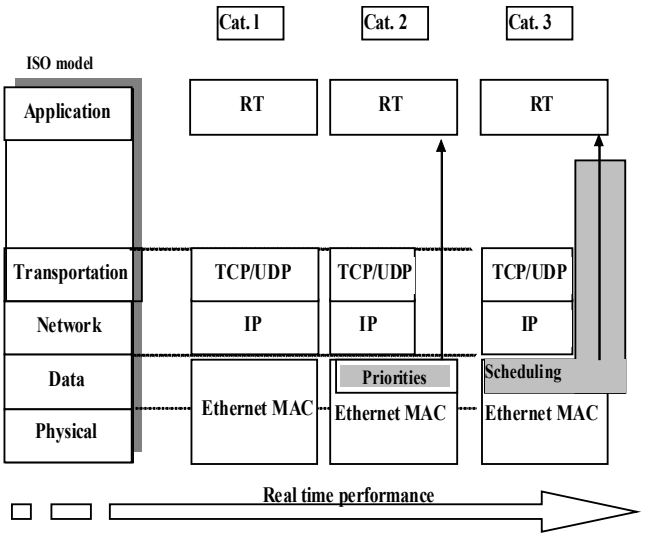

Fig.2 Classification of industrial Ethernet (IEC 61 784-2)

III. OPC

OPC [6] is a widely accepted open industrial communication standard that enables the exchange of data without any proprietary restrictions between multi-vendor devices and control software (horizontal integration) as well as between different software applications (vertical integration, Fig.3). Currently, most of plant-level control systems can be configured to be the OPC servers for supervisory control levels of the factory.

OPC is based on the Microsoft DCOM specification. Although OPC actually consists of many different data exchange specifications, its most commonly used form is Data Access (OPC DA), which supports both client-server and publisher-subscriber data exchange models. OPC DA deals only with current process data - not historical data or alarms.

An OPC client can be connected to several OPC servers provided by different vendors. Important feature of OPC is that the client is able to connect the OPC server which is located in other computer in the network. This is possible because of DCOM (Distributed COM) the COM standard extension which enables client running on one computer to create instances and invoke methods of servers on another computer within the network.

The OPC standard defines numerous ways to communicate between the server and its clients to satisfy different OPC applications. The client can specify that some operations should be performed on "cache" or "device". If "device" is chosen, operation directly on the physical device will be requested. If "cache" is selected data will be read from server's internal memory where server keeps a copy of device data received during the last OPC refresh cycle.

Both reading and writing can be done synchronously or asynchronously:

- OPC synchronous functions run to completion before returning. It means that the OPC client program execution is stopped till operation is over.
- OPC asynchronous functions use callback mechanism to inform the requested OPC operation is over. The client program execution is not stopped while waiting for OPC operation completed.

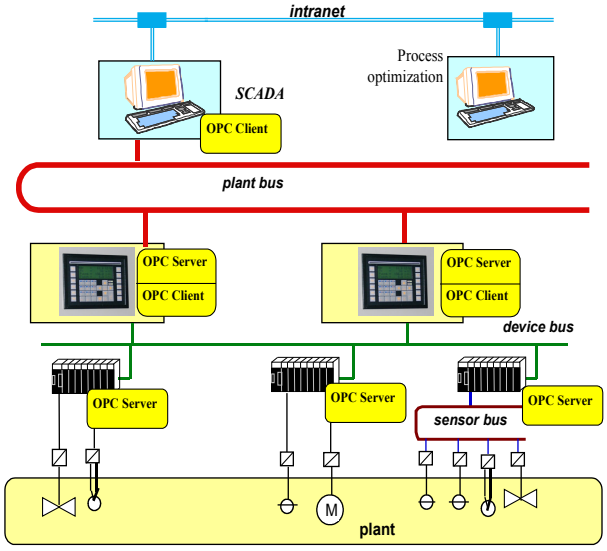

Fig. 3 Vertical integration with OPC

The OPC standard was not primarily intended for feedback control or communication with high-bandwidth, hard real-time requirements. However, many of I/O devices require real-time capabilities that are specific to the hardware and essential to proper operation of the high-level SCADA application. Therefore, one of the most important issues for communication between devices is to the assure proper configuration and synchronization.

IV. Experiments

The control algorithm must be designed according to a specification that provides the required performance when changes occur at the input or in the disturbances. For distributed control systems variable delays are considered to be the most important disturbances. For longer delays observed in the control loop it is much difficult to ensure the stability of the system within the response specification. Another factor that affects the operation of the control system is the variability of the delays, which again cannot

be too large without making the system unstable.

The control system considered in this paper has two data processing modules (Fig. 4): the OPC server and OPC client. The main objective of these experiments was to investigate how fast the OPC server responds to a client request and what is uncertainty associated with such a data exchange? The configuration shown in Fig. 4 describes idea of the experiments. Analogue Process Simulator (PS) have been used as a model of real controlled process. The device can simulate a third order linear process with different time constants.

The PS has two current inputs (4..20 mA) and one current output. The process control value was calculated in real-time in Simulink and send to PLC (Siemens) during the next OPC server cycle. In this configuration PLC was only a bridge between PC and Process Simulator. During each PLC controller program cycle process variables are read from PS and control variables are send to PS. The OPC server makes the variables available for OPC clients operating in the upper layer.

The Matlab OPC Toolbox™ was applied at the upper layer. It is a set of functions which make able to connect with an external OPC server from Matlab/Simulink programming environment. For the experiments described in in this paper only Simulink part of the OPC Toolbox was used. The PID controller block of Simulink was running using *pseudo real-time simulation* mode.

To configure the reading or writing block of the OPC client it is also important to decide the method how data will be captured from the OPC server. We had selected one of two options:

Synchronous read – MATLAB stops simulation and wait for the server to return data from a read request before continuing processing. The data returned by the server can come from the server's *cache*, or the server read values directly from the *device* that the server item refers to. Reading from the device can be slower but data are more time accurate.

Asynchronous read – MATLAB client sends a request to read data from the server. Ones the request is accepted it continue processing without waiting to receive values from server. Server will use callback to inform MATLAB about new data occurrence. MATLAB will handle that event as soon as it is able to perform that task.

With each OPC data read time stamp value is associated. Theoretically, distinction in time between successive time stamps should be equal to OPC item refresh rate (T_{ITEM}). In reality it varies depending synchronization method and CPU load.

Default server's cycle time T_{SERVER} for Siemens PLC was set as 00ms. OPC item refresh rate is specified in OPC Client and should be multiplication of OPC Server cycle time. OPC performance was calculated as sum of absolute errors between two successive OPC item read time stamps and OPC item refresh rate (T_{ITEM}) and divided by number of time stamps. The results of this calculation will presents the average time deviation of OPC item reading:

$$OPC_{Perform}=\frac{1}{N}\sum_{i=1}^{N-1}\left|\left(TimeStamp_{i+1}-TimeStamp_{i}-T_{ITEM}\right)\right|$$

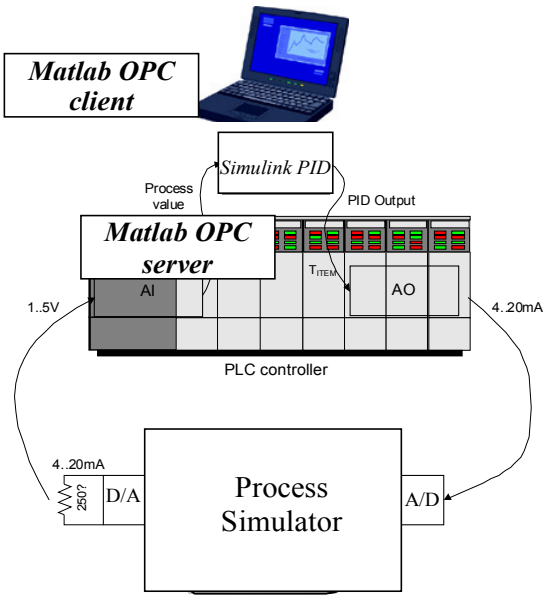

Fig. 4 Laboratory control system

The results of the above experiment are summarized in Table 1.

TABLE 1. OPC PERFORMANCE [13]

Configuration	OPC item rate T_{ITEM}=100ms, OPC server rate T_{SERVER}=100ms
	OPC performance
Synchronous cache read, synchronous write	8,72
Asynchronous cache read, asynchronous write	162,80
Asynchronous device read, synchronous write	162,39
Synchronous cache read, asynchronous write	13,31
Synchronous device read, synchronous write	6,99
Synchronous device read, asynchronous write	6,39

It can be seen for the delay of 100ms between readings, that the values of jitter (i.e. average of the deviation from the data mean delay) obtained for synchronous read are much lower compared to asynchronous operations. As it can be seen in Fig.5d the OPC client very often needs 500ms or even 700ms to asynchronously read data from the PLC (see Fig.6b). Notice, that while waiting for this event processing the controller is running and obsolete data are used for control. If one looks in details into asynchronous time stamps vector, he will recognize also, that several items have the same time stamp.

The lower jitter values obtained for synchronous mode indicate the significance of proper configuration of OPC interface.

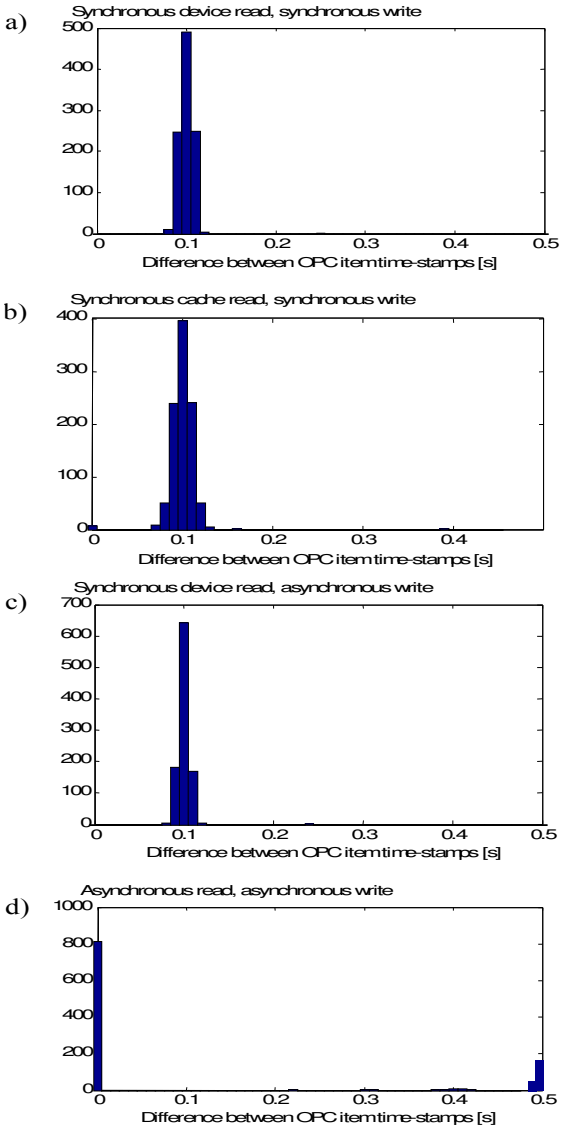

Fig.5 OPC jitter

V. Influence of network traffic

In this part impact of network traffic on data delivery was tested. PLC and PC with MATLAB were connected each other via hub and additionally, three other computers were connected to the hub. Between additional computers artificial TCP traffic was generated. OPC Server refresh rate and OPC item refresh rate were set to T_{ITEM}= =T_{SERVER}=100ms. One of the tests was performed using wireless (WiFi) connection. The results are presented in Table 2.

Digital control systems are based on periodic operations. A shortest sampling period is limited by OPC server refreshment rate. It must guarantee acceptable control performance. The network communication delays and jitters will be added to the OPC delays and jitters, creating total

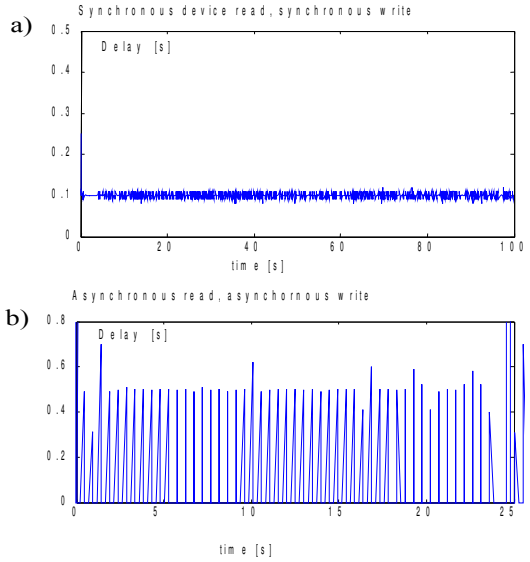

Fig.6 Time delays for synchronous and asynchronous modes

Table 2. Jitter introduced by network [13]

Network	Synchronous		
	Mean jitter [ms]	Jitter highest [ms]	Jitter lowest [ms]
Ethernet	0.4231	0.4323	0.493
Wireless	0.7222	0.8889	1.7778
TCP Ethernet with traffic 50%	5.7105	20.7895	24.4737

delays and uncertainty in the control loop. General conclusion after this experiment is, that low network traffic is not influencing very much the time accuracy of data delivery. In this case OPC jitter is more significant.

Network type jitters and delays dominate for heavy traffic. In this case the highest jitter gets 20% of T_{ITEM} time and might influence on control quality i.e. to violate conventional control theories assumptions such as evenly spaced sampling sensing and actuation.

VI. Conclusions

Various requirements for distributed, real-time control of a process in process automation can be formulated, for example stability, speed of response etc. Most important is to ensure that the data required by the controller are delivered to the controller and for the data from the controller to the process again, in less time that it takes for the any appreciable change to occur in the output of the process. With the industrial network speeds that increase application layer delays can become more important than network delays. As a consequence, application layer delays must be taken into account but they are often not specified.

OPC is a standard that has been developed to help with inoperability problems with the number of different devices and communications protocols in the process industry. OPC was never intended for a hard real-time operation. The

latency and jitter associated with OPC (and DCOM in general) is significant. Even though data read from an OPC server bears timestamps, there is no guarantee that the client will receive them in a reasonable time. Therefore, adding OPC to a process could influence the dynamics of the control loop and could cause problems in controller design and implementation.

There are a number of researches focused on COM technology to demonstrate that the COM/DCOM model after some extensions can guarantee real-time communication to industrial control systems [14], [15].

Our experiments have shown that standard OPC DA in asynchronous mode has slow and unpredictable read times. However, if properly configured, the OPC is capable of providing a loop time less than the time constants of many industrial processes. Especially, it is useful for supervisory control level, where sampling intervals can be in the range of seconds or even minutes. It can be also used to send set-point commands down from the SCADA to the device level. OPC standard has its place in the distributed control configurations of all processes that do not have fast responding outputs.

ACKNOWLEDGMENT

Author thanks Mr. Marcin Bajer for valuable experiments he performed as a ;part of his Master Thesis [13].

REFERENCES

[1] P. Tatjewski, *Advanced Control of Industrial Processes*. Springer-Verlag, London, 2007.

[2] W. Grega, W. Byrski and J. Duda, "InStePro – Integrated Production Control", Available (2010): http://www.InStePro.agh.edu.pl .

[3] W. Grega, "Design of Distributed Control Systems: from Theoretical to Applied Timing", in *Recent advances in control and automation*, Warszawa, pp. 343–352, 2008.

[4] L. Almeida, "Networks for Embedded Control Systems", in: *ARTIST2 China Summer School, Shanghai*. Available ((2010): http://www.artist-embedded.org/docs/Events/08/China.

[5] [Online]. Available (2010): http://www. opcfoundation.org.

[6] [Online]. "OPC DA 3.00 Specification. OPC Foundation", Available (2010): http://www. opcfoundation.org.

[7] W. Zhang, M. Branicky and S. Philips, "Stability of Networked Control Systems", *IEEE Control System Magazine*, vol. 21, pp. 84-99, 2001.

[8] W. Grega, "Stability of Distributed Control Systems with Uncertain Delays", in *8th IEEE International Conference on Methods and Models in Automation and Robotics*, pp. 303 – 307, 2002.

[9] N. Kalappa, J Moyne, J Parrott and Y.Shian Li, "Practical Aspects Impacting Time Synchronization Data Quality in Semiconductor Manufacturing", In: *Proceedings of the IEEE 1588 Conference*, October 2006.

[10] L. Larsson, Technical article: Fourteen Industrial Ethernet solutions under the spotlight, *The Industrial Ethernet Book* – 2005 – 28 – Available (2005): http://ethernet.industrialnetworking.com/articles/articledisplay.asp ?id=854.

[11] M. Felsner, "Real-Time Ethernet – Industry Prospective", in: *Proceedings of the IEEE*, vol. 93, pp. 1118- 1129, June 2005.

[12] D. Jansen and H. Buttner, "Real-time Ehernet the EtherCAT Solution", *Computing & Control Engineering Journal*, vol. 15, pp. 16 – 21, Jan. 2006.

[13] M. Bajer, "Control Systems Integration using OPC Standard", AGH Master Thesis, W. Grega -Supervisor, Krakow &Antwerp 2008.

[14] Paul Fischer, "Real Time Extensions to OPC", *Real Time Magazine*, vol. 98Q3, p .76.

[15] Y. Veryha, "Going beyond performance limitations of OPC DA implementation", In *Proceedings of the 10th IEEE Conference on Emerging Technology and Factory Automation*, pp. 47−53, 2005.

[16] 16. J. T. Parrott, J. R. Moyne, and D. M. Tilbury, "Experimental Determination of Network Quality of Service in Ethernet: UDP, OPC, and VPN", In *2006 American Control Conference, ACC'06*, Minneapolis, USA, June 2006.