ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

*International Journal of Advanced Research in Computer and Communication Engineering*
*Vol. 4, Issue 9, September 2015*

# Integration of the Data Distribution Service middleware protocol in an SCADA system

**Ioan Ungurean[1], Nicoleta Cristina Gaitan[2]**

Faculty of Electrical Engineering & Computer Science, Stefan cel Mare Univesity of Suceava, Suceava, Romania[1,2]

**Abstract**: SCADA systems are widely spread in automation, monitoring and control systems for industrial processes. These systems are used to develop the factories of the future. Currently, the most SCADA systems allow data sharing only within a local network. With the development of the Internet of Things concept, it was started to be used in the systems for the factories of the future. One of the most powerful middleware systems used for the Internet of Things is these based on the DDS middleware systems. In this paper, it is presented the integration of the DDS middleware based systems within an SCADA system in order to enable the Internet of Things concept.

**Keywords**: Internet of Things, middleware, data distribution service, SCADA.

## I. INTRODUCTION

The Supervisory Control and Data Acquisition (SCADA) systems [1] are software/hardware systems designed to acquire information from a specific physical environment through sensors and to provide a graphic interface to the human operators. Through these graphic interfaces, the operators can monitor and can send commands to actuators, which are installed in the physical environment [2]. These systems are the most used to monitor industrial processes or within the smart home. The acquired information can be transmitted to other systems such as the Enterprise resource planning (ERP) [3] and Manufacturing Execution System (MES) [4]. At this time, SCADA systems are found in all industrial branches.

SCADA systems typically consist of three main components: Remote Terminal Units (RTU), Master Terminal Units (MTU), and Human–Machine Interface (HMI) [1]. The RTUs are devices from the process (PLCs, field devices or actuators) interconnected via a fieldbus [5]. MTUs are hardware/software systems that communicate with RTU and distribute information to the HMI modules through the local network or the Internet. Most SCADA systems are middleware-based [6] in sense that data distribution within a local network or even the Internet is performed through middleware systems. With the development of the Internet of Things concept [7], it has emerged a new generation of SCADA systems that integrate this concept using middleware systems that allow data transport throughout the Internet infrastructure.

In the SCADA systems, the most common middleware systems used to distribute data are based on OPC specifications [8](OPC DA, OPC .NET and OPC UA). DA OPC uses DCOM technology from Microsoft (is platform dependent), OPC .NET uses WCF technology Microsoft (is platform dependent), and OPC UA uses SOAP and web services technologies (OPC UA is platform independent) [8]. With the development of IoT (Internet of Things) concept [9], middleware specific to this paradigm can be used for the development of the SCADA systems in order to activate IoT concept.

One of the most popular middleware systems used for IoT concept is DDS (Data Distribution Services for real time systems) [10]. These middleware systems were designed for critical and real time systems and are suitable for SCADA systems. In this paper, it is proposed an SCADA solution based on DDS middleware that was designed by adapting an SCADA solution [11] originally based on OPC specification.

Further, this paper is structured as follows: Section II shortly presents the DDS middleware standard, and Section III presents the proposed SCADA solution. The conclusions are drawn in Section IV.

## II. DDS – DATA DISTRIBUTION SERVICES FOR REAL TIME SYSTEMS

DDS is a middleware protocol data centric that was originally designed for critical systems. He began to be used for IoT systems because it allows the development of distributed systems throughout the Internet. DDS is sustained by the Object Management GROUP [12], and the last update was performed in April 2015. Since he was designed for critical systems, it provides a set of QoS parameters that can be set the communication [13] depending on the specific environment in which application work. This can be a major advantage in using the protocol for monitoring critical industrial processes in the detriment of OPC specifications. Since the first version was published in 2003 and currently is more than 10 implementations of the protocol [14], it can be said that is a mature protocol. DDS works on the producer consumer paradigm in the sense that to data from one producer can be connected more data consumers. Therefore, it can be created peer to peer architecture, architecture that can be more faults tolerant and scalable towards the client-server architectures. DDS is organized by domains and topics. A domain can contain multiple topics of read or write type. A topic structure is defined by type IDL files that must be compiled and included in each application. Practically these IDL files define the communication interface between the two topics. A significant disadvantage, compared to OPC specification, is that these communication interfaces are not standardized.

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

*International Journal of Advanced Research in Computer and Communication Engineering*
*Vol. 4, Issue 9, September 2015*

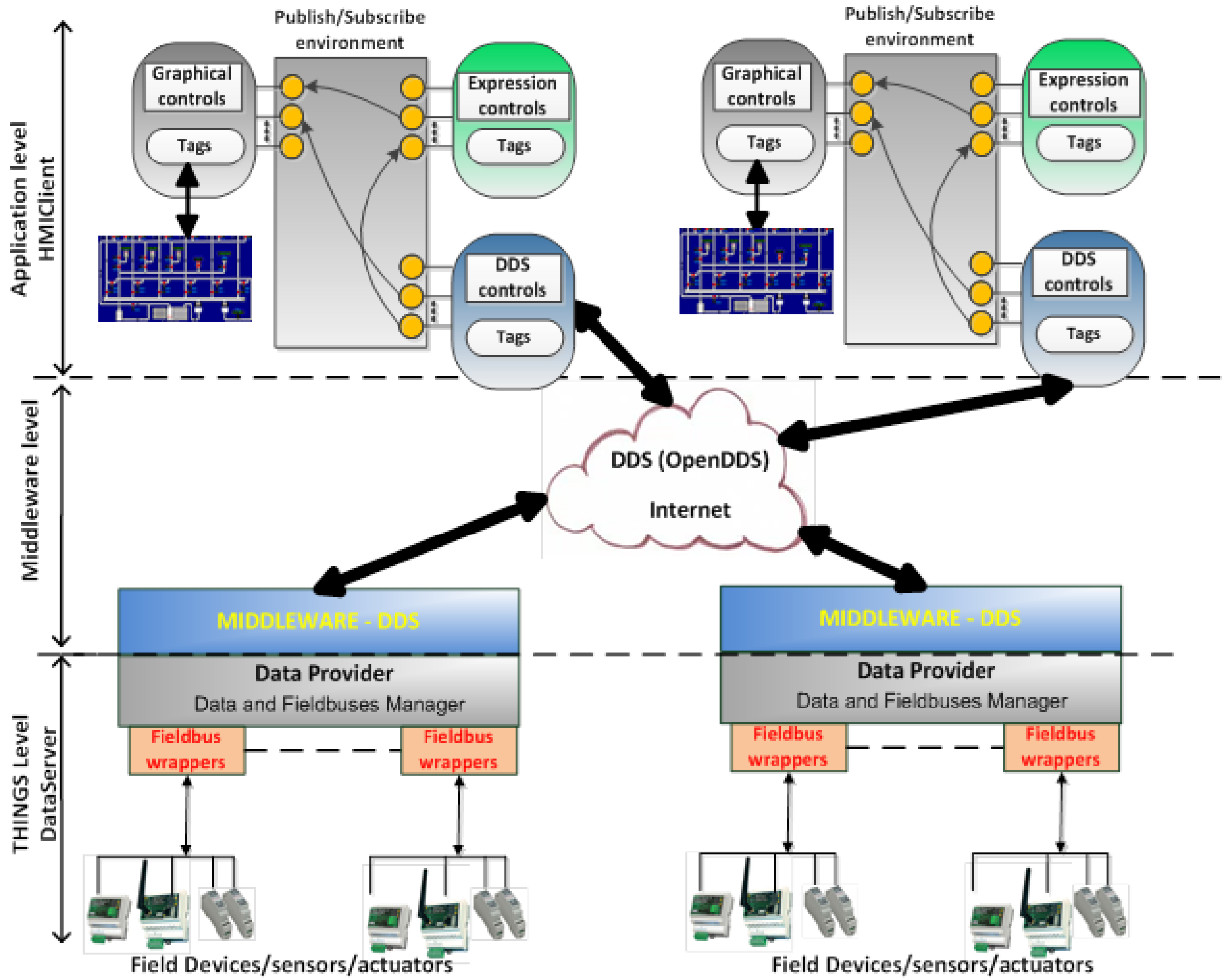


Fig. 1. The proposed SCADA architecture based on DDS middleware protocol

## III.INTEGRATION OF THE OPENDDS IN A SCADA SYSTEM

Starting from the architecture presented in [11], it is intended to adapt it in order to include the DDS middleware protocol. For the DDS, it was chosen the OpenDDS implementation [15] because is open source, and it is developed in C++.

The resulted architecture from the redesign process is shown in Fig. 1. The architecture is now organized on three main levels: application, middleware and things. This paper is focused on the middleware level.

The architecture contains two software applications: DataServer and HMIClient applications. The DataServer includes middleware and things levels, and HMIClient includes the middleware and application level. By including DDS protocol in the architecture, now it can be interconnected two or more DataServer without involving the HMIClient application. This allows the creation of the more scalable network architectures towards the original solution presented in the [11].

The HMIClient applications can be interconnected without being brokered by the DataServer applications.

The DataServer application was developed in C ++, and for this reason, the integration of the OpenDDS middleware was performed much easier. The first step was the definition of the structure of a topic. In this case, a topic structure is similar to the structure of an item defined in the OPC specifications. Such a topic contains a unique identifier (a string that can identify physical/virtual things that read/write data), the data type of the value (logic, string or numeric), value, quality of the value (GOOD or BAD), and a timestamp associated to the value (date and time when it was generated the value).

All such information is also provided for an item in the OPC specifications. It was adopted this solution in order to integrate the field buses in the same way as the field buses in the original system. Furthermore, in the application there is a module which allows the configuration of the topics that will publish data in a DDS domain and topics that can subscribe to the topics within the DDS domain. By introducing this module, it can be controlled what is published from the information that is acquired on field buses and what data can be transmitted to the field buses (information that is retrieved from the topics of the associated DDS domains). This method can directly interconnect DataServer applications.

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

IJARCCE

*International Journal of Advanced Research in Computer and Communication Engineering*
*Vol. 4, Issue 9, September 2015*

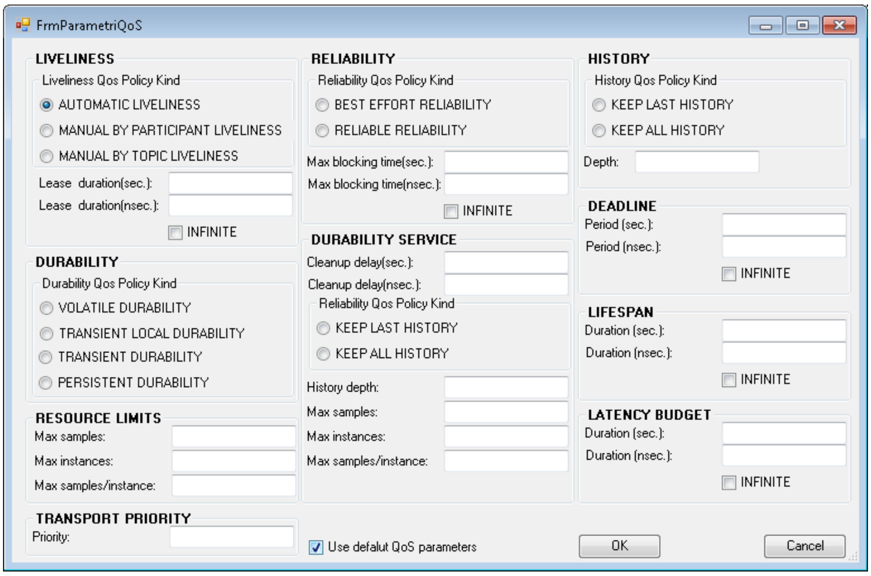

Fig. 2. QoS parameters for a DDS domain

The HMIClient application is described in detail in [6]. In the existing application it was introduced an OpenDDS object that allows the publication of data within a DDS domain and subscription to data producers within the same DDS domain. As well as in the DataServer, each OpenDDS object has a module in order to configure what data is published and to what topics are subscribed in order to bring information in the HMIClient application to be displayed in a graphical manner. Each OpenDDS object can be assigned to a single DDS domain, and if it is desired the publication of data in another DDS domain, then it must be created/instantiated another OpenDDS object.

Both applications have in the configuration module the option to set a series of QoS parameters at the domain level. Each topic within a domain will use the QoS parameters set for the domain. The window from the DataServer and HMIClient which allows the setting of the QoS parameters is shown in Fig. 2.

## IV. CONCLUSION

In this paper, it is proposed an architecture for an SCADA system based on the OpenDDS implementation of the DDS middleware standard. The architecture is designed starting from an architecture based on the OPC specifications. The use of the DDS middleware allows the distribution of information via the Internet, and because DDS is a device to device (D2D) middleware, it can enable the concept of the Internet of Things. An advantage is the QoS parameters set that are provided by DDS in order to tune the communication, which may be specific to the monitored/controlled processes.

## ACKNOWLEDGMENT

This paper was supported by the project ―Sustainable performance in doctoral and post-doctoral research PERFORM–Contract no. POSDRU/159/1.5/S/138963, project co-funded from European Social Fund through Sectorial Operational Program Human Resources 2007-2013.

## REFERENCES

[1] Boyer, Stuart A. SCADA: supervisory control and data acquisition. International Society of Automation, 2009.
[2] Smith, Sidney C. A Survey of Research in Supervisory Control and Data Acquisition (SCADA). No. ARL-TR-7093. ARMY RESEARCH LAB ABERDEEN PROVING GROUND MD COMPUTATIONAL AND INFORMATION SCIENCES DIRECTORATE, 2014.
[3] Xu, Li Da. "Enterprise systems: state-of-the-art and future trends." Industrial Informatics, IEEE Transactions on 7.4 (2011): 630-640.
[4] Knapp, Eric D., and Joel Thomas Langill. Industrial Network Security: Securing critical infrastructure networks for smart grid, SCADA, and other Industrial Control Systems. Syngress, 2014.
[5] Zurawski, Richard, ed. Industrial communication technology handbook. CRC Press, 2014.
[6] Ungurean, Ioan, Nicoleta Cristina Gaitan, and Vasile Gheorghita Gaitan. "Transparent interaction of SCADA systems developed over different technologies." System Theory, Control and Computing (ICSTCC), 2014 18th International Conference. IEEE, 2014.
[7] Holler, Jan, et al. From Machine-to-machine to the Internet of Things: Introduction to a New Age of Intelligence. Academic Press, 2014.
[8] Schwarz, M. H., and Josef Borcsok. "A survey on OPC and OPC-UA: About the standard, developments and investigations." Information, Communication and Automation Technologies (ICAT), 2013 XXIV International Symposium on. IEEE, 2013.
[9] Da Xu, Li, Wu He, and Shancang Li. "Internet of Things in industries: A survey." Industrial Informatics, IEEE Transactions on 10.4 (2014): 2233-2243.
[10] DDS Portal – Data Distribution Services, http://portals.omg.org/dds/
[11] Gaitan, Vasile Gh, et al. "Keeping industrial systems and communicators up-to-date using interoperable communicating components and electronic data sheets." EUROCON 2009, EUROCON'09. IEEE. IEEE, 2009.
[12] Object Management GROUP, http://www.omg.org/
[13] Hoffert, Joe, Douglas Schmidt, and Aniruddha Gokhale. "A QoS policy configuration modeling language for publish/subscribe middleware platforms." Proceedings of the 2007 inaugural international conference on Distributed event-based systems. ACM, 2007.
[14] H. Perez, J.J. Gutierrez, "A survey on Standards for real-time distribution middleware" Journal ACM Computing Surveys, vol. 46, issue 4, March 2014.
[15] Computing, Object. "Inc. OpenDDS Developer's Guide.", http://download.ociweb.com/OpenDDS/OpenDDS-latest. pdf.