Some Solutions for Improving OPC UA Performance

Marcin Fojcik[1(✉)], Olav Sande[1], Martyna Katarzyna Fojcik[2], Are Sjåstad Bødal[1], Tor Erik Haavik[1], Kristoffer Hjartholm Kalstad[1], Bjørnar Brask Sittlinger[1], and Torbjørn Ryland Steinholm[1]

[1] Western Norway University of Applied Sciences, Bergen, Norway
{marcin.fojcik,olav.sande}@hvl.no,
{246941,160260,160271,160262,160264}@stud.hvl.no
[2] Volda University College, Volda, Norway
martyna.fojcik@hivolda.no

Abstract. The need to exchange more data in industry is growing. Automation requires more and more data to adapt technology to industrial needs correctly. This is visible in both industry (Factory 4.0) and in the Internet of Things (IoT). All these data are produced, transferred, processed and stored for further use. An affordable communication system that permits all of the necessary operations to be performed quickly and reliably is needed. One of the proposals is OPC UA. Unfortunately, there are many situations in which OPC UA is not efficient for various reasons. One of them is the lack of real-time communication in OPC UA. Another situation that is not suited for OPC UA is the setup of the server and incoming data.

In this paper, the authors will try to present some of the limitations and propose ways in which to improve the time parameters of OPC UA systems in practical applications.

1 Introduction

Nowadays, there are new technologies and factories that can design each product to meet customer requirements without increased costs relative to a large-scale production series. Figure 1 presents a typical production facility that is organised according to ISA95 [1]. The customer orders (and pays for) a product electronically using some type of web technology. The Enterprise Resource Planning (ERP) system confirms the order, adapts the order to the capabilities of the production control system and transfers the modified order to the Manufacturing Execution System (MES). The MES converts the order into production instructions and transmits the production instructions to the Human Machine Interface (HMI) for further modifications and verifications in real-time. Ideally, the MES transmits production instructions directly to the Production Control System (PCS) without any human interaction. The PCS automatically adapts and coordinates the production assets in order to produce the ordered product.

N. T. Nguyen et al. (Eds.): ICCCI 2019, LNAI 11684, pp. 249–258, 2019.
https://doi.org/10.1007/978-3-030-28374-2_22

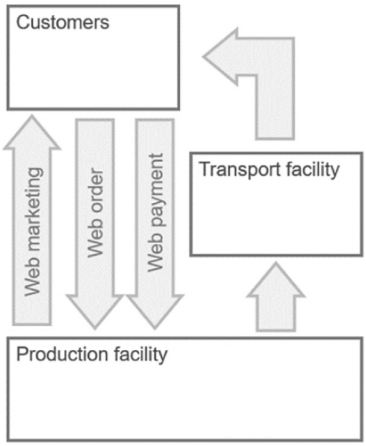

Fig. 1. General view on production method

During the production process, the product communicates interactively with the PCS, which makes the production process highly dynamic.

The communication protocol for data exchange in an I4.0-compliant plant is OPC UA (Fig. 2) [2,3]. The OPC UA (Unified Architecture), protocol is highly flexible with respect to data modelling, which is necessary in an I4.0-compliant production facility [4]. Models are dynamic with model views so that the production system elements and production assets can adapt and expose their model depending on the production context.

OPC UA can also be a protocol that is used by the production system to communicate with the customer. The production system exposes the product to the customer as a dynamic view of the complete OPC UA model, the customer modifies the OPC UA view and thereby the product features. The production system capabilities limit the possibilities of customer modifications. During production, the customer can dynamically interact with and modify this view as long as the actual production process permits it. If a product feature is pending in the manufacturing process and there is no conflict with the already manufactured features, the customer production system permits the customer to alter the product features without or with little or no impact on the costs. The production system dynamically exposes the state of the product in OPC UA view to the end customer through the constant flow of information from the production assets to the customer [5].

OPC UA has a communication that corresponds to the addressing memory model on the server. The model fits the hardware and helps with the compatibility and cooperation between devices. On the other hand, OPC UA is not effective in sending information to many clients quickly [6,7].

One important parameter on the communication system, between the server and the client, depends on the quality of the transmission of the data. This

quality can be measured by certain network communication features such as throughput, transmission delay, jitter etc.

This paper describes some of the parameters and attempts to find a solution to improve them. The paper is divided into six sections. Section 2 presents the OPC UA standard. Then, Sect. 3 shows some methods for data flow that are described in the standard. Section 4 describes the experimental stand. The results from tests are presented in Sect. 5. Section 6 is a summary and ideas for future research.

2 The OPC UA Standard

The wide variety of industrial automation and communication systems presents a technological challenge. The harmonisation and exchange of data between the different components of industrial enterprises is required. All of these activities are called Factory 4.0. [8] OPC UA plays a key role in this process and is a widely recognised standard in industrial automation in terms of interoperability and data exchange from the sensor level, through control and communication systems to production management and even to the cloud. The goal of OPC UA is to achieve uniform, standardised and secure communication between suppliers, devices and automation levels. In order to achieve this, OPC UA provides both a communication model and an addressing model in one system. OPC UA is based on a Client/Server architecture and it uses an object-oriented data model. All of the available services are independent of the specific protocols. An additional advantage is the object-oriented data structure, which combines data from several sources and the relationship between the information as the semantic model of the data (Fig. 2).

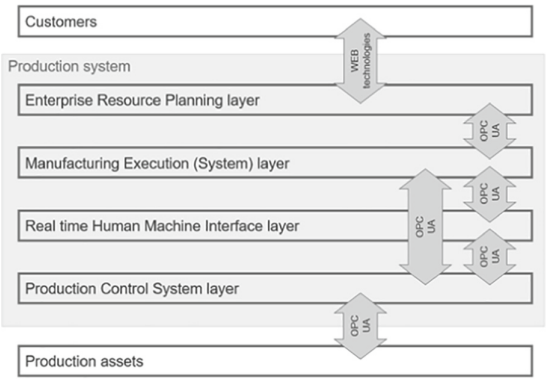

Fig. 2. Structure of production with OPC UA

The disadvantage of OPC UA is that because of the service-oriented structure, it is not easy to calculate the time requirements. In many situations, all of

the advantages of OPC UA cannot balance the problems with the failure of data delivery. If a system cannot deliver the data on time, it means that the system is not useful [7]. There are some new extensions [9] in OPC UA standard such as the PubSub service (Fig. 3) or Time Sensitive Network (TSN) [10]. Both of these improve data communication between the Server and Clients. The Pub-Sub service separates the Server and the Client by using Middleware. The server sends the data to the Middleware first, after which all f the Clients can read the required data from the Middleware.

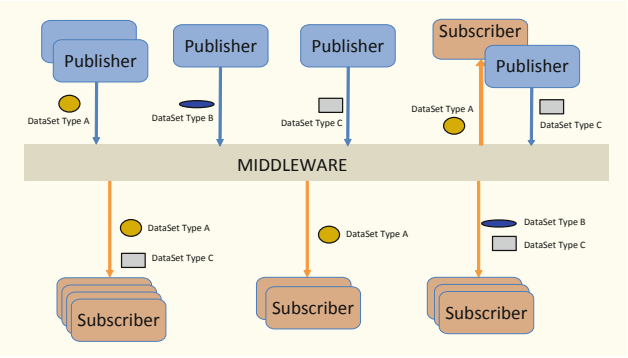

Fig. 3. PubSub structure [11,12]

In the subscription service, each Client with the same subscription elements creates a separate copy of the elements (MonitoredItems) on the server and starts a new cyclic data transmission. In the PubSub service, the server performs all of the activities simultaneously. It has one copy of the elements, which is sent to the Middleware. TSN is an extension in which a real-time protocol is used for data transmissions. This enables the security and guarantees the data transfer. There are different applications and solutions for OPC UA and this creates a problem with calculating and forecasting if the specific system using an OPC UA server and client will work correctly.

3 Data Flow in OPC UA

OPC UA comes with a ready-made model of addressing variables and dependencies, which is more than a communication system. The server initially prepares an information model that includes all of the knowledge about a specified installation, device or process. This information will be is then available via services (discover, read, write, subscribe, event, call method). In practical applications, it is necessary to have real-time properties on the entire system. The OPC UA standard only partially covers the production process. The process of data collection by the server is very important. In the first idea of OPC UA, the client

sends the requirements to the server. After this, the server should receive the data from sensors, Programmable Logic Controllers (PLC) or similar devices. In this situation, a special network is required for OPC UA transmission, which can double the costs. An alternative is that the server does not interfere with the lower level network, but only retrieves all of the data from it. Although this is safer for the lower network, it requires much more processing-power from the server because then the server should receive, analyse and eventually filter all of the incoming data packages.

The OPC UA system has some limitations and problems with OPC UA services that are mainly caused by processing delays, which change when the system parameters are frequently used. In the next part of the article, the authors will present some elements that affect server performance. In general, they can be divided into three parts:

1. processing incoming data (from the source to the server)
2. processing address model (inside the server)
3. processing outgoing communication (from the server to the clients) [12]

All of the elements above were tested in [13]. Now, it is necessary to determine whether it is possible to improve OPC UA.

4 Experiment Description

Performance tests were carried out to test the communication between these elements: Datasource, Server and Client. The first part of the tests consisted of ten Personal Computers (PC), which were connected by two independent networks (cable, 1 Gb/s, with switch).

All of the PC were divided: four of them produced the test data, two PC with OPC UA servers and four PC acting as clients. One of the servers (Server1) was set on an i7-8700K with six cores (12 threads)@3.7 GHz, 64 GB RAM, 2×1 Gb/s network cards with Windows 7. The second (Server2) was set on an i5-2320 with two cores (four threads)@2.5 GHz, 8 GB RAM, 2×1 Gb/s network cards also on Windows 7. The Datasources sent data every 50ms over Ethernet (100 values) to both servers. The servers retrieve data, filter and process it and then use it to update the address model. The clients on the other hand, subscribe all the available data (Fig. 4).

The second stand consisted of two Siemens PLCs with an embedded OPC UA server, 1 Gb/s network and up to 20 clients. The clients were based on JAVA or Python, respectively, and all of them subscribed to four analogue values each 250 ms (Fig. 5).

Only subscription was used in the test. Both the process data and internal parameters were collected by the server and then observed on the clients as Historical data after the tests. These parameters were network processing time, filtering, updating, memory usage, CPU usage, network usage, the amount of data for each client and the amount of MonitoredItems.

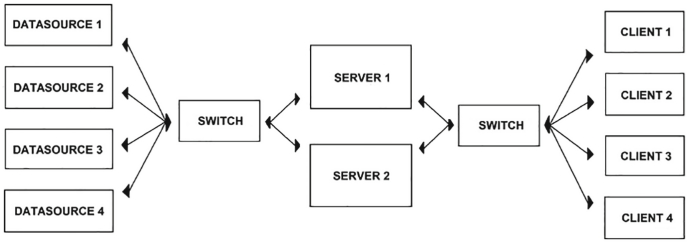

Fig. 4. Structure of the system with the PC

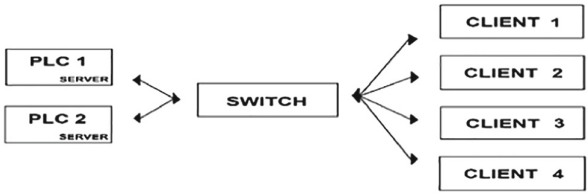

Fig. 5. Structure of the system with the PLS's

5 Experiment

The aim of the experiment was to find some solutions that can improve the effectivity of a server's performance. The experiments were conducted for applications written in different programming languages (Java, Python) and for operating systems (Windows, Siemens PLC). The idea was to determine how to detect any limitation of server processing and to use a secondary server if a limit was detected.

5.1 OPC UA Servers on Two PC

The servers (Server1 and Server2) collected the same data from the underlying network. The CPU load on both servers was almost identical (in shape), and the only difference was in the level. The "better" PC used only 10–25%, while the "worse" PC used 10–90%. Regularly (Fig. 6), the clients subscribed to the same data on both servers. It was visible that both servers had a larger CPU load.

5.2 Connecting Five Clients to OPC UA Server on the PLC

In the tests, there were five clients, each of which subscribed to four values from the server on the PLC. The server tried to process all of the requests but had a delay. For every subscription, the timestamps from the server and from the client was saved. All of the times that were measured in the Java program are presented in Fig. 7. It is visible that the clients received the data in the defined periods. A client requested a subscription every 250 ms. The server sent the subscriptions in

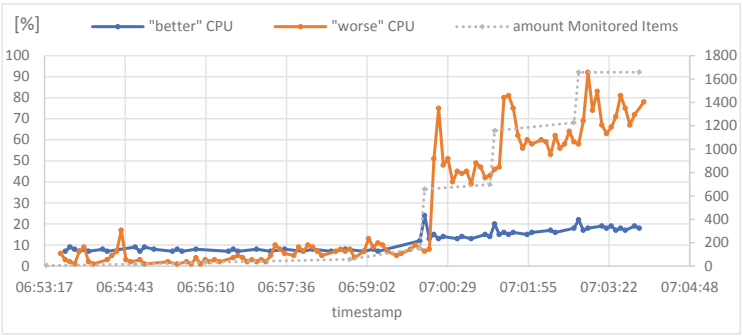

Fig. 6. CPU load on both servers

periods that were multiples of the subscription period. Simultaneously, the server tried to process the incoming data and MonitoredItems as quickly as possible. Nevertheless, the server was not able to keep the requested time.

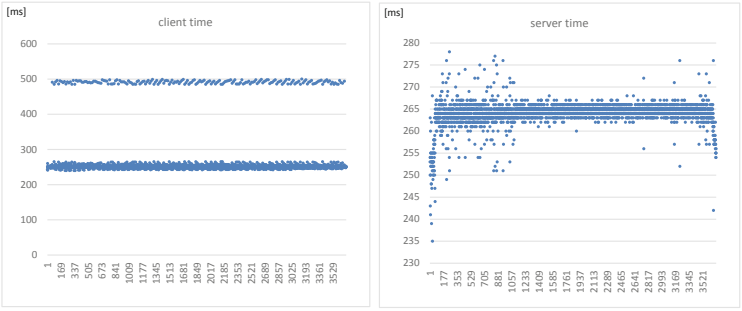

Fig. 7. Time difference for the incoming packages for a client and server on PLS – clients in the JAVA applications

The next figure (Fig. 8) presents the same results (20 clients) that were performed by the Python software.

It is quite visible that different applications had different results. The measurements from Python are more coherent and concentrated/both for server and client. The software on OPC UA Server was identical in both situations. It was not possible to calculate time parameter before the tests.

5.3 Twenty Clients with a Subscription to One OPC UA Server

It is visible that server cannot send data properly. Instead of 250 ms, the data are transmitted each 250, 500 and 750 ms. Clients have similar pattern. It was not possible to find those patterns before the tests (Fig. 9).

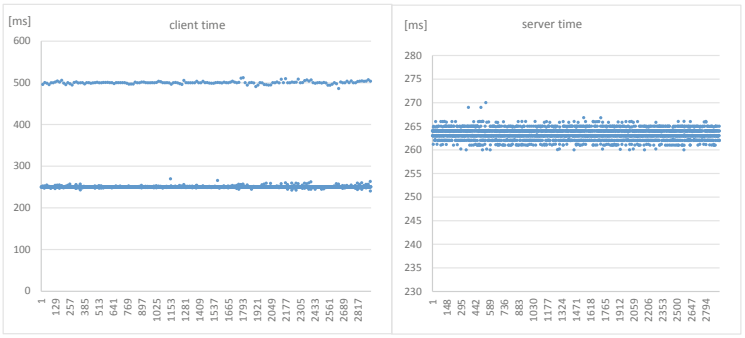

Fig. 8. Time difference for incoming packages for a client and server on PLS – clients in the Python applications

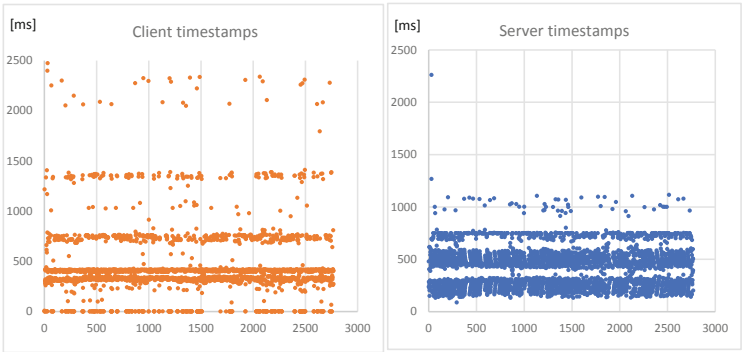

Fig. 9. Time difference between the timestamps for the server and client for 20 clients

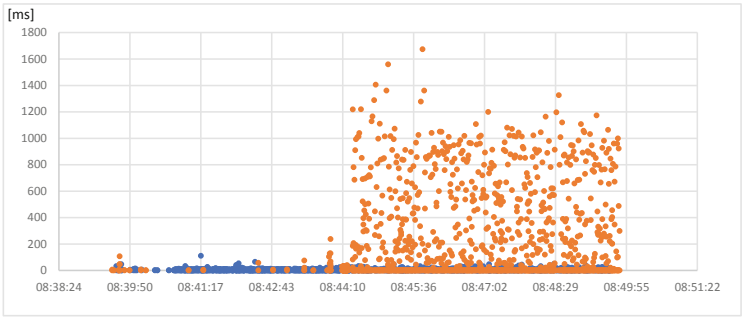

Fig. 10. Different loads on two servers

5.4 Two PLC with OPC Servers – Case of Different Loads

One of the ways to have a system work correctly is to use many servers. The results from the experiment – the time differences between the requested and

obtained timestamp are visible in Fig. 10. In the beginning, there were two clients with four subscribed variables. Both servers worked correctly – the time differences were near zero. At some point, 48 more clients were connected to one of the servers, which caused an increased tine difference of up to 1000 ms on that server. All of its clients did not receive the correct data. The second server worked correctly all of the time. It was possible to monitor the server traffic and in the event of delays, to block the server to another clients.

6 Conclusion

It seems impossible to find the one parameter that can represent the load of a server. It is possible to find many parameters and their relationships. Nevertheless, a parameter set can only be used when the source code of an application is available. It is almost impossible to monitor these parameters on a PLC. The other solution looks more promising. However, it is necessary to have an additional device that can monitor the servers. If a delay is detected, the server should be blocked for any new subscriptions. All of the clients should first ask this device about the available server. Then, if the client encounters a problem, the subscription should transfer to the next server. With some of the available servers, the system can process requests (subscriptions) from many more clients than a single server.

References

1. ISA95. https://opcfoundation.org/markets-collaboration/isa-95/
2. Industrie 4.0 Communication Guideline. https://industrie40.vdma.org/documents/4214230/20743172/Leitfaden_OPC_UA_Englisch_1506415735965.pdf/a2181ec7-a325-44c0-99d2-7332480de281
3. Zezulka, F., Marcon, P., Bradac, Z., Arm, J., Benesl, T., Vesely, I.: Communication systems for industry 4.0 and the IIoT. IFAC-PapersOnLine **51**(6), 150–155 (2018)
4. Haskamp, H., Meyer, M., Möllmann, R., Orth, F., Colombo, A.W.: Benchmarking of existing OPC UA implementations for industrie 4.0-compliant digitalization solutions. In: 2017 IEEE 15th International Conference on Industrial Informatics (INDIN), pp. 589–594. IEEE (2017)
5. Industrie 4.0. https://www.plattform-i40.de/PI40/Redaktion/EN/Downloads/Publikation/criteria-industrie-40-products.pdf?__blob=publicationFile&v=5
6. Profanter, S., Tekat, A., Dorofeev, K., Rickert, M., Knoll, A.: OPC UA versus ROS, DDS, and MQTT: performance evaluation of industry 4.0 protocols. In: Proceedings of the IEEE International Conference on Industrial Technology (ICIT) (2019)
7. Fojcik, M., Folkert, K.: Introduction to OPC UA performance. In: Kwiecień, A., Gaj, P., Stera, P. (eds.) CN 2012. CCIS, vol. 291, pp. 261–270. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-31217-5_28
8. Usländer, T., Epple, U.: Reference model of Industrie 4.0 service architectures. at-Automatisierungstechnik **63**(10), 858–866 (2015)
9. OPC Foundation. https://opcfoundation.org/about/opc-technologies/opc-ua/opcua-roadmap/

10. Bruckner, D., et al.: OPC UA TSN-A new solution for industrial communication. In: B&R Industrial Automation, Schneider Electric, ABB Automation Products, TTTech Computertechnik, General Electric Company, Huawei Technologies, Fraunhofer IOSB-INA, Phoenix Contact Electronics, Intel Corporation, Bosch Rexroth, Cisco Systems, Hirschmann Automation and Control, Moxa, Kalycito Infotech (2018)
11. OPC Foundation. https://opcfoundation.org/developer-tools/specifications-unified-architecture
12. OPC Foundation. https://opcfoundation.org/developer-tools/specifications-unified-architecture/part-14-pubsub/
13. Fojcik, M., Cupek, R., Ziebinski, A., Sande, O., Fojcik, M.K.: Quality of service in real-time OPC UA applications. In: Nguyen, N.T., Chbeir, R., Exposito, E., Aniorte, P., Trawiński, B. (eds.) ICCCI 2019. LNAI, vol. 11684, pp. 239–248. Springer, Heidelberger (2019)