Annales Univ. Sci. Budapest., Sect. Comp. 55 (2023) 175–190

AN ANALYSIS OF OPC UA COMMUNICATION WITH CNC TURNING MACHINES: EMPHASIS ON SYNCHRONIZED DATA ACQUISITION

Yazan Alomari and Mátyás Andó

(Budapest, Hungary)

Communicated by Attila Kiss

(Received May 17, 2023; accepted September 5, 2023)

Abstract. This paper delves into the operational efficacy of the OPC UA communication protocol in tandem with CNC machines. An OPC UA client, embedded within Node-Red, was devised to establish seamless communication with a CNC turning machine. A primary facet of our investigation centered on the protocol's performance during simultaneous multi-client interactions, with each client sending multiple queries—a scenario termed as "multi-client overlapping". Our analysis underscores the intricacies of the prioritization criteria employed to streamline queries. Findings indicate that queries from a single client aren't processed consecutively, leading to unsynchronized value retrievals. Comparative experiments, conducted directly between clients and server and via an intermediary switch, highlighted the marked advantages of the latter approach. Leveraging a switch bolstered the prioritization of queries, enhancing reliability. In essence, this research underscores the pivotal role of synchronized data acquisition in optimizing communication efficacy.

1. Introduction

The dawn of automation technology has brought with it an exponential rise in sensors, cameras and robotic devices, fueling an ever-increasing need

Key words and phrases: OPC UA, CNC, Industrial Communication Protocol, MES System.
2010 Mathematics Subject Classification: 68M10.

https://doi.org/10.71352/ac.55.175

to adeptly monitor and manage these resources. This essential task is typically achieved by reading and analyzing data gathered from machines or the peripherals connected to them. The Open Platform Communications United Architecture (OPC UA) emerges as a pivotal solution in this regard. Introduced by the OPC foundation, OPC UA represents a service-oriented approach, divorced from platform and technological limitations [12]. Standing out as an ideal communication protocol for effectuating a Service Oriented Architecture (SOA) at the shop-floor level, OPC UA is inclusive of eventing mechanisms, gaining rapid traction in contemporary automation systems [4]. This evolution towards harnessing advanced information technologies for streamlining production and product flow has birthed the concept of intelligent or smart manufacturing. Rooted in scientific and technological advancements, this novel manufacturing paradigm promises tangible enhancements throughout the product lifecycle, notably in design, management and actual production phases [15]. Supervisory Control and Data Acquisition (SCADA), defined as a control and monitor system architecture, primarily serves the purpose of overseeing machines, sensors, programmable logic controllers (PLC), and the overarching processes. Central to SCADA is a synergistic integration of a graphical user interface (GUI), an efficient data communication network, and sophisticated computing devices. Within the realm of metal component production, technologies underpinned by computer numerical control (CNC) — encompassing turning, milling, or electric discharge machining — are dominant. Monitoring these machines frequently entails the adoption of direct data acquisition solutions, especially prevalent in small and medium enterprises (SME). However, a common challenge remains: the absence of a universal application and a foundational systematic blueprint guiding their tangible implementation. Venturing into this complex tapestry, our study establishes an OPC UA communication framework with a CNC turning machine tailored for data collection and monitoring. A pivotal aspect of our exploration sheds light on the potential overlapping issue stemming from simultaneous multi-client connections to the OPC UA server. Our analysis extends to elucidate how the OPC UA server adeptly manages such multi-client overlaps. This introduction is a precursor to our findings that strive to blend technical expertise with practical implications, driving further advancements in the realm of automation communication.

2. OPC UA protocol and performance

OPC UA defines a reliable, secure, and interoperable data exchange mechanism between different industrial automation systems and devices like manufacturing execution system (MES), SCADA, and PLCs, human-machine interface (HMI) [10]. Figure 1 (a) shows the OPC UA Client architecture where the

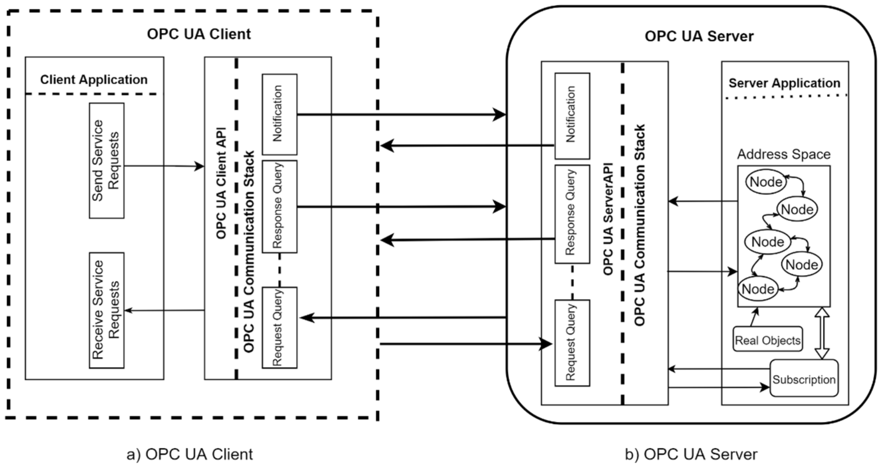


Figure 1. OPC UA a) Client architecture b) Server Architecture

application program interface (API), which is an embedded software development kit used by the clients to invoke the services implemented in a server [7]. The client's API calls will be converted into messages by the OPC UA communication stack, which implement the TCP/IP communication protocol and send the main communication entity to the server [3]. The OPC UA server architecture is shown in Figure 1 (b). The Server Application is the application that implements the Server's function [7]. Real objects such as physical devices are the objects that the OPC UA Server may access and manage. Additionally, to represent real objects, their definitions, and their references, the OPC UA Server employs particular objects called Nodes; the collection of Nodes is referred to as the AddressSpace [3].

OPC UA is a Service-oriented architecture which can ensure interoperability [7], which means that the generated data by a specific system or device is represented by the address space concept, so other systems would be able to interpret and use the generated data using the same protocol. Therefore, all data and information are represented in a regulated way to be accessible by client systems. Moreover, the system data and behavior are represented by OPC UA using the concept of an object. Variables, events, and methods can be stored in objects and linked to each other by references. Designers can fulfil their own application needs by using the standard information model, which allows type definition. Additionally, information models can be mixed for specialized domains like CNC systems. Companion Specifications (CS) are specific models that build on the basic model by inheriting its characteristics while also allowing for alterations that can be merged based on the system's requirements [9]. Data exchange in OPC UA can take place in different mechanisms, Read and Write are some of the most important features, along with the attributes of nodes to access metadata in the address space [8]. Subscription is another way of reading data, where it reads the data when it is changing. Subscription is the favored process for the clients who need updates periodically of variable value changes. The subscription offers different types of information to the client from the OPC UA server; for instance, it can be used to group a source of information all together, to manage a source of information, a Monitored Item is used [8]. Node-Red is a programming environment built on Node.js and uses graphical flows and nodes that make it easy to create event-driven applications simply and intuitively. It provides a browser-based editor to concatenate hardware devices, APIs and online services altogether. Node-RED, which is deployed on a central IoT server as an IoT middleware tool, represents one OPC UA client that gathers data from the OPC UA server for further processing and/or transformation. In the context of the IoT, Node-RED acts as a simple programming tool for connecting various types of software and hardware interfaces in a web browser-based editor. A wide range of pre-programmed nodes are available for the development of IoT flows in Node-RED. Also, JavaScript functions can be developed directly in the middleware using a text editor. Node-Red has been used in scenarios spanning a wide range of industries including manufacturing, healthcare agriculture, smart homes, and industrial automation. The lightweight runtime allows it to run on devices such as the Raspberry Pi or in cloud environments making it a very versatile tool [5].

3. Research activity and related work

The OPC UA standard has been used to monitor and control CNC machines over a wide range of parameters, such as in [11], which introduced a framework based on OPC UA for the modelling of milling and lathe CNC machine-tool to provide a different view of machine shops proceeding the 4.0 concept of the machine shop. Additionally, they have developed a device for data acquisition to promote old machine-tools in the digital age allowing the integration of the machine-tools beyond connectivity capabilities to a holistic framework. They validated their work on a case study presented in their laboratory. Additionally, in [6], they proposed a platform for Cyber-Physical Machine-Tools (CPMT) based on OPC UA and MTConnect, that provides standardized, interoperable, and efficient data communication between machine tools and various software applications. Several applications were developed to illustrate the advantages of the introduced CPMT platform, which includes the OPC UA client, the AR (Augmented Reality) supported portable human-machine interface, and the conceptual framework for the CPMT-based cloud manufacturing environment. In [14], the author presented an open numerical control system framework, followed by a data acquisition technique based on the OPC standard. They further claimed that using time sequence similarity analysis and cluster analysis, the state model could be predicted. The prediction result may then be used to provide fault warning and diagnostics for the CNC machine-tool. However, in their experiments they have utilized only Siemen's controller machines. Furthermore, they have not stated whether or not there might be problems in the time sequence if several clients were used. Further analysis and evaluation for the OPC UA round-trip timings have been conducted in [3] considering a variety of message sizes. They also supplied measures of the resultant delay for subscriptions in the event of short update intervals, as well as network bandwidth utilization. Moreover, the cost of round-trip delays because of using an encrypted connection have been evaluated too. Nevertheless, this article has not studied the impact of increasing the number of client sessions. In [2], they have conducted several experiments with OPC UA client/server and pub/sub communication. Additionally, they have developed a linear regression model for predicting CPU consumption, which may be used to size hardware setups. On a Raspberry Pi Zero, the throughput of up to 40,000 signals per second is possible due to the server CPU being the main bottleneck. They have also shown that if a server is used by more than 20 clients, the overhead of managing client/server sessions can have a significant influence on performance. However, they have not examined the multi-client overlap. In [1], They have measured the time synchronization between nodes by conducting a test bed of 50 devices. In laboratory settings, the time synchronization latency was about 100 microseconds, with a 50 ns jitter. Additionally, they highlighted the theoretical calculation of the shortest attainable cycle times with 100 MBit and 1 GBit switches. They anticipated a cycle duration of about 100 microseconds for up to 100 nodes utilizing a 1 GBit TSN switch with 100 bytes of payload. Mainly the paper has focused on TSN communication. However, some critical topics have not been addressed, such as the OPC UA communication mechanisms (client/server and pub/sub), multi-client communication and overlap.

In this article, a study in realizing the full potential of Industry 4.0 was introduced, our research delineates a pioneering method to seamlessly integrate and digitalize machine shop operations. Utilizing the OPC UA communication protocol, complemented by the versatility of Node-Red as its client, we've forged a system to addresses the complexities of simultaneous multi-client interactions. Our deep dive into data acquisition, especially in the realm of CNC lathe machines, highlights both the challenges faced and the solutions devised. The insights garnered here, though centered on a specific machine, hold implications for a broader industrial framework, paving the way for future innovations in the domain of synchronized, real-time manufacturing communication.

4. Methodology

After ensuring that the firmware of the Numeric Control Unit was up-to-date and the OPC UA communication was supported, the node-red-contrib-opcua library was installed in the Node-Red framework on the client side. The node OPC UA Client was then configured using the TCP/IP address of the OPC UA server. Parameters and variables in the CNC lathe machines were assigned with a NodeID in the address space. The values of these nodes were accessed by assigning the NodeID along with the DataType of the node in a template box.

4.1. Network configuration

The OPC UA connection to the server was established through a regular Ethernet switch, which was used to bridge communication between the client machine and the Siemens CNC turning machine. The switch was used in its default configuration without any specialized settings, ensuring a straightforward yet efficient network setup for our experiments.

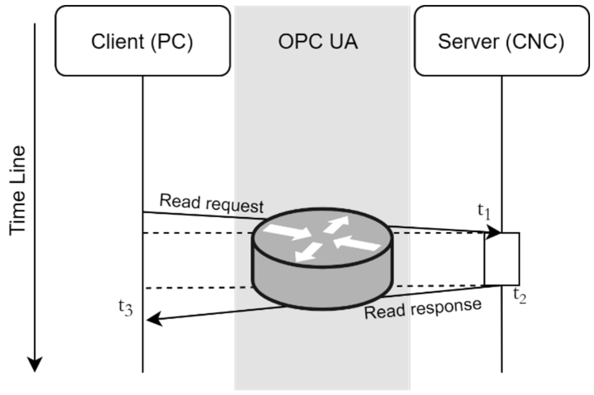


Figure 2. OPC UA "Read" and "Update" Sequence Diagram

Figure 2 illustrates the sequence diagram for OPC UA read function and the response of the request. The OPC UA connection is established through a switch device. The proposed mechanism in our data acquisition system is as follow. Initially, the 'Read' function accessed the values. The desired parameters were listed line by line, accompanied by the respective data type. These parameters were then formatted into CSV for ease of processing. Following this, a JavaScript for loop function forwarded the desired NodeIDs, along with their data type, to the OPC UA Client that maintained communication with the server. A subsequent function retrieved the value, timestamped it, and stored it in the database. The experiments were carried out using a Siemens CNC turning machine with Sinumerik (828D v4.7). We utilized Node-Red 2.0 with Node.js 17.0.0 and the node-red-contrib-opcua library version 0.2.244. The client machine ran on the Windows 10 operating system and accessed the framework through the Google-Chrome version 96 web browser. To ensure the reliability and robustness of our findings, our experimental setup mimicked real-world conditions as closely as possible. This realistic environment, combined with the versatility of the tools and software we used, provides a clear indication of the efficacy of the proposed data acquisition system in actual industrial scenarios. This section offers a comprehensive analysis of our proposed model. The OPC UA server is adept at serving multiple clients concurrently. In our experiments, the CNC turning SIEMENS SINUMERIK 828D control was used, which, according to technical data in [13], can accommodate up to 5 clients.

5. Results and discussion

In this section, we delve into the intricate dynamics of the OPC UA server's responsiveness under varied client loads, specifically within the SIEMENS SINUMERIK 828D control system. The objective of this study was to comprehend the processing patterns of the server when multiple clients communicate concurrently and to uncover associated challenges.

5.1. Communication setup

For the purpose of this study, communication between the OPC UA clients and the server was facilitated through an intermediary ethernet switch. This is noteworthy as switches, by nature, can introduce latency. Yet, the consistency of our results underscores the robustness of the OPC UA protocol within the SIEMENS SINUMERIK 828D control, even in such configurations.

5.2. Multi-client overlap and priority

When multiple clients attempt to communicate with the OPC UA server simultaneously, a phenomenon termed "multi-client overlap" arises. This overlap invariably results in an extended time-to-completion for individual requests. Though queries from various clients might be dispatched synchronously, the time of their arrival at the server can differ, invariably favoring some over others. As illustrated in Figure 3 (a), this mechanism ensures clients joining the queue later, particularly the last two, experience a diminished priority over their antecedents. The client's query arrival time holds paramount importance in determining its position on the priority ladder.

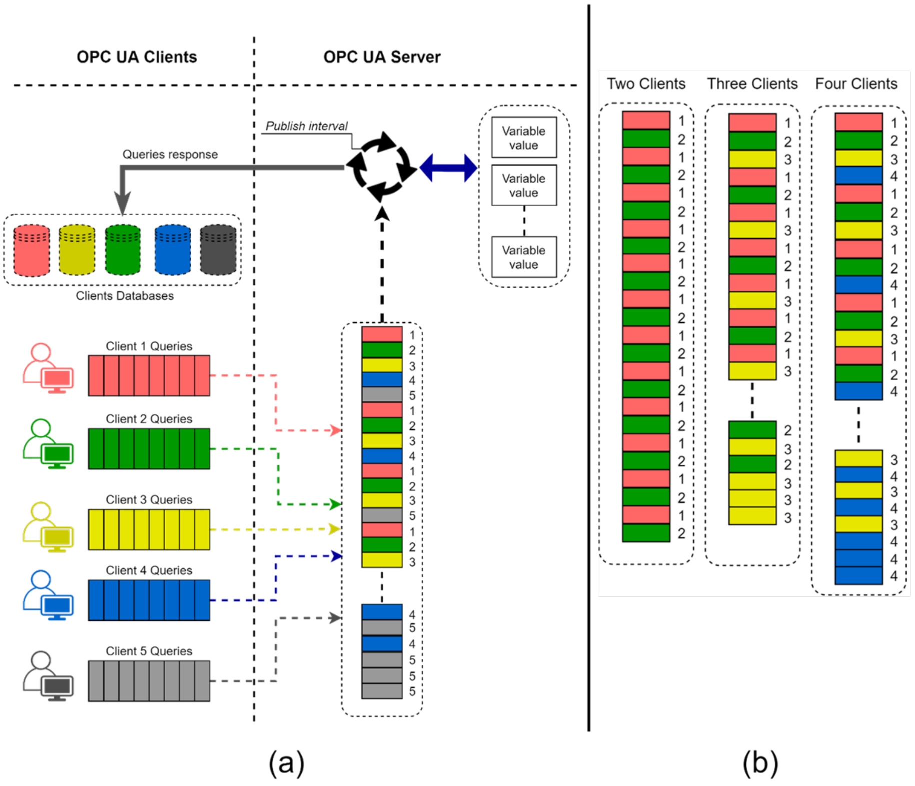


Figure 3. a) Multi-Client Communication System b) different cases of different number of clients

5.3. Serving multiple clients

The operational intricacies of the server, when serving a varying number of clients, is depicted in Figure 3 (b). For dual-client scenarios, the system serves requests impartially, leading to consistent time-to-completion rates — a testament to the reliability of these values. However, as the client count escalates beyond two, the inherent priority system becomes more pronounced, leading to divergent completion times and, arguably, less consistent results.

5.4. Discussion and analysis

The purpose of this experiment is to understand how the client's system, using OPC UA on CNC machines, handles simultaneous queries from multiple clients. The key metrics analyzed are:

$$\mathrm{APT} = \frac{\sum_{i=1}^{n} (\mathrm{ResponseTime}_i - \ \mathrm{QueryTime}_i)}{n},$$
$$\mathrm{ATT} = \frac{\sum_{i=1}^{n} (\mathrm{ArrivalTime}_i - \ \mathrm{ResponseTime}_i)}{n},$$
$$\mathrm{ATRT} = \frac{\sum_{i=1}^{n} (\mathrm{ArrivalTime}_i - \ \mathrm{QueryTime}_i)}{n},$$
$$\mathrm{SDPT} = \sqrt{\frac{\sum_{i=1}^{n} (\mathrm{APT}_i - \ \mathrm{APT}_{avg})^2}{n}},$$
$$\mathrm{SDTT} = \sqrt{\frac{\sum_{i=1}^{n} (\mathrm{ATT}_i - \ \mathrm{ATT}_{avg})^2}{n}},$$
$$\mathrm{SDTRT} = \sqrt{\frac{\sum_{i=1}^{n} (\mathrm{ATRT}_i - \ \mathrm{ATRT}_{avg})^2}{n}},$$

where

APT	–	Average Processing Time,
ATT	–	Average Transmission Time,
ATRT	–	Average Total Response Time,
SDPT	–	Standard Deviation for Processing Time,
SDTT	–	Standard Deviation for Transmission Time,
SDTRT	–	Standard Deviation for Total Response Time.

Client Metrics	APT (ms)	ATT (ms)	ATRT (ms)
Client 1	113	62	176
Client 2	121	67	189
Client 3	134	63	197
Client 4	**251**	63	315
Client 5	**283**	64	347

Client Metrics	SDPT (ms)	SDTT (ms)	SDTRT (ms)
Client 1	108	4.8	105
Client 2	108	13	114
Client 3	116	5	116
Client 4	**159**	3.7	158
Client 5	**155**	3.3	155

Table 1. Metrics for 5 clients sending 10 queries simultaneously

This experiment (Table 1) offers a lucid understanding of the interplay between the OPC UA server's processing mechanism and its behavior under different client loads. Such insights are invaluable for industries harnessing the SIEMENS SINUMERIK 828D control, providing them with a blueprint to optimize their communication strategies.

Starting with APT, it's noticeable that Client 4 and Client 5 experience significantly higher processing times, with APT values of 251 ms and 283 ms, respectively. This can be attributed to the priority mechanism, where these clients are likely given lower priority, thus taking longer to process their queries.

ATT values are more consistent across the board, ranging between 62 ms to 67 ms. This suggests that the server's transmission time is relatively stable, irrespective of the client's priority. It also signifies that the network between the client and server is not a significant bottleneck in this setup.

The ATRT values are the most critical for real-time applications. These numbers highlight that clients with lower processing times (Client 1, Client 2, and Client 3) also have more favorable total response times, corroborating the significance of the server's processing mechanism in overall performance.

The standard deviation values (SDPT, SDTT, SDTRT) further elucidate the system's performance. For instance, Client 4 and Client 5 have high SDPT values (159 ms and 155 ms), indicating greater variability in processing times, which could be problematic for applications requiring high consistency.

5.4.1. Variability in processing times

The processing time represents the time taken by the server to process the queries of different clients. The significant variation in the standard deviation of processing time across clients suggests that the server's processing time isn't consistent across multiple simultaneous requests. This inconsistency can be attributed to the server's mechanism of prioritizing certain requests over others as illustrated in Figure 3.

5.4.2. Transmission time stability

The transmission time represents the time taken for the response to be transmitted back to the client after processing. A key observation is the relatively low standard deviation of transmission times across all clients. This indicates a stable network connection and efficient server-client communication. Even in scenarios with multiple clients, the data transmission seems consistent, which is a positive sign for real-time applications.

5.4.3. Divergence in total response time

Total response time (processing + transmission) showcases a higher standard deviation, especially as we increase the number of clients. This divergence can be directly attributed to the variability in processing times, indicating that the processing phase is the primary contributor to the inconsistency in response times.

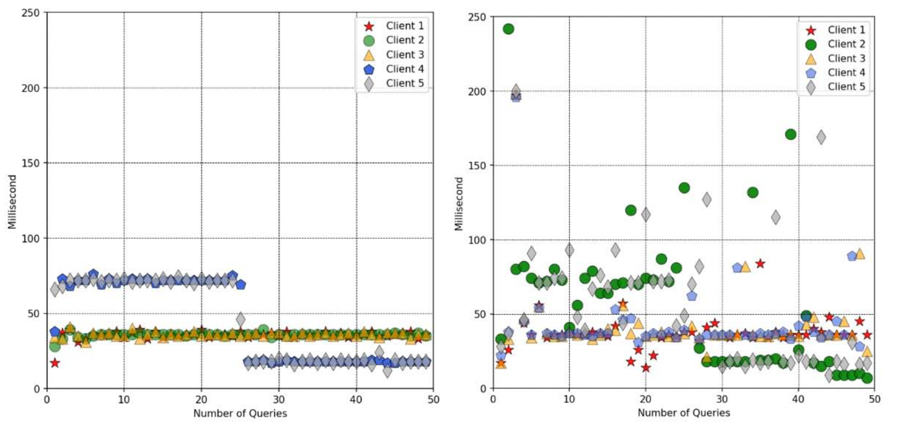
5.4.4. Server prioritization mechanism

As gleaned from the prior information from Table 1, the server utilizes a unique mechanism for serving up to 5 clients. This mechanism prioritizes the last and second last clients to join the queue, which inevitably leads to variability in processing times. Such a mechanism can be both beneficial and detrimental:

- Beneficial because it ensures that newer clients (who might be sending urgent requests) are prioritized.
- Detrimental as it introduces inconsistency in response times, making it challenging for applications that rely on predictable server responses.

5.4.5. Network topology impact on OPC UA communication performance

In a separate experiment, we bypassed the switch device and established a direct connection between the client and the server. The results were strikingly variable, with the time to completion ranging unpredictably from 3 to 250 milliseconds, as depicted in Figure 4 (b). This significant fluctuation underscores the unreliability and inaccuracy of the acquired values in such a configuration.



(a) Via Switch device *(b) Direct connection to NCU (No Switch device)*

Figure 4. The order of serving 50 queries for 5 clients

The presence of a switch device plays a crucial role in orchestrating and stabilizing the communication flow, especially when multiple clients attempt to communicate concurrently. Figure 4 contrasts the outcomes of two distinct experiments. One with a direct connection to the numeric processing unit (NCU) and the other through a switch device. It reveals the stark difference in the completion time of the queries and highlights the increased likelihood of outliers when directly connecting multiple OPC UA clients to the server. Of particular note is the prioritization mechanism for serving clients. Specifically, the last two clients (n and n-1) receive only half the priority of the preceding clients. When examining Figure 4 (a), it's evident that the first three clients experience consistent and nearly identical completion times, showcasing the stability provided by the switch. In stark contrast, Figure 4 (b) displays results from the direct connection setup. Without the mediating influence of a switch device and its inherent prioritization, the communication becomes erratic. This is evidenced by the considerable number of queries taking an inordinately long time — hundreds of milliseconds in some cases — to process.

6. Case scenario: The need for millisecond precision in CNC lathe operations

6.1. Background

Considering a high-end manufacturing facility that specializes in producing precision components for aerospace applications. Given the critical nature of these components, even a minuscule error can lead to catastrophic failures in the field. The facility uses state-of-the-art CNC lathe machines, which can operate at a spindle speed of up to 10,000 rpm. At this speed, the spindle covers approximately 0.1666 revolutions every millisecond.

6.2. Real-time monitoring imperatives

Modern PLCs often operate with cycle times in the nanosecond range to manage real-time control tasks effectively. However, when an external software system, like an MES, is responsible for monitoring quality or making production-related decisions, the requirements for communication speed and data synchronicity become increasingly critical. This is particularly true for applications involving high-speed spindle operations, such as in CNC machining. A delay in data acquisition or a lack of synchronization, even on the order of milliseconds, can lead to significant errors. For example, at a spindle speed of 10,000 rpm, a 10-millisecond delay in data acquisition would mean the spindle has already made approximately 1.666 revolutions, potentially leading to defects in the component being machined. Therefore, until external software can match the internal computational speed of the PLC, it is imperative to consider the PLC's CPU capabilities to ensure that production quality is maintained.

6.3. Critical factors in real-time data retrieval

Latency: Given the high rpm, the latency in data retrieval and communication becomes a significant factor. Even a delay as small as a few milliseconds can lead to substantial errors. Network Congestion: If multiple clients or systems are querying the CNC machine simultaneously, it can lead to network congestion, increasing the response time. This can be especially problematic during peak operational hours.

Server Load: The server's processing time can vary based on its current load. If the server is processing multiple queries, it might take longer to respond to a particular request, leading to outdated or incorrect data being retrieved.

Prioritization of Queries: As we observed in our experiments, the order of client requests can impact the response time. Critical operations, such as monitoring spindle speed, should be given higher priority to ensure real-time data retrieval. Buffering and Data Loss: If the communication protocol involves buffering data before transmission, there's a risk of data loss or delay, especially if the buffer gets filled faster than it's being read.

Feedback Loop: A real-time feedback loop should be established, wherein the MES system can instantly adjust machine parameters based on the data retrieved. This can help in correcting any deviations in real-time.

For industries where precision is paramount, such as aerospace component manufacturing, the importance of real-time data retrieval cannot be overstated. Given the high speeds at which CNC lathe machines operate, even a slight delay in communication can lead to significant errors. It's crucial for the designers and users of the MES system to understand these challenges and implement strategies to mitigate them. This might involve network optimizations, server upgrades, prioritizing critical queries, or even implementing real-time feedback loops.

7. Conclusion and significance

With the increasing complexity of modern industrial systems and the need for real-time data synchronization, the challenge of acquiring simultaneous values from multiple nodes at a specific time becomes paramount. Such precise synchronizations, like reading the concurrent position values of a 3-axis system (X, Y, and Z) in tandem with the actual feedrate and rotational speed, are critical in ensuring the accuracy of the manufacturing process. Furthermore, as industries tailor their operational technologies, including MES, ERP, and SCADA systems, the possibility of multiple databases recording varied values becomes a significant concern.

This research has introduced a novel preliminary study on the data acquisition from CNC machines using the OPC UA communication protocol combined with Node-Red. It has delved into the intricacies of the OPC UA client and server architectures, shedding light on the process of data acquisition in-depth. A significant discovery made in this study was the ”Multi-Client Overlap” issue. When multiple clients communicate with the server simultaneously, the inherent prioritization mechanism in OPC UA can result in asynchronous values.

Through systematic experimentation, the research elucidated the handling and prioritization of queries, especially when clients’ requests overlap. The intricate dynamics of how priority allocation functioned across different scenarios were presented, offering valuable insights into the inner workings of the OPC UA protocol.

A pivotal aspect of this study was the performance evaluation of the proposed model. Utilizing metrics such as Average Processing Time (APT), Average Transmission Time (ATT), and Average Total Response Time (ATRT), alongside their standard deviations, the study meticulously assessed the system’s performance across various scenarios. Moreover, the research offered a comparative view of direct connection to the NCU versus a connection via a switch device in multi-client scenarios, emphasizing the pivotal role of intermediaries in ensuring stable and synchronized communication. In light of the new advancements discussed, future endeavors should focus on optimizing the prioritization mechanisms, integrating AI-driven analytics for enhanced performance, and further evaluating the scalability of the proposed model in even more complex industrial setups. The findings of this research serve as a foundational step towards achieving perfect synchronization in data acquisition, crucial for the precision and efficiency demanded by modern industries.

References

[1] **Bruckner, D., R. Blair, M-P. Stanica, A. Ademaj, W. Skeffington, D. Kutscher, et al.,** OPC UA TSN A new Solution for Industrial Communication, n.d.

[2] **Burger, A., H. Koziolek, J. Rückert, M. Platenius-Mohr, and G. Stomberg,** Bottleneck Identification and Performance Modeling of OPC UA Communication Models, 2019, 12–19. https://doi.org/10.1145/3297663

[3] **Cavalieri, S., and F. Chiacchio,** Analysis of OPC UA performances, *Computer Standards and Interfaces*, **36** (2013), 165–177. https://doi.org/10.1016/j.csi.2013.06.004

[4] **Garcia Izaguirre, M.J.A., A. Lobov, and J.L. Martinez Lastra,** OPC-UA and DPWS Interoperability for Factory Floor Monitoring using Complex Event Processing, 2011.

[5] **IEEE Staff,** 2018 17th International Symposium INFOTEH JAHORINA (INFOTEH), IEEE, 2018.

[6] **Liu, C., H. Vengayil, Y. Lu, and X. Xu,** A Cyber-Physical Machine Tools Platform using OPC UA and MTConnect, *Journal of Manufacturing Systems*, **51** (2019), 61–74. https://doi.org/10.1016/j.jmsy.2019.04.006

[7] **Mahnke, W., S.H. Leitner, and M. Damm,** *OPC Unified Architecture*, Springer Science & Business Media, 2009.

[8] **Mahnke, W., S.H. Leitner, and M. Damm,** *OPC Unified Architecture*, Springer Berlin Heidelberg, 2009. https://doi.org/10.1007/978-3-540-68899-0

[9] **Martins, A., J. Lucas, H. Costelha, and C. Neves,** Developing an OPC UA Server for CNC Machines, *Procedia Computer Science*, **180** (2021), 561–570. https://doi.org/10.1016/j.procs.2021.01.276

[10] **Martins, A., J. Lucas, H. Costelha, and C. Neves,** CNC Machines Integration in Smart Factories using OPC UA, *Journal of Industrial Information Integration*, 100482 (2023).

[11] **Mourtzis, D., N. Milas, and N. Athinaios,** Towards Machine Shop 4.0: A General Machine Model for CNC machine-tools through OPC-UA, *Procedia CIRP*, **78** (2018), 301–306. https://doi.org/10.1016/j.procir.2018.09.045

[12] **Schleipen, M., S.S. Gilani, T. Bischoff, and J. Pfrommer,** OPC UA & Industrie 4.0 - Enabling Technology with High Diversity and Variability, *Procedia CIRP*, **57** (2016), 315–320. https://doi.org/10.1016/j.procir.2016.11.055

[13] **Siemens AG, DF MC,** SINUMERIK, SINUMERIK 840Dsl/828D SINUMERIK Access MyMachine / OPC UA Configuration Manual Valid for: OPC UA server Version 2.1, 2018.

[14] **Wang, W., X. Zhang, Y. Li, and Y. Li,** Open CNC Machine Tool's State Data Acquisition and Application Based on OPC Specification, *Procedia CIRP*, **56** (2016), 384–388. https://doi.org/10.1016/j.procir.2016.10.061

[15] **Zhong, R.Y., X. Xu, E. Klotz, and S.T. Newman,** Intelligent Manufacturing in the Context of Industry 4.0: A Review, *Engineering*, **3** (2017), 616–630. https://doi.org/10.1016/J.ENG.2017.05.015

Y. Alomari and M. Andó
Eötvös Loránd University (ELTE)
Faculty of Informatics
Budapest
Hungary
`yazan@inf.elte.hu`
`am@inf.elte.hu`