machines

MDPI

Article

Digitising a Machine Tool for Smart Factories

Anton Averyanov *, Shohin Aheleroff, Jan Polzer and Xun Xu

Department of Mechanical and Mechatronics Engineering, The University of Auckland, Auckland 1010, New Zealand
* Correspondence: aave618@aucklanduni.ac.nz

Abstract: Smart factory development renders an incredible opportunity for the manufacturing industry to join the Fourth Industrial Revolution (Industry 4.0). However, an incredible number of conventional CNC machine tools are populating the world's factories. Replacing these machines is an expensive process. This task might be considered unliftable by most small businesses. An inexpensive digitalisation of Machine Tool 3.0 to an Industry 4.0-compatible tool might be one way for small businesses to keep up with the progress and stay competitive. The developed framework uses recent advances in the open-source community to transform a conventional CNC machine into Machine Tool 4.0. The suggested approach opens up a way to bypass the proprietary computer numerical control and enable connectivity and efficient data communication with the machine tool. At almost no cost, the provided strategy converts an average CNC machine into a machine tool with the full spectrum of accessibility and interoperability of Machine Tool 4.0. The proposed solution can enable small- and medium-sized enterprises to step up and propel them into the Industry 4.0 era.

Keywords: digitalization; CNC machines; Industry 4.0; retrofit; decoding; SME

Citation: Averyanov, A.; Aheleroff, S.; Polzer, J.; Xu, X. Digitising a Machine Tool for Smart Factories. *Machines* **2022**, *10*, 1093. https://doi.org/10.3390/machines10111093

Academic Editor: Gianni Campatelli

Received: 19 October 2022
Accepted: 16 November 2022
Published: 18 November 2022

1. Introduction

The Fourth Industrial Revolution (Industry 4.0) is leading to digitising machine tools, further flexibility, and collaboration among manufacturing resources, targeting mass personalisation as an emerging manufacturing paradigm [1]. Digital twin (DT) as part of a cyber-physical system (CPS) is one of the core concepts used for many areas of machine tool advancements [2–5]. A CPS combines models and methods from different engineering and science fields for improved efficiency, sustainability, and scalability of the system [6]. A generic system architecture for a cyber-physical machine tool (CPMT) is shown in Figure 1. Three principal components of this system are physical devices, networks, and machine tool cyber twin (MTCT). The CPMT system provides the following functions: data fusion, presentation of the machine tool (MT), optimisation of the performance, and management of big data [7]. Moreover, network communication between MT and DT is one of the most important components of CPS and is a challenging task in the realisation of CPMT [8]. As a crucial part of the digitalisation of an MT, network communication needs to be linked to the MT controller.

Since its introduction, numerical control (NC) and computer numerical control (CNC) come through a series of iterations. Starting from hard-wired NC controllers more than 70 years ago, they advanced into soft-wired CNC closed systems in the 1980s. In the 1990s, they changed to PC-based open architecture controllers (OAC) and are expected to become soft-NC STEP-NC-based with open hardware and software [9]. It is not uncommon to find a working CNC machine from the 1980s on the shop floor of small- and medium-sized enterprises (SMEs). However, machines from the 1990s and younger machines are more common in SMEs.

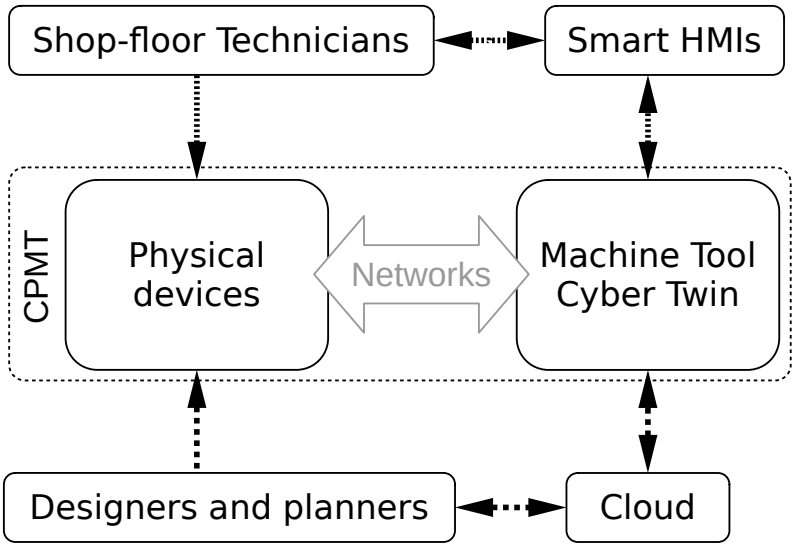

Figure 1. Generic system architecture for CPMT. Adopted from [7].

Functionally, the CNC system is formed by three units. The human–machine interface (HMI) unit provides the user interface and access to functions. In contrast, the numerical control kernel (NCK) unit controls servo motors and interprets the program. Lastly, the programmable logic control (PLC) unit is responsible for sequential control of everything else in the machine tool [9].

In contrast to the earlier closed CNC, the OAC system was built on PC architecture and utilised a real-time operating system (RTOS). Commonly, one of the following three types of architectures is employed for designing a PC–NC system:

1. One PC with HMI and NCK/PLC is connected to the PC via an extension slot.
2. Two PCs with high-speed communications; one with HMI and the other host in RTOS and NCK/PLC functionalities.
3. One PC with RTOS combines all three units: HMI, NCK, and PLC.

One of the industrial protocols, such as SERCOS, EtherCAT, PROFInt, etc., is utilised to ensure precise synchronisation between two PCs. Ethernet and RS-485 are the most common physical layer standards among them [10]. The proposed framework is intended to be applied for the digitalisation of an MT with the type 2 CNC architecture and was tested on a CNC system with Ethernet communication protocols.

The main objective of this research was to provide an affordable solution for converting a traditional CNC machine into a smart one, with data enrichment, accessibility, and interoperability toward Machine Tool 4.0. The rest of this paper is organised as follows. A research method is presented in Section 2. Section 3 describes an experimental implementation in a laboratory environment to validate the work. Section 4 discusses the results. The conclusions and future research directions are presented in Section 5.

2. Research Method

Accessibility of the network link between two components of the CNC system offers one possible solution for rendering spaces for existing MTs in smart factories. Front-end PCs or HMI hosting PCs constantly communicating with the back-end PC, the one hosting RTOS with NCK/PLC functionalities. By investigating the communication and reproducing the responses of the front-end PC, it is possible to bypass it and control the CNC system from an external source. For example, install additional sensors, such as dynamometer and current meters, and establish asset parameters, online monitoring, and optimisation. Many options are possible, e.g., connect the MT to a network as part of a service, lunch the remote augmented reality AR support, or access and control the MT from mobile devices.

This approach's realisation relies on the man-in-the-middle (MITM) technique. Since this type of CNC depends on an open communication protocol between units of the CNC system, the application of such a technique can reveal the necessary knowledge to interact with the NCK unit. To perform MITM, an additional PC with two network interface controllers (NIC) of the appropriate type is required and will be referenced as the MITM PC. Almost any computer can be used for this task. Unfortunately, two NICs are not common computer configurations. It is often possible to extend the number of network interfaces through an expansion slot or another interface, for example, by connecting a USB network card, the so-called USB-to-Ethernet adapter. Hence, the name, including the MITM PC in the existing communication channel of the CNC, should be sufficient for the physical setup, as shown in Figure 2.

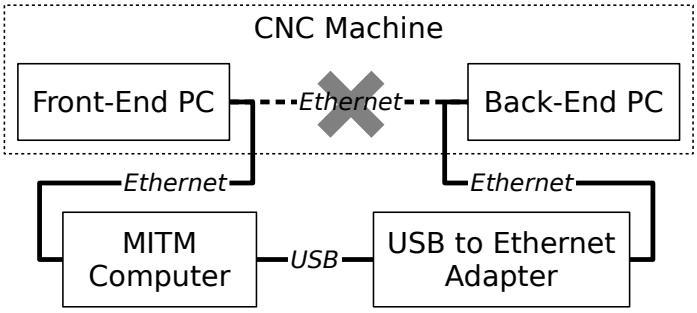

Figure 2. MITM physical connection arrangements.

The two NICs involved must be set into a network bridge to allow uninterrupted communications for the front-end and back-end PCs. To gather the network parameters, one can observe settings at the front-end PC, where a non-real-time operating system OS is installed. In most cases, the front-end PC is equipped with Windows OS [9]. Alternatively, port scanner software is necessary. Once the MT can operate as normal with the MITM PC in place, communication investigations can be conducted. A packet analyser software package is essential for this task. Network packets consist of control information and payload. Since control information is generated during the communication process, only the payload is considered.

The initialisation stage is a vital part of the CNC's internal communication. During this phase, the front-end PC and back-end PC establish the communication and exchange essential information, such as system parameters and the initial state of components. Attention must be paid to the maximum packet size since the manufacturer can artificially limit it, which can cause packet rejection. After initialisation is complete, the machine generally goes into standby mode, and only communication maintenance packets are transferred. Executed commands from the user interface or operator panel on the front-end PC are converted into the payload and sent to the back-end PC. Typically, the back-end PC sends a response message. However, the type of response differs depending on the command executed and the current operation mode. For example, during axial movements in program execution, NCK sends changing coordinates, creating a constant data stream. The captured sequences of packets can be analysed and processed for future use.

The list of executed and captured commands can be customised to current needs and can include common auxiliary functions, such as door and coolant operations. Furthermore, they can include axes movements and spindle control. Motor control functions require detailed samplings and approximating funding to discover the closest mathematical functions representing the control value [11]. Table 1 shows the sample data captured during executing "Door open" command. As can be seen, when the button is pressed, two packets with eight bytes of data each are sent from HMI to NCK. Then, NCK sends a signal to the pneumatic door actuator. Consequently, the door sensor is triggered, and 40 bytes of related data are conveyed to the back-end PC. Once the action is confirmed without an alarm, the NCK unit replies with the changed status of the MT representing the opened

door and restriction of some functions, such as rapid axis and spindle operation. The NCK unit reaction time measures in milliseconds, meaning the actual execution command will be sent before the operator lifts a finger from the button. Hence, the latter data observed in the sequence of the two packets represent the release of the "Door open" button.

Table 1. Hexadecimal representation of the door open button event.

Hexadecimal Representation	Bytes	Note
27 68 00 00 00 00 09 00	8	Button down—packet #1
0c 60 00 00 00 00 01 00	8	Button down—packet #2
10 00 20 00 00 00 00 00 00 00 00 00 00 01 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	40	Door sensor—off
07 00 00 00 01 00 00 00	8	Machine status changed packet #1
07 00 00 00 00 00 00 00	8	Machine status changed packet #2
27 68 00 00 00 00 09 80	8	Button up—packet #1
0c 60 00 00 00 00 01 80	8	Button up—packet #2

In order to send packets, a wide range of software tools can be considered. Common practice is to write a simple server or a driver in languages, such as Python or JavaScript. Typically a functional, basic server requires less than ten lines of code. Depending on the level of control required, reading or replying to messages coming from the back-end PC/NCK unit might not be required, simplifying the initial development. A PC directly connected to the back-end PC and able to send sequences of packets with respective payloads can be used to manipulate the MT. Replacing the original front-end PC with a custom one allows complete freedom in adding new functions and functionalities to the MT.

3. Implementation

The realisation of a CPMT requires a fusion of models and methods from different engineering fields and computer science [6].

EMCO Concept Turn 155 is a small CNC lathe designed for training purposes (Figure 3). For this reason, a GE FANUC Series 21 CNC controller is emulated on a PC; it could be SIEMENS or another one. Despite the fact that it is a training machine, it has all of the standard features of an industrial one: pneumatic chuck, eight-position tool turret, pneumatic sliding door, coolant pump, air blow function, and a full-sized corresponding FANUC control panel. The lathe was designed to be as close to the industrial version as possible; hence, it was chosen to conduct this research.

The machine's CNC system includes two personal computers. One is positioned right behind the front panel, as shown in Figure 4, and will be referenced as a front-end PC. It runs an MS Windows XP-embedded operating system and EMCO WinNC GE Series FANUC 21 TB software. The front-end PC is responsible for reading user input from the front panel and emulating a FANUC 21-screen interface. The second is a PC located in the electrical cabinet at the rear of the machine and will be referenced as the back-end PC, as shown in Figure 4. It runs a version of a real-time operating system called RTLinux and is responsible for machine logic behaviour. Based on the information received from the front-end PC and sensor readings, it manipulates the actuation devices of the machine accordingly. These two computers are bound into a network by an Ethernet cable and communicate over the transmission control protocol TCP. While the front-end PC utilises standard MS Windows OS tools to configure network access, the back-end PC has hard-coded network settings and is impossible for changing in a practical manner. Ranges of IP addresses and subnet masks vary significantly across different networks. The IP address and a network mask must match the requirements of the network to connect to a network. Hard-coded network settings in the back-end PC make it impossible to connect

to a network with different settings. Furthermore, two of such machines cannot coexist on the same network and will create IP conflicts since IP addresses must be unique to each device on a network.

Figure 3. EMCO Concept Turn 155.

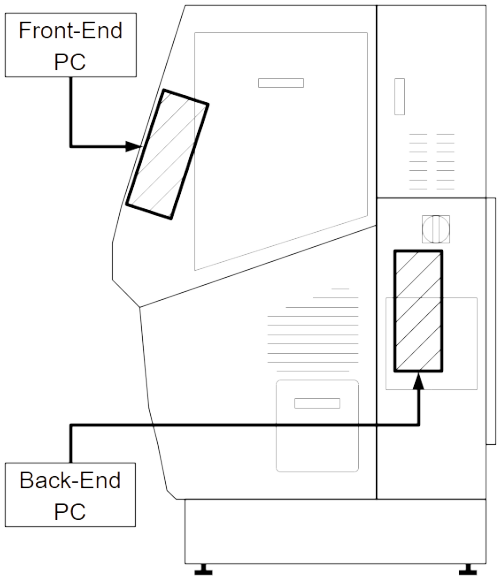

Figure 4. EMCO Concept T155 right side view.

The front-end PC internet protocol (IP) address is set to 192.168.10.15, and the back-end PC IP address is 192.168.10.12. The routing prefix of the network is 24-bit-length, equivalent to the historically used subnet mask 255.255.255.0. The back-end PC utilises port number 1122 and the front-end PC listens on port number 1036 [12]. The EMCO Concept T155 network limitation can be overridden by a network address translation method. An affordable Single board computer, such as a Raspberry Pi, was used to establish a highly configurable NAT server to configure the network settings of the lathe according to the network requirements. Network communication with the back-end PC enables an operator to send control commands to the lathe directly over a network. Figure 5 presents a schematic diagram of the proposed NAT server configuration.

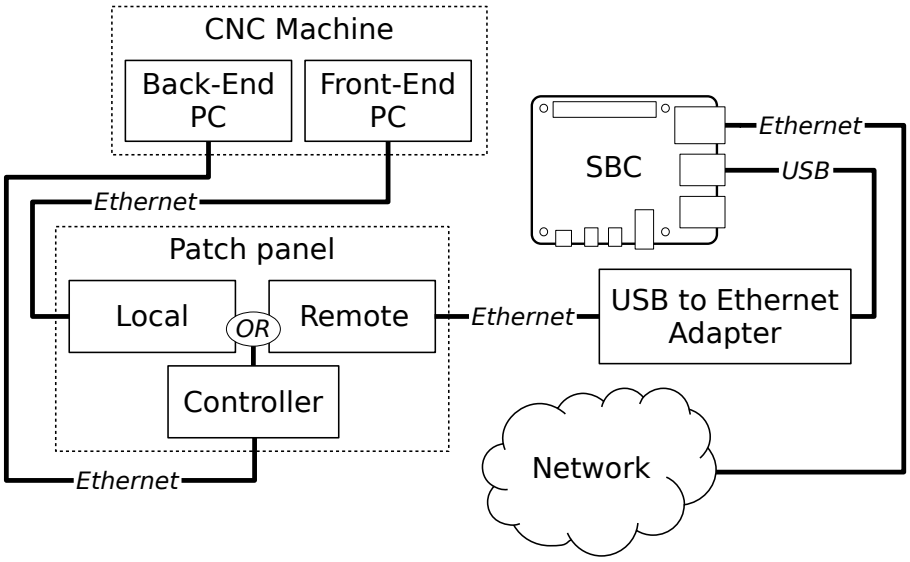

Figure 5. Principle NAT server arrangement.

In order to control the lathe, an investigation of its internal communications needs to be conducted beforehand. Since the front-end PC and back-end PC liaise over the Ethernet, the man-in-the-middle (MITM) technique was chosen for conducting the investigation. The MITM technique establishes a relay between two parties on a communication channel without these parties knowing about the existence of the relay. This allows parties to communicate normally. This relay could then be used to listen to or interfere with the communication. After harvesting the communication data, the data can be processed for ease of use and analysed to extrude necessary commands.

Specific preparations are necessary to perform the MITM technique successfully. In the case of EMCO Turn 155, a relay for MITM will be established over an Ethernet cable and expected to deal with TCP. For this reason, a computer with two network interface controllers NIC is required. Additionally, a packet analyser software or sniffer is essential for capturing packets. An average laptop with a USB-to-Ethernet adapter was used. Wireshark 3.2.7 as a packet analyser on Debian 10 (Buster) OS was utilised for the software side.

The original Ethernet cable connecting the front-end PC and the back-end PC was removed. Instead, additional Ethernet cables were installed so the laptop would intervene in the Ethernet link between the two PCs. The software side of the technique requires a network bridge between the two NICs. The network bridge should permit free communication linking the front-end PC and back-end PC to ensure normal operation of the CNC machine.

Communication data samples were collected by executing single commands from the operator panel of the lathe and listening on either NIC of the network bridge with packet analyser software. Every captured packet consists of protocol control information and the payload. Since only the payload may contain the information in question, only the payload was investigated. Packets transferred over a network are usually shown in hexadecimal representation because ASCII characters are not always appropriate. After capturing, sets of communication data were exported from Wireshark into text files for processing. A sample of only two packets of unprocessed data with 0 and 8-bit payloads is shown in Figure 6 from such file.

As can be seen, the resulting file contains a significant amount of information, most of which is unwanted for the investigation. Since each file contains around 1000 lines of text or more, manual reading or editing is impractical. The files were processed in a Vim text editor by employing a set of macro instructions to automate the process of making data samples more human-readable. Only needed fields were left, such as packet number, time, the last octet of source, destination IP addresses, payload length in bits, and hexadecimal

representation of the payload. Every field is separated by tabulation characters, making it possible to import the data into a spreadsheet or other software for further analysis.

```
No.     Time           Source                Destination           Protocol Length Info
     31 3.785773258    192.168.10.15         192.168.10.12         TCP      60     1036 →
1122 [ACK] Seq=49 Ack=81 Win=63360 Len=0

Frame 31: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface 0
Ethernet   II,   Src:   KontronE_db:c8:7a   (00:00:60:db:c8:7a),   Dst:   InsideTe_14:61:c3
(00:e0:f4:14:61:c3)
Internet Protocol Version 4, Src: 192.168.10.15, Dst: 192.168.10.12
Transmission Control Protocol, Src Port: 1036, Dst Port: 1122, Seq: 49, Ack: 81, Len: 0

No.     Time           Source                Destination           Protocol Length Info
     32 4.026462259    192.168.10.15         192.168.10.12         TCP      62     1036 →
1122 [PSH, ACK] Seq=49 Ack=81 Win=63360 Len=8

Frame 32: 62 bytes on wire (496 bits), 62 bytes captured (496 bits) on interface 0
Ethernet   II,   Src:   KontronE_db:c8:7a   (00:00:60:db:c8:7a),   Dst:   InsideTe_14:61:c3
(00:e0:f4:14:61:c3)
Internet Protocol Version 4, Src: 192.168.10.15, Dst: 192.168.10.12
Transmission Control Protocol, Src Port: 1036, Dst Port: 1122, Seq: 49, Ack: 81, Len: 8
Data (8 bytes)

0000  13 60 00 00 00 00 00 00                        .`......
```

Figure 6. Sample of exported data.

During the booting process of the lathe, the front-end PC and back-end PC communications—besides exchanging short commands—involve sending configuration files that are compulsory for a successful boot. This process is similar to the EMCO Concept Mill 105 machine [11] with a difference in the file content. Both machines share the same limitations; any single packet of data sent to the back-end PC must not exceed 512 bits. In practice, this means that data in sizes more than 512 bits, such as configuration files, must be broken into chunks of 512 bits or less and consequently sent to the back-end PC. Packets coming from the back-end PC can be more than 512 bits. Since files are transferred in packets, although the process of capturing files is the same as described above, post-processing is divergent. The payload from a set of captured packets representing a single file should be extracted, and then the hex dump should be combined into one file in the order it is received. The resulting hex file needs to be converted into a binary one. This could be accomplished using the `xxd` tool on Linux.

Captured and preprocessed data were analysed to determine a hexadecimal representation of commands sent from the front-end PC to the back-end PC responsible for different tasks. Additionally, reply and auxiliary patterns were identified and suggested. After establishing a connection, the back-end PC sends a command, which indicates that it is ready to receive initialisation settings files. The front-end PC replies with eight files, each prefixed with an 8-bit command, except the last one. The back-end PC replies with a series of commands and files and finally indicates the end of the initialisation process. Most other tasks are accomplished in a much simpler form, usually a request followed by a reply in one or two packets ranging from 8 to 40 bits each. Furthermore, during the connection, the two PCs broadcast auxiliary packets. These auxiliary packets can be seen as repeated patterns in the communication stream. Depending on the state of the machine, this pattern can represent the current mode of the lathe, a timer, or a ping indicator.

Digitalising a machine tool based on the proposed approach enables users to communicate with a machine in the same way the front-end PC communicates. Similarly, other connected machines can exchange data. All commands that can be executed from the front-end PC can be initiated through the digitalisation approach described. Moreover, all messages broadcasted by the back-end PC, such as errors, alarms, and current coordinates, can be captured. Furthermore, this approach allows greater flexibility since the working algorithm can be redesigned according to the user's needs. For instance, a different algorithm can update the toolpath on the fly or initiate action on the reception of an alarm or error from the back-end PC.

A network connection to the back-end PC is established via a patch panel, which also provides backward compatibility with the front-end PC. The patch panel has three Ethernet ports named "Local", "Remote", and "Controller". The port named "Controller"

represents the back-end PC; "Local" and "Remote" represent the front-end PC and network connection, accordingly. Using a patch cable, an operator can specify from which source control the commands are sent to the controller. The NAT server is configured on Raspberry Pi with a second NIC connected through a USB port, obligatory for the NAT method. Raspberry Pi needs to be shut down properly, initiating the process with appropriate software to prevent possible damage to the file system. A momentary switch plays the role of a power button for Raspberry Pi, and an integrated LED indicates the status of the SBC. Moreover, 220- and 24-volt power lines for the Raspberry Pi power supply and LED were drawn from the electrical cabinet of the lathe. A relay was used to control the 24-volt LED from the Raspberry Pi general-purpose input/output (GPIO).

4. Results

The working concept was built to prove the functionality of the proposed framework. This arrangement included the IoT-enabled EMCO Concept T155 CNC lathe, a JavaScript web server running on a Linux laptop, a network switch with a Wi-Fi access point, and a gyroscope-enabled mobile phone. The laptop and the IoT-validated lathe with the NAT method were connected to a network switch through Ethernet. The JavaScript server running on the laptop was set to liaise with the back-end PC of the lathe through the port forwarding provided by Raspberry Pi on the lathe. The Wi-Fi access point on the network switch was configured to be a part of an established Ethernet network. The mobile phone was connected to the network over Wi-Fi. By connecting to this network over the Ethernet or Wi-Fi with any device with a browser and JavaScript support, a user could initialise the lathe, activate auxiliary drivers, reference the machine, and control movements of the axis through a web interface provided by the server.

Furthermore, to discover new features of the concept, it was possible to utilise a gyroscope on a mobile phone to control movements of the lathe axis. While rocking the mobile phone around the lateral or longitudinal axes, the gyroscope reading was captured by a JavaScript program on a web server page, converted into appropriate commands, and sent to the lathe in real time. The proposed framework requires no special software on the user or machine side. Virtually any device with a web browser can provide access to this functionality to the user. Additionally, the same framework was proven to work on the EMCO Concept 105 Mill CNC machine with the same success.

The user can initialise, switch on auxiliary drivers, and reference the machine by navigating to the JavaScript server's web page from a smart device, such as a mobile phone with a browser. After the referencing is complete, the user can switch the machine to "Jog" mode from the web page and proceed to a dedicated page to capture the mobile phone's gyroscope readings. A JavaScript function reads the phone's current position and sends it to the server, where the data are converted into the appropriate hex code and passed on to the machine for execution. This algorithm is executed in a loop a few times a second. Communication between the server and the client is established over a WebSocket connection, making it possible to control the machine's axes in a near real-time manner. However, software-defined safety control using a mobile device for handling a machine in a factory setting requires a separate study.

Most of the machine's safety features are realised through fundamental electronic components, such as relays, switches, and circuit breakers. Due to such an arrangement, most safety features are still in place and active, even employing the discussed method. Moreover, since the user can implement new features, some could be dedicated to safety, for example, the implementation of collision prevention with the use of a DT. Similarly, as in the DIKW pyramid, the data assembled into information from the machine can produce knowledge, and wisdom can be produced with the use of knowledge [2]

5. Conclusions

While employing low-cost technologies, retrofitting can solve numerous concerns. Many still fail to confront the issues that expiring machines bring to companies. This

study shows that the digitalisation of former assets could be accomplished even under a strict budget of SMEs and with relatively small time investments. Moreover, digitalisation adds value to the existing machinery and opens doors for customised functionalities and deeper integration with smart factories. Several initiatives, such as the shoestring approach, were recently created to facilitate access to digital tools and support digitalisation for SMEs [13,14].

CNC machines, robotic assembly lines, and collaborative robots (Cobots) could be potential IoT-enabled assets in the Industry 4.0 era and in the interaction between humans and machines in the next industrial revolution (to achieve higher sustainability and resilience) [15]. Humans work alongside machines and are connected to intelligent manufacturing plants via smart devices, such as CNC 4.0, which will likely continue toward more advanced human–machine interfaces. This will mean improved integration, allowing faster, better automation paired with the power of human brains. Therefore, further investigations and case studies concerning automating, loading, and tending CNC machines with Cobots for Industry 5.0 are expected.

Author Contributions: Writing—original draft, A.A.; Writing—review & editing, S.A., J.P. and X.X.; Supervision, X.X. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Not applicable.

Acknowledgments: The authors would like to express appreciation to the researchers and staff at the Laboratory for Industry 4.0 Smart Manufacturing Systems (LISMS) at the University of Auckland.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

SMEs	small and medium-sized enterprises
CPS	cyber-physical system
DT	digital twin
AR	augmented reality
NC	numerical control
CNC	computer numerical control
OAC	open architecture controller
HMI	human–machine interface
NCK	numerical control kernel
PLC	programmable logic control
STEP-NC	standard for the exchange of product data compliant numerical control
CPMT	cyber-physical machine tool
MTCT	machine tool cyber tween
MT	machine tool
MITM	man-in-the-middle
TCP	transmission control protocol
NIC	network interface controller
OS	operation system
PC	personal computer
RTOS	real-time operating system

References

1. Aheleroff, S. Mass personalisation as a service in industry 4.0: A resilient response case study. *Adv. Eng. Inform.* **2021**, *50*, 101438. [CrossRef]
2. Aheleroff, S. Digital twin as a service (DTaaS) in industry 4.0: An architecture reference model. *Adv. Eng. Inform.* **2021**, *47*, 101225. [CrossRef]
3. Liu, C.; Zheng, P.; Xu, X. Digitalisation and servitisation of machine tools in the era of Industry 4.0: A review. *Int. J. Prod. Res.* **2021**, 1–33. [CrossRef]