A Digital Twin Implementation
for Manufacturing Based on Open-Source
Software and Standard Control Systems

Christian Dalheim Øien$^{(\boxtimes)}$, Håkon Dahl, and Sebastian Dransfeld

SINTEF Manufacturing, Enggata 40, 2830 Raufoss, Norway
christian.dalheim.oien@sintef.no

Abstract. The concept of Digital Twins (DTs) can be utilized to solve complex problems in manufacturing based on the principle of Cyber-Physical Systems (CPS). While there are several reference architectures for CPS, there seems to be a knowledge gap between such high-level outlines and actual shopfloor-level implementations. This paper focuses on process control applications of the DT concept and proposes a specific implementation setup using fieldbus communication with a computer, aiming to fulfil a set of defined requirements. The setup is tested in an industrial use-case and the resulting characteristics of the solution are presented. A resulting assertion is that a DT of a process must be as specialized and customized as the system controlling it.

Keywords: Digital twins · Cyber-physical production systems · Smart manufacturing · Process control

1 Introduction

The principle of Cyber-Physical Systems (CPS) is often a desired base for implementing decision-making autonomy and self-adaptiveness of machines in industry. A CPS is the integration of a virtual world that interacts with a physical world [1]. Through interconnection, the computational part of the system can monitor, control and affect the physical process, and vice versa [2]. Lee et al. proposed the 5C reference architecture for implementation of a CPS [3] which describes five necessary functional levels named connection, conversion, cyber, cognition and configuration. This gives a conceptual understanding of the main constituents of a CPS, but they do not correspond specifically to necessary technology or implementation methods.

Another Smart Manufacturing approach that has been given much attention in research recent years is the Digital Twin (DT) concept. While dating back to 2003 as a virtual representation of a physical product or process [4], a consensus of its actual meaning is still not reached. A literature review conducted by Kritzinger et al. [5] categorized Digital Twin-related papers based on data integration level between physical assets and their digital representation. This resulted in the following three-level classification

scheme; 1) Digital Models are characterized by manual data transfer, 2) Digital Shadows are characterized by one-way automatic data transfer from physical to digital, and finally that 3) *Digital Twins are characterized by two-way automatic data integration.*

In slight contrast to Kritzinger et al. [5], VanDerHorn and Mahadevan [6] proposed, on the basis of on 46 different definitions found in literature, a generalised definition of a DT as "*a virtual representation of a physical system (and its associated environment and processes) that is updated through the exchange of information between the physical and virtual systems*". Consequently, this definition does not strictly require a two-way data integration and invites for a more open understanding. VanDerHorn and Mahadevan in fact stated that a requirement of two-way data interaction is restrictive and could exclude applications where the output is not intended to be used automatically in a machine-interaction but, for instance, as an operator interaction via a human-machine-interface (HMI). As a contrast, it is an underlying perspective behind this paper that the transition of small and medium-sized enterprises (SMEs) towards digital manufacturing rely on data integration solutions at process level, and specifically, that they need to be cost-effective, use open-source software, need relatively few hardware components, and are based on an easily understandable framework. At the same time, it seems this perspective is little represented in current research.

Regardless of the definition used, several sources found in literature describe CPS and DT as high-level shopfloor integrations that can enhance a large part of a value chain. While such integration can bring significant value to larger companies, it requires complex and expensive solutions that would be inappropriate for most SMEs that, in the sense of digital transformation, are more likely to benefit from process level enhancements. Furthermore, as this section is intended to show, DT concepts and implementations found in literature are often used as simulation models with the purpose of conducting offline studies leading to optimizations of production plans or layouts, and not as a means of process enhancement based on data integration.

1.1 State of the Art of Manufacturing DT Applications

Tao et al. [7] have done a systematic review of the state-of-the-art of industrial DTs and concluded that production control and prognostics and health managements were the most relevant areas covering more than two-thirds of the applications. Out of 50 papers this review points to only one related to an interacting and collaborating DT, i.e., with a data integration focus. That paper, written by Vachálek et al. [8], describes a setup that simulates an assembly line for hydraulic pistons in order to optimize its production plan, using the proprietary software SIEMENS Tecnomatix Plant Simulation [9]. The study does not involve a two-way data integration providing feedback to the physical process, but proposes, however, an enhanced operator feedback based on potential inconsistencies between simulation result and actual measurements.

The aforementioned literature review by Kritzinger et al. [5] covers 43 DT-related papers, out of which 12 describing case-studies, and characterized only one of them as using a two-way data integration. In that study, Bottani et al. [10] implemented a DT in a laboratory setting for independent decision making of an automated guided vehicle (AGV) in the sense that scheduling problems solved by a decision-making algorithm

were executed remotely in addition to locally on the AGV microcontroller. The paper, however, does not state which value or enhanced functionality this lead to.

Cimino et al. [11] reviewed and categorized 52 manufacturing related DT applications out of which 19 were classified as aiming to monitor and improve the production process. Only one out of these involved process enhancement, namely Karanjkar et al. [12] who used a DT approach to suggest an enhanced process flow in a Printed Circuit Board (PCB) assembly line that would reduce its energy consumption. In this application, analyses were done offline based on historical data in order to tag machine states and then calculate the energy consumption per state. Based on these state-wise energy consumptions and a discrete-event simulation model made using SimPy [13] the authors identified process flow enhancements. The generated information was, however, neither used for process control via a data connection back to the control system of the production line nor for operator feedback via HMI.

1.2   Research Question and Scope

None of the above three referenced DT applications [8, 10, 12] exemplify an automated use of information generated by the DT. This paper, however, focuses on using the DT concept for enhanced control of manufacturing processes that, due to complexity and varying conditions, cannot be robustly and accurately controlled by a standard process control algorithm using a programmable logic controller (PLC) only. Instead, in such cases, the control accuracy could be increased through a two-way data integration between the PLC and a DT enabling a more advanced process control based on real-time analyses. The research question that this paper seeks to answer is – *What is a viable method for implementing a process control DT in an SME, that ensures the needed functionality while being easy to understand, maintain and adjust?* In order to do this, a set of requirements for such DT implementations is suggested in Sect. 2, as well as an explicit implementation method that fulfils those requirements. An industry use-case is described in Sect. 3 where the resulting DT characteristics and performance is described while a discussion is given in Sect. 4. The use-case itself is not concluded and its specific results are therefore not part of this paper.

2   Method

In an industry use-case related to increasing the process control robustness and accuracy of a machine, a DT with two-way data integration was developed. In the work with defining a suitable implementation setup, the following requirement was defined – *A process Digital Twin implementation should ensure programmability and availability of support through standardized software, without impeding the desired functionality.* An application written in Python running on an industrial PC (IPC) equipped with a high-speed PLC communication interface was consequently chosen as an outline for the DT.

Python is one of the most used general-purpose programming languages and it gives the application a fair chance of being understood by operations or maintenance engineers. Other advantages are a high number of available libraries and easily accessible support from a vast web user community. These were all desired key characteristics of the solution.

In terms of connectivity, the use of a fieldbus interface was chosen. As industrial edge computing solutions are designed to enable fast database interactions and applications with reduced bandwidth consumption, its general area of application is to store and do computations on production data. However, in most cases it is not used for deterministic, millisecond-level, two-way communication with PLCs. Based on the categorization done in [5] one could claim that industrial edge computing is mainly suitable for digital model or digital shadow functionality when it comes to single process applications. In order to achieve a two-way data integration with a PLC and the mentioned communication characteristics, a fieldbus communication interface seemed to be the best option. For practical reasons, PROFINET was chosen.

As a conclusion, the following specific requirements are suggested for SME DT applications related to process control enhancements.

- Uses deterministic, millisecond-level, two-way communication with one PLC
- Facilitates easy visualization and analysis of received PLC data
- Is based on program code that is intended to be easy to maintain and customize
- Is cost-effective

2.1 Industrial Use-Case

The studied use-case focuses on the control of two magnetic induction ovens that preheat aluminium extrusion billets. These billets are cast cylinders of aluminium alloy approximately 1 m in length and 0.3 m in diameter. The ovens heat up one billet each from room temperature to about 500 °C with a cycle time of around 4 min. They work by each rotating four separate ring-shaped assemblies of permanent magnets around the billets at about 1500 rpm using an individual electrical motor per magnet assembly. The heating process is intended to ensure a certain temperature profile along the longitudinal direction of the billets which is beneficial for the proceeding extrusion process.

As the magnet rings cover most of the billets their temperatures are only measured on the billet ends during heating, and then measured on several surface positions along the billet length after completion of each process cycle. Process control is ensured by a SIEMENS PLC maintaining rotational speed and longitudinal position of the four magnet rings according to a pre-defined recipe and continues the process until set-point temperatures at the billet ends have been reached. This control method is, however, unable to accurately account for the following characteristics and dependencies.

- The magnetic field from each single magnet is non-uniform.
- The magnetic field strength is dependent on the temperature of the magnets.
- The billet specific heat capacity is dependent on local aluminium temperature.
- An end-effect reduces the magnetic induction in regions close to the billet ends.

Due to the variation resulting from these characteristics the original heating process control leads to an accuracy of around 5% for the temperature profile. This causes an unwanted variability of the main extrusion process leading to suboptimal stability of product quality and overall equipment effectiveness (OEE). The aim of the DT approach is to combine the capabilities of the PLC with the computational power of an IPC and increase the accuracy to around 1%. Based on finite element analysis (FEA) and thermal dynamics calculations, the DT setup will allow for an estimation of the final temperature profile to be regularly updated and returned to the PLC for usage in an improved control routine. Based on initial work with FEA, it is estimated that the temperature profile can be calculated within one or two seconds which makes it possible for the PLC to do 100 logic-based adjustments during a single process cycle.

2.2   Digital Twin Implementation

The DT setup in this use-case facilitates the use of Python-based software to perform estimations and calculations on an IPC, along with a communication interface between the PLC and IPC via PROFINET fieldbus. An Ixxat® INpact PIR Slave PCIe [14] dedicated device card on the IPC was used to send and receive data in the form of struct object instances. The struct types were defined on the PLC in Structured Text (ST) according to IEC 61131-3.

Specifically, three main parts of source code ensure the DT functionality. Firstly, 288 lines of Python code that translates the ST struct definition from the PLC to Python. Secondly, 772 lines of C code and 118 lines of Python code handling the I/O communication between the PLC and the actual DT Python program. This is done by making use of an application programming interface (API) with the device card (which requires the C code). Thirdly, 20 lines of Python code defining a main loop that sets up the connection and works as interface for the user program.

By this setup, the bytes of data that the struct consist of can be sent between the PLC and the IPC without overhead, as the C code receives and forwards them via a socket connection to the Python program which has been provided with identical struct definitions. This communication concept is illustrated in Fig. 1.
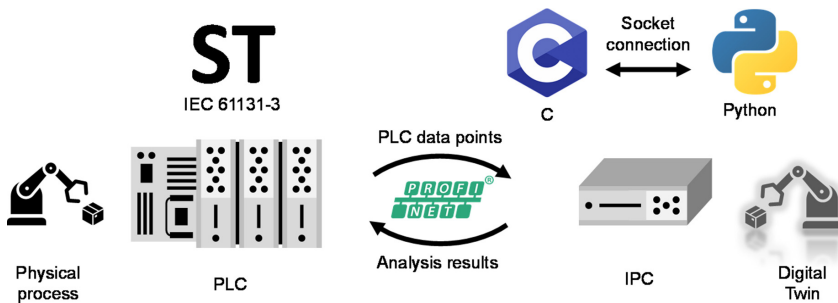

Fig. 1. Process Digital Twin (DT) communication concept. The DT is constituted by a Python program on an IPC where a separate program written in C regularly exchanges data with the PLC over PROFINET fieldbus.

By exchanging data on a predefined, bytewise format there is no need of descriptions or metadata to be sent on every communication cycle. This way the necessary amount of transferred data is reduced to a minimum, and a millisecond-level communication between the DT and PLC is possible, depending on the amount of data. It is due to the identically defined struct datatype that this is achieved. To visualise the concept, example definitions in Structured Text and Python are shown in Fig. 2. This specific struct would require 29 bytes of data.

```
TYPE "Regulator_Data"
VERSION : 0.1
  STRUCT
    Version : Byte;
    timestamp : UDInt;
    alloy : DInt;
    length : Real;
    actual_speed_rpm : Real;
    actual_motor_temperature : Real;
    actual_torque : Real;
    temperature_front : Real;
  END_STRUCT;

END_TYPE
```

```
class RegulatorData(BigEndianStructure):
    _pack_ = 2

    _fields_ = [
        ("version", c_byte),
        ("timestamp", c_uint),
        ("alloy", c_int),
        ("length", c_float),
        ("actual_speed_rpm", c_float),
        ("actual_motor_temperature", c_float),
        ("actual_torque", c_float),
        ("temperature_front", c_float),
    ]
```

Fig. 2. Struct definition code excerpt in Structured Text (left) automatically translated to Python (right). An instance of this struct would require 29 bytes of data.

3   Results

The key result of this study is that the described DT implementation setup enables usage of gathered PLC data points as direct input to an advanced analysis routine, and reversely that it enables the use of the results from that routine as direct input back into the PLC. Generally, any type of analysis could be suitable for this setup, including finite element analysis (FEA), model-based analysis, statistical analysis, and machine learning.

The cycle time for the PROFINET fieldbus was set to 2 ms, which can be handled when sending up to 1440 bytes of data in each direction for each cycle. In a communication test the use-case PLC was set to tag each data package with its internal millisecond clock in order to evaluate jitter. The jitter distribution in Table 1 is based on this test, which had a duration of about 20 min.

Table 1. Number of sent data packages vs. their respective measured cycle time.

	2 ms	3 ms	4 ms	>4 ms
PLC to IPC	543394	14242	247	
IPC to PLC	545187		12692	4

In other words, the set cycle time of 2 ms was achieved for about 97,5% of the sent data packages in both directions. Due to the 2 ms PROFINET cycle time and since new

data was only registered at the IPC if the timestamp had changed, we see that occasional packages at 3 ms from the PLC was converted to 4 ms at the IPC. In any case, the IPC-to-PLC communication was shown to be similarly deterministic as the PLC-to-IPC communication. This shows that the setup of C code forwarding data via socket to the Python code handles the 2 ms cycle time well.

For the use-case 446 bytes is sent from PLC to IPC at each cycle, and 182 bytes in the other direction. This is a mix of process parameters that changes sporadically and continuously changing variables. The data is connected to 3 processes, namely the two ovens and a proceeding temperature measuring station. From the aforementioned communication test the sent and received data were stored yielding a total of 174 MB in csv format. However, this was reduced to 9,3 MB by converting the data to Parquet format which is only about 5%. For the single process studied in this use-case it would mean a yearly amount of about 5 TB uncompressed, or 250 GB compressed. On the IPC, reading the csv-data took about 7 s while reading the Parquet-data took about 0.5 s. Hence, the achieved compression ratio makes it realistic to continuously store and analyze data from such systems.

4   Discussion

In this paper it has been intended to substantiate that DT applications meant to be easy to program, customize, and maintain are more likely to succeed than proprietary or turn-key solutions. In other words, as manufacturing processes are predominantly customized for its specific purpose, a DT of a process must be as specialized and customized as the system controlling it. One could say that the specificity required on PLC level, or the 5C connection level [3], will also be required in the higher levels of a CPS framework. Based on this assumption a standardized DT solution with limited configuration possibilities will always yield limited functionality.

The purpose of this study has been to define and evaluate a process level DT setup that achieves a two-way automatic data integration with its physical counterpart. The main motivation for this was to enhance the control of a machine in order to increase product quality and the OEE of the connected production line. It is the authors' understanding that such process level data integration will play an important role in digital transformation of manufacturing SMEs, laying the foundation for higher-level integrations in terms of data, competence, and system ownership. This is a contrast to similar studies found in literature where DT applications do not utilize two-way data integration but instead provide a basis for discrete improvement or enhancement actions.

While the intended usage of FEA is relevant at a second-level communication rate, it was still the wish in the described use-case to establish a millisecond-level two-way communication and a millisecond-level dataset for future analysis. It is not unlikely that other faster analysis methods could be used to further enhance the performance. Furthermore, Parquet compression gave tolerable data amount from continuous logging and does not significantly compromise the reading speed. Keeping a fully detailed dataset for future work was therefore seen as valuable.

References

1. Negri, E., Fumagalli, L., Macchi, M.: A review of the roles of digital twin in CPS-based production systems. Procedia Manufact. **11**, 939–948 (2017)
2. Lee, E.A.: Cyber Physical Systems: Design Challenges. IEEE (2008)
3. Lee, J., Bagheri, B., Kao, H.-A.: A cyber-physical systems architecture for industry 4.0-based manufacturing systems. Manuf. Lett. **3**, 18–23 (2015)
4. Grieves, M.: Digital Twin: Manufacturing Excellence through Virtual Factory Replication (2015)
5. Kritzinger, W., et al.: Digital twin in manufacturing: a categorical literature review and classification. IFAC-PapersOnLine **51**(11), 1016–1022 (2018)
6. VanDerHorn, E., Mahadevan, S.: Digital twin: generalization, characterization and implementation. Decis. Support Syst. 113524 (2021)
7. Tao, F., et al.: Digital twin in industry: state-of-the-art. IEEE Trans. Industr. Inf. **15**(4), 2405–2415 (2019)
8. Vachálek, J., et al.: The digital twin of an industrial production line within the industry 4.0 concept. In: 2017 21st International Conference on Process Control (PC) (2017)
9. SIEMENS. Plant Simulation & Throughput Optimization. https://www.plm.automation.siemens.com/global/en/products/manufacturing-planning/plant-simulation-throughput-optimization.html
10. Bottani, E., et al.: From the Cyber-Physical System to the Digital Twin: the process development for behaviour modelling of a Cyber Guided Vehicle in M2M logic (2017)
11. Cimino, C., Negri, E., Fumagalli, L.: Review of digital twin applications in manufacturing. Comput. Ind. **113**, 103130 (2019)
12. Karanjkar, N., et al.: Digital twin for energy optimization in an SMT-PCB assembly line. In: 2018 IEEE International Conference on Internet of Things and Intelligence System (IOTAIS) (2018)
13. Overview — SimPy 4.0.2.dev1+g2973dbe documentation. https://simpy.readthedocs.io/en/latest/. Accessed 10 Mar 2021
14. Ixxat®. Industrial Ethernet and fieldbus interface for PCIe, PCIe Mini and M.2. 2021. https://www.ixxat.com/products/products-industrial/inpact-overview/ixxat-inpact—profinet-irt. Accessed 26 Jan 2021