POLITECNICO DI TORINO
Repository ISTITUZIONALE

Digital Transformation of a Production Line: Network Design, Online Data Collection and Energy Monitoring

Original
Digital Transformation of a Production Line: Network Design, Online Data Collection and Energy Monitoring / Dall'Ora, N., Alamin, K.S.S., Fraccaroli, E., Poncino, M., Quaglia, D., Vinco, S.. - In: IEEE TRANSACTIONS ON EMERGING TOPICS IN COMPUTING. - ISSN 2168-6750. - 10:1(2022), pp. 46-59. [10.1109/TETC.2021.3132432]

Availability:
This version is available at: 11583/2944812 since: 2021-12-13T14:13:01Z

Publisher:
IEEE

Published
DOI:10.1109/TETC.2021.3132432

Terms of use:

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

Publisher copyright
IEEE postprint/Author's Accepted Manuscript

©2022 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collecting works, for resale or lists, or reuse of any copyrighted component of this work in other works.

(Article begins on next page)

27 November 2026

Digital Transformation of a Production Line: Network Design, Online Data Collection and Energy Monitoring

Nicola Dall'Ora, *Student, IEEE,* Khaled Alamin, *Member, IEEE,* Enrico Fraccaroli, *Member, IEEE,* Massimo Poncino, *Fellow, IEEE,* Davide Quaglia, *Member, IEEE,* Sara Vinco, *Member, IEEE*

Abstract—The concept of Industry 4.0 originates from the will to introduce the benefits of digital computation into new and existing industrial plants to save time, materials and energy. The digital transformation requires that all machinery of the production line are connected together and with the enterprise applications, to capture and analyze data across all manufacturing stages. Then, such collected data can be exploited to take strategic decision on the production and to monitor it, reacting to unexpected behaviors and thus reducing downtime and maintenance costs. This article aims at supporting production engineers approaching digital transformation by exemplifying its key elements on a real life scenario, the Industrial Computer Engineering laboratory of the University of Verona. First of all, the article discusses network design, as communication is an enabler of the other technologies. Network is realized through automatic network synthesis from requirements and characteristics of the production line data flow. Then, the paper discusses data collection and the construction of a digital twin monitoring power consumption of the production line, with the goal of detecting any discrepancy between real time data and digital twin data. This allows to trigger an early intervention on the line, to guarantee an effective maintenance.

Index Terms—Industry 4.0, digital twin, IIoT sensors, online data collection, energy monitoring, network design.

Author version

1 Introduction

In the last decade, the technological advancements, both in terms of networking, electronics, and information systems, deeply affected the manufacturing environment at all levels: both existing production lines and under construction ones now can consider opportunities that might have been marginal until now [1]. Every industry must thus adapt to external changes involving people, economy, and the environment: it must react quickly to survive in the extreme globalization context, while at the same time reducing costs and downtime to the minimum.

To support these changes, production management must have decision-making processes and self-configuration capabilities that are as flexible as possible [2]. Information that comes from the production line about its real-time evolution can be used to support the decision-making process, so to effectively react to external changes at every level.

This article proposes some solutions to *help the digital transformation of a production line* throughout its life span, from when it is designed and built to when it is reconfigured to produce different targets. It is important to note that all solutions proposed in this work can be applied to existing plants for their digital transformation. This is especially relevant considering that, even though more and more digital technologies are installed in shop floors every year, low-technology production systems are still existing, as the estimated life span of machine tools average out between 15 and 20 years [3]: it is thus desirable to extend the opportunities offered by smart manufacturing also to such obsolete production plants. To support production engineers in this transformation, this paper applies the main ingredients of digital transformation to a real-life scenario.

Network infrastructure design is the enabling step for all other technologies, as the communication infrastructure delivers data flows between all the different systems that make up the production line. When the number of systems to be connected is large and communications are subject to constraints, an automatic synthesis methodology is desirable.

In addition to flowing through the company network, data must also be efficiently stored and processed, to preserve its availability and value. This requires the construction of a *data collection and management architecture*, that involves data gathering from sensors placed on the line, its storage according to a precise organization (i.e., local or on cloud-based), and its pre-processing and cleaning, so to allow an effective analysis to do what-if analysis, research, development, and process optimization.

Finally, the collected data can be exploited to *monitor the operation of the production line* through the construction of a digital twin, that predicts plant behavior at run time and detects any unexpected discrepancy to activate maintenance and monitoring actions, to early identify any malfunctioning and reduce both downtime and costs [4]. In this perspective, *energy consumption* is a very crucial aspect of the production line since it is related to costs and

- *N. Dall'Ora and E. Fraccaroli and D. Quaglia are with the Department of Computer Science, University of Verona, Verona, IT. E-mail: name.surname@univr.it*
- *K. Alamin and S. Vinco and M. Poncino are with the Department of Control and Computer Engineering, Politecnico di Torino, Torino, IT. Email: name.surname@polito.it*

Manuscript received May 7, 2021; revised .
This work has been partially supported by the MIUR "Dipartimenti di Eccellenza" 2018-2022 grant.

waste of natural resources and can give insights on possible malfunctioning of machinery. Furthermore, energy monitoring allows to find inefficiencies (thus enabling production optimization) and to support the predictive maintenance process [5].

The paper outlines the key aspects that impact an *Industry 4.0* installation in the three aforementioned directions. Section 2 presents background. Section 3 describes the network synthesis problem for a production plant, and Section 4 focuses on the infrastructure for data collection. Section 5 addresses the concepts necessary to monitor the energy consumption of a plant. In Section 6, the concepts are exemplified on a physical setup in an Industry 4.0 research facility. Finally, conclusions are given in Section 7.

2 Background

2.1 Operations and communications

The industrial shop floor is managed by different types of applications, which are usually organized as a pyramid (shown in Figure 1). At the bottom, we have the field devices like sensors that inform the Programmable Logic Controller (PLC) about the status of machinery and actuators. Information circulating in such control loops requires low and constant propagation delay but usually not a huge bandwidth. The Supervisory Control and Data Acquisition (SCADA) system consists of applications and graphical user interfaces for high-level process supervisory management; it communicates with PLCs in the shop floor to configure machine-level control strategies. The Manufacturing Execution System (MES) is used to organize and track the transformation of raw materials into finished goods, and it provides information that helps decision-makers to understand how current conditions on the plant floor can be optimized to improve production output. Information flowing from the shop floor to the MES does not require low delay but its aggregated throughput can be large. Finally, the Enterprise Resource Planning (ERP) is a suite of applications that connects production to other high-level business activities related to suppliers and customers.

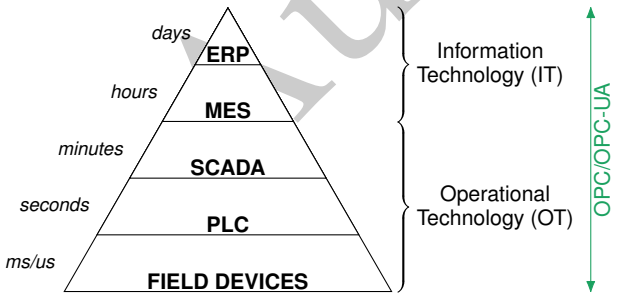

Fig. 1. Automation pyramid with corresponding applications, network types and typical communication delays.

These applications are hosted by two types of network: Information Technology (IT) and Operational Technology (OT) networks. ERP, SCADA, MES are all hosted by the IT network, while control applications like PLCs are hosted on the OT network. All these applications must guarantee efficiency and functional safety of the production line [6].

In Industry 4.0, it is essential to structure and collect as much information as possible regarding each level of the automation pyramid. The more information available, the greater the ability to promptly identify all the anomalies on the production line, *e.g.*, consumption, delays, and bottlenecks. However, data from the field are difficult to analyze if their semantics is not clearly defined. There are two emerging communication standards that propose to facilitate the organization of transmitted data, namely, Open Platform Communications (OPC) and OPC UA (OPC-Unified Architecture) [7]. These two standards focus on the exchange of data between programmable logic controllers (PLCs), human-machine interfaces (HMIs) and other industrial applications for the purpose of interoperability and information sharing between the production line layers. In the context of Industry 4.0, the concept of *digital twin* has been recently introduced to support decision making once the network infrastructure and a data collection mechanism have been set up.

2.2 Digital twin

The digital twin concept merges all digital transformation aspects: data, both sensed from the line equipment and generated by the company management infrastructure, is collected and merged to enable what-if analysis and future predictions, to monitor the evolution of the line at runtime, and to improve the production process [4], [8], [9]. The manufacturing physical space is thus connected with its virtual representation: the virtual part records the historical evolution of the physical plant and predicts its evolution with the goal of identifying malfunctioning and possible optimizations; meanwhile, the physical part provides sensed data and behaviors, to allow a continuous refinement and calibration of the virtual part.

The role of the digital twin in the lifetime of a production line is crucial at different stages [10], [11]: it acts as a *virtual prototype* of the production line, to evaluate its behavior before its actual implementation; it enables *effective decision making* to make informed data-driven decisions and reduce possible sources of inefficiencies; and it *monitors line operation at run time* and predicts its behavior, thus providing a reference golden model of its evolution for failure detection [12], [13].

To enable the construction of a digital twin, the production line must be equipped with a data collection architecture that is accurate, scalable and fulfilling real-time constraints [14]. This objective can be achieved through a careful network design.

3 Network design

Machine-to-machine communications as well as interactions with applications outside the plant are crucial in smart manufacturing. To support the designer in managing communication complexity we need to start from the identification of communication flows and synthesize the physical network infrastructure by satisfying plant constraints (e.g., required quality of service) and optimizing a given metric (e.g., overall cost). This network design can be effectively adopted both for new production lines and for exiting production

Author version

lines, thanks to the flexibility offered by the more recent network technologies, like the Internet of Things (IoT) [15].

3.1 Communication requirements

In the simple example in Figure 2, factory layout is annotated with *tasks* (denoted by circles) and their *data flows* (denoted by arrows). Red circles represent sensing and actuation tasks which generate measurements and apply commands, respectively, and control tasks (i.e., PLCs) that interact with sensing and actuation tasks (red arrows) to implement control strategies. Tasks' position is a design requirement, i.e., in the discussed example, the designer stated that the mobile robot has local control tasks while the transport belt is controlled by a software process hosted in the office. Blue circles denote cooperation tasks that interact to coordinate the movement of pieces between the mobile robot and the transport line. Green circles denote monitoring tasks that collect data from the environment, robots, machinery and belt for further analysis (e.g., for predictive maintenance). Green data flows originate from a large number of sensors in the shop floor and are directed towards the digital twin process hosted in the office.

From the communication perspective, data exchanges on different arrow types have different typical requirements. Red data flows require very low delays (around 1-10 ms) while the data rate is usually below 400 kb/s. Blue data flows require moderate delays (around 10-100 ms) with a data rate around 20 kb/s. Green arrows have no constraints on delay but exhibit a very high data rate of about 1 Mb/s. *All these Quality of Service requirements should be considered in the design of the OT network infrastructure.*

The design of the OT network should also take into account the position of tasks in the factory layout. As depicted in Figure 2, the drilling station, the mobile robot and the transport belt occupy specific areas in the shop floor and are separated from the office. In the specific case of the transport belt, a sensor should be placed at each end of the equipment, while the actuator should be placed in the left end. *The specification of the spatial displacement of the data flows end points affects the design of the network.* For instance, the two sensing tasks of the belt cannot be hosted by the same node, while the sensing task and the actuating task in its left end can be implemented in the same node if this choice reduces costs. *Furthermore, the size of the rooms and the presence of walls affect the network topology both for wired and wireless communications, while we can assume that network properties are homogeneous inside a room.* To group tasks inside areas and to represent areas of homogeneous network properties we introduce the concept of *zone* denoted in Figure 2 by rounded boxes. Finally, the network design methodology should consider that the mobile robot is a *moving object*. Data exchanges for vehicle control (red arrows) are inside the zone represented by the robot and thus they can be handled by a static network while data exchanges for cooperation and monitoring should be handled by a mobile (i.e., wireless) communication channel.

3.2 Methodology for Network Synthesis

The physical network infrastructure to be added to the plant can be considered as a container of data flows necessary to

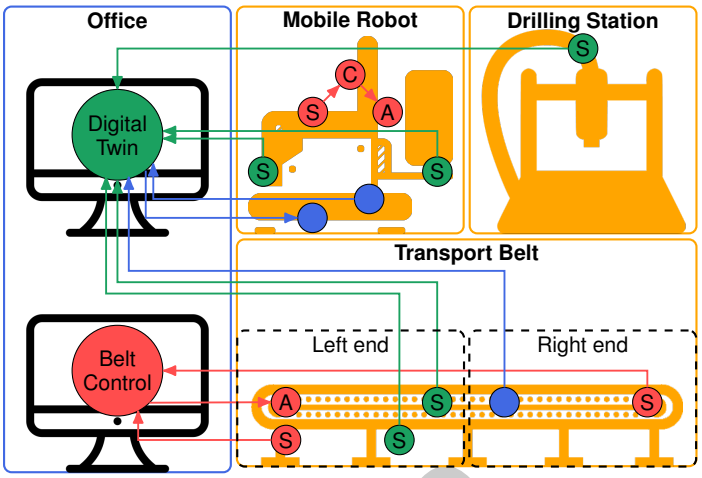

Fig. 2. Example of production plant information for network design.

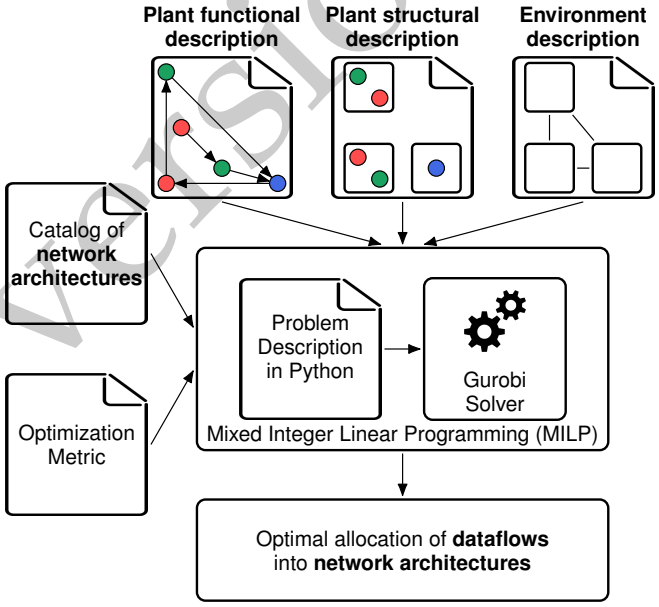

Fig. 3. Schematic representation of the network synthesis methodology.

exchange information to improve manufacturing. The problem of the automatic allocation of data flows into physical channels and network protocols can be seen as a *Network Synthesis* problem as stated in [16]. For this purpose, the designer should map the communication requirements of the plant onto the set of entities formalized in the network synthesis approach, i.e., tasks, data flows, abstract channels, zones and contiguity relationships [17].

Figure 3 describes the methodology for network synthesis. The functional description of communications inside the plant consists of the list of entities named *tasks* and *data flows*, annotated with computational and communication requirements, respectively. The structural description of communication consists of grouping tasks into *zones*. Tasks belonging to the same machine can be grouped into the same zone. The designer should also provide the Network Synthesizer with a catalog of network architectures (e.g., Ethernet, WiFi, CAN bus), mapped onto the concept of

Author version

Abstract channel. The environment description consists of instances of *Contiguity relationship* which ties two zones and an abstract channel. It can be used to model the impact of the environment between the two given zones on the behavior of the given network architecture (e.g., bandwidth reduction or cabling cost). Such information can be specified by performing a site survey of the shop floor.

Starting from this formalization, network synthesis consists of finding a suitable mapping of data flows onto abstract channels (i.e., network architectures) taken from the catalog which optimizes a given metric.

3.3 Design tool suite

As described in [16] and sketched in Figure 3, the optimization problem was formulated as a Mixed Integer Linear Programming (MILP) problem. The MILP formulation contains a set of *variables* that represent specific aspects of the solution and a set of *constraints*, that allow to check the valid assignments to the *variables*. Finally, the formulation contains an *objective function*, a mathematical expression defined using the model's *variables* that usually evaluates to a numerical value. The goal of the underlying solver is to minimize or maximize the value of the *objective function*, by assigning a value to the *variables* while making sure that the *constraints* still hold. An example of variable is:

$$h_{d,c,p} = \begin{cases} 1 & \text{if the data flow } d \text{ is placed in} \\ & \text{the } p\text{-th channel of type } c, \\ 0 & \text{otherwise.} \end{cases}$$

where $h_{d,c,p}$ is a boolean variable that, if *true*, associates a given data flow to a specific instance of a channel. An example of constraint is one that makes sure that the capacity of a given channel instance is higher than the sum of the throughput values of the contained data flows. In MILP formulation, constraints like this, as well as the objective function, are represented as linear combinations of integer variables. This formulation is written in Python and relies on the solver Gurobi [18].

Even if this approach could be sound from a scientific perspective, it is quite hard to be addressed by the plant designer. Therefore a graphical front-end tool was created as shown in Figure 4. The graphical front-end tool is a stand-alone application written in Python to be executed on different platforms. It follows a drag-and-drop approach to specify the input data for the network synthesis, namely, zones and tasks inside zones. Zones are represented by rectangles and can be nested, and tasks are represented by circles. It allows to draw data flows as arrows between tasks and contiguity relationships as links between zones. For each task, data flow and contiguity relationship, their attributes for the synthesis can be specified by using the second vertical panel on the left. Figure 4 also shows another window to provide the catalog of the network architectures that will be instantiated to host the data flows as a result of the network synthesis. The graphical front-end tool maintains a JSON description of all this information, that can be saved locally or uploaded to the back-end tool for network synthesis. The back-end tool is a Python server-side application which takes the JSON description and generates the problem description for the Gurobi solver. The back-end service is exposed as a REST webservice called by the graphical front-end tool. The optimization result is not provided synchronously in the REST response since optimization may take time to be performed by the solver; the back-end tool sends the textual description of the result to the user by email.

4 Data collection and management

Achieving communication with the physical plant and collecting real time data opens the door to smart manufacturing, as they allow manufacturers to better measure, understand, and optimize production and to leverage analytics to make decisions that are rooted in facts [19].

4.1 Data sources

In manufacturing, relevant data comes both from the operations management infrastructure (Figure 1) and from sensors installed on the production environment: the former provides information about the operation of the line (e.g., the production recipe being executed, status and configuration information about the equipment), while the latter allows to monitor the physical evolution of the machinery (e.g., power consumption, vibrations, movements) [14]. The correlation of these different sources of data provides the necessary knowledge about the evolution of the production line. Note that sensor installation (e.g., to monitor power consumption, vibration, temperature) is possible also on obsolete production lines with no digital support: sensors will indeed enable connection to the digital infrastructure, as will be shown later in this section.

The choice of sensors to be installed on the production line is crucial, as they are the real heart of a smart factory. Considering the heterogeneity of the quantities to be monitored, different sensors will be applied to data acquisition. Transversely to all monitored quantities, the choice must take into account the technological characteristics of the sensor in terms of connectivity and of integration with the enterprise software.

Data is then collected in a variety of ways: environmental sensors usually adopt IoT infrastructures, whereby equipment and product information can be collected directly from the operations management infrastructure, e.g., through dedicated servers and database technologies [14], [20]. Different sources imply different data formats and types, that must be integrated to allow efficient processing and intelligence extraction. Data formats involving a large number of bits may lead to reduce the sampling rate if the channel capacity is low, thus compromising the real-time effectiveness of data collection.

4.2 Data storage and processing

The large amount of collected data must be securely stored and effectively integrated. In this sense, *cloud computing* allows to effectively store large quantities of data in highly cost effective, energy efficient and flexible fashion. Additionally, the distributed nature of cloud storage allows a highly scalable and shareable storage [21]. This is critical as the stored data includes both real-time monitored data and historical data, that must be available to increase awareness

Author version

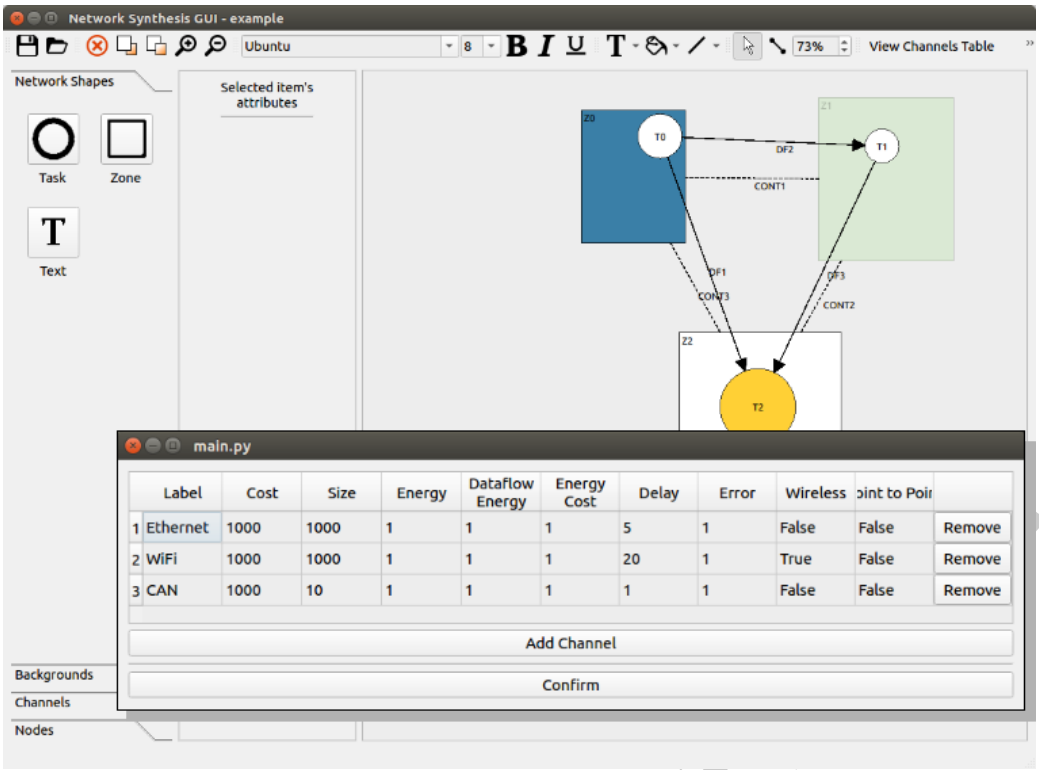

Fig. 4. Screenshot of the graphical front-end tool for network synthesis.

of the evolution of the equipment over time and to build prediction and control models [22].

Collected data must then be pre-processed to ease extraction of relevant knowledge: redundant, misleading, duplicated, and inconsistent information must be removed, and data reduction techniques must be applied to transform a massive amount of data into ordered, meaningful and clean information, useful for subsequent analysis [22], [23]. To improve the scalability of the proposed data architecture and meet real time constraints, such data pre-processing can be moved closer to data collection, e.g., in dedicated edge nodes that collect data from sensors, clean it, and then upload it to the centralized cloud infrastructure [24], [25].

After these steps, data is made available to services such as digital twins, that exploit the real-time line monitoring and data coming from the management infrastructure to improve the production process.

5 Energy Monitoring

Equipment energy consumption is a large portion of the total consumption of manufacturing ($\sim$ 75%), and thus should be tightly monitored and optimized to cope with serious situations such as rising energy price, global resource depletion and climate warming [26].

To achieve energy monitoring, the equipment of interest is enriched with sensory devices, that measure power or current demand over time. Energy sensor installation is straightforward, as energy sensors simply clip around single-phase or three-phase power cables (thus being applicable to any kind and any age of production machinery). Additionally, they provide built-in connectivity to dedicated gateways that transmit information collected from one or more sensors over a wireless network [27]. The monitored data can thus be easily collected and monitored in real time, and it can be transmitted over the network to make further analysis about the correlation of such power consumption w.r.t. equipment operation parameters [28], [29], [30].

The construction of the energy monitoring infrastructure requires to intervene both on the physical layer and on the virtual layer:

- identify relevant *machine parameters*, i.e., machine-related information that allows to identify the operation mode of the equipment (e.g., moving, cutting, idle) and the relative configuration (e.g., speed, acceleration) that is considered relevant from the perspective of power consumption;
- insertion of *power consumption sensors* to extract power measurements in real time during machine operation;
- construction of an *IoT collection and transmission infrastructure*, that allows fast collection and integration of data, through the adoption of edge technology or of cloud servers for data storage.

Once that real-time data is available, it is compared against *models of power consumption*, either based on historical data or on models built on equipment specifications. This comparison allows to detect any misalignment of the real equipment w.r.t. the expected behavior, as an effect of gradual degradation (e.g., due to wear or corrosion) and sudden disturbances [31]. In this perspective, energy monitoring becomes thus an important instrument to achieve *predictive maintenance*, i.e., localization of quality losses in machining, better prioritization of the maintenance schedule, avoiding unscheduled downtime and losses in product quality

Author ver

[32]. Additionally, the power consumption models allow to *optimize the production process*, by comparing the impact of different production scheduling settings or of different parameters of the same production process [31], [33].

A relevant role is thus performed by the chosen energy models, that identify relations between machine operation and the corresponding power consumption. Different models have been proposed in the literature, depending on the available data and on the goal of the analysis:

- *power state machines*, a power consumption model inherited by Electrical System Design that identifies the device typical operation states and associates each state with the corresponding power consumption [17], [34]. Power values are either derived from available documentation or extrapolated from available data sensed from the plant;
- *mathematical and statistical models* describing the manufacturing system as a stochastic dynamic system, characterized by analyzing on sensed data [33], [35];
- *learning-based approaches*, that exploit actual historical data obtained from the plant to train neural networks or learning algorithms, that capture nonlinear relationships between input parameters of the plant and output power consumption [36], [37], [38].

The kind of approach to be adopted strictly depends on the characteristics of the production line and of its data monitoring infrastructure:

- *availability of data sensors*: if data is collected at run time from the plant, it is possible to build data-aware models, with a higher level of fidelity w.r.t. plant operation; else, it is necessary to rely on available documentation, thus including an implicit glitch;
- *availability of information related to the equipment*: building a detailed correspondence between device operation and power consumption requires for the equipment to export relevant information about its operating parameters, like speed, acceleration, position, etc., without whom it becomes possible to reason only in terms of general idleness or activity. This kind of information is not available for obsolete production machinery, that does not have a data transmission capability to export its internal status. In this case, simpler models will be adopted for relating power consumption with macro-information related to equipment status (e.g., ON/OFF);
- *desired level of detail* w.r.t. the actual plant operation, i.e., whether it is necessary to have a high accuracy and adaptability w.r.t. the evolving conditions of the physical plant.

In this way, the power consumption model becomes a *digital twin of the production plant* [4], focusing only on one major aspect, i.e., energy consumption monitoring (Figure 5). The digital twin is enabled by a sound data-collection infrastructure, that collects real-time data along with the historical data to track its energy consumption, and by the power estimation models, that are used for parameter optimization, scheduling, and equipment upgrading and maintenance. The digital twin can also be extended with error correction, compensation, and feedback control, whose

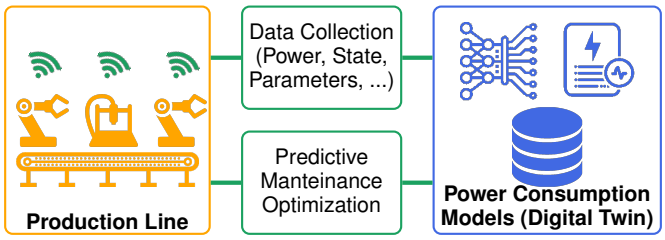

Fig. 5. Power monitoring digital twin.

operation is regulated by a *maintenance decision support module* that may try to restore normal operation condition, stop the production or raise an alert to the user [39]. Different techniques can be used to correct and compensate the errors, as they are mainly dependent on the types of machines and processes, on the severity of the alerts and on the requirements of the specific application.

6 Case Study: Industrial Computer Engineering (ICE) Laboratory

The reference production line for this work is an Industry 4.0 research facility called *Industrial Computer Engineering (ICE) Laboratory (ICELab)*. The structure of the ICELab includes a fully fledged production line (Figure 6) consisting of:

- a vertical warehouse storing materials and products;
- two collaborative Autonomous Mobile Robots (AMRs), i.e., two Robotnik RB-Kairos AMR [40] equipped with anthropomorphic manipulators, that can load and unload materials from the warehouse to a dedicated point on the conveyor belt, actively cooperate with an operator, and perform advanced and cooperative handling tasks;
- a quality check station;
- a collaborative robotic assembly station, comprising two lightweight collaborative robots: an ABB Yumi [41], and a Kuka Lightweight Robot (LR) [42];
- two 3D printers: a stereo-lithography mono material 3D printer and a multi-material polijet 3D printer;
- a milling machine;
- an electronic automatic tester;
- a complex transportation system composed of a main conveyor belt that spans across the entire laboratory in a ring configuration, and an unloading conveyor bay for each machine and for the AMRs.

This laboratory allows to represent the most modern automation technologies adopted in production processes. This paper focuses on a subset of the equipment, specifically, the parts labeled in Figure 7.

The considered production process is divided into four phases. First, a set of LEGO-like blocks is transported from the vertical warehouse to the assembly station by means of the mobile robots and the transport belt. Second, the pieces are assembled by the two cooperating robotic arms, i.e., Kuka and ABB. Third, the assembled product is transported to the quality check station by means of the same transportation belt used before. At the quality check station, a robotic arm rotates the assembled piece and exposes all critical parts to the cameras: if the desired quality standards are achieved,

Author version

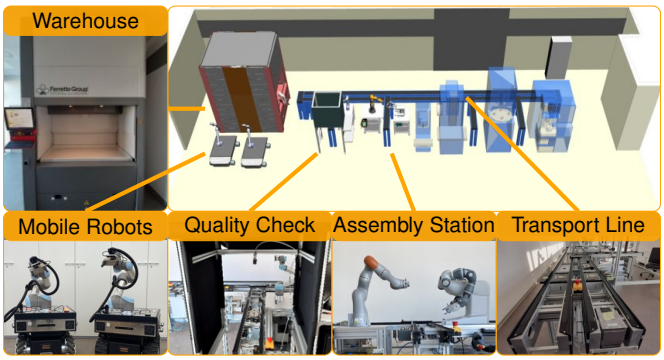

Fig. 6. ICELab production line stations.

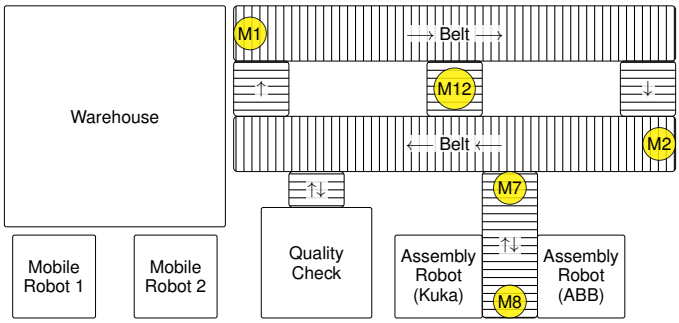

Fig. 7. ICELab production line schematical view.

the piece is put back on the belt and transported to the vertical warehouse for storage.

All the active entities of the production line are instrumented to provide the IT office with real-time data by using OPC UA protocol with end-to-end encryption [43]. Each station implements an OPC UA server that exposes relevant parameters of the equipment. This OPC UA server is implemented inside the PLC of the equipment.

6.1 Network design

This section provides a small but exhaustive example of network synthesis for the plant described in Figure 6. Figure 8 shows the problem described by using the proposed graphical tool in which network entities can be superimposed on a picture of the real environment. Tasks are denoted by circles whose radius is proportional to their computational complexity. Zones are denoted by rectangles

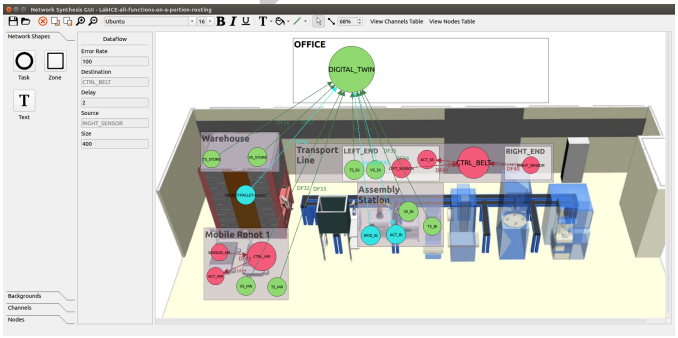

Fig. 8. Screenshot of the tool for the network synthesis of a portion of the ICELab.

```
<ZONE_LABEL>
        0       Environment
        1       MobileRobot1
        2       OFFICE
        3       Transport Line
        4       Assembly Station
        5       LEFT_END
        6       RIGHT_END
        7       Warehouse
</ZONE_LABEL>
<ZONE>
        1 0
        2 0
        3 0
        4 0
        5 3
        6 3
        7 0
</ZONE>
<CONTIGUITY>
        7       2       0       1       10
        7       2       1       0.5     1
        7       2       2       1       10
        1       2       0       1       10
        1       2       2       1       10
        1       2       1       0.5     1
        3       2       0       1       10
        3       2       2       1       10
        3       2       1       0.5     1
        4       2       0       1       10
        4       2       2       1       10
        4       2       1       0.5     1
        5       2       0       1       10
        5       2       2       1       10
        5       2       1       0.5     1
        6       2       0       1       10
        6       2       2       1       10
        6       2       1       0.5     1
</CONTIGUITY>
<TASK>
        SENSOR_MR       1       1       0
        ACT_MR  1       1       0
        LEFT_SENSOR     1       5       0
        CTRL_BELT       5       2       0
        RIGHT_SENSOR    1       6       0
        TS_MR   1       1       1
        TS_S6   1       5       0
        DIGITAL_TWIN    10      2       0
        RFID_Bi 1       4       0
        ACT_Bi  1       4       0
        VS_S6   1       5       0
        OBJ-PALLET-ASSOC        1       7       0
        VS_MR   1       1       1
        CTRL_MR 5       1       0
        TS_Bi   1       4       0
        VS_Bi   1       4       0
        ACT_S6  1       5       0
        VS_STORE        1       7       0
        TS_STORE        1       7       0
</TASK>
<DATAFLOW>
        DF32    VS_MR   DIGITAL_TWIN    50 50 100
        DF33    TS_MR   DIGITAL_TWIN    50 50 100
        DF34    TS_Bi   DIGITAL_TWIN    50 50 100
        DF35    VS_Bi   DIGITAL_TWIN    50 50 100
        DF36    TS_STORE        DIGITAL_TWIN    50 50 100
        DF37    VS_STORE        DIGITAL_TWIN    50 50 100
        DF40    RIGHT_SENSOR    CTRL_BELT       1 2 100
        DF41    LEFT_SENSOR     CTRL_BELT       1 2 100
        DF42    CTRL_BELT       ACT_S6  1 2 100
        DF43    OBJ-PALLET-ASSOC DIGITAL_TWIN 10 10 100
        DF44    RFID_Bi DIGITAL_TWIN    10 10 100
        DF45    DIGITAL_TWIN    ACT_Bi  10 10 100
        DF78    SENSOR_MR       CTRL_MR 1 2 100
        DF79    CTRL_MR ACT_MR  1 2 100
        DF16    VS_S6   DIGITAL_TWIN    50 50 100
        DF17    TS_S6   DIGITAL_TWIN    50 50 100
</DATAFLOW>
```

Listing 1. Network synthesis problem description for the ICELab.

Author version

to group tasks belonging to the same item, *i.e.*, warehouse, office, mobile robot and belt (as well as parts of it). As shown in Figure 3, the front-end tool also allows to specify the catalog of network architectures as depicted in Figure 4 which in this examples are Ethernet, CAN and WiFi. As reported in Figure 4, each network architecture has also a cost associated.

Listing 1 shows the input file generated by the graphical tool for the solver. It consists of several sections for the different plant aspects reported in Figure 3. Highlighted text denotes examples of pieces of information which are described in the following. The first three sections contain the environment description of Figure 3. The first section lists the zones into which the plant has been decomposed. The second section describes their hierarchy. For instance, `LEFT_END` and `RIGHT_END` are two sub-zones of `Transport Line`. The section on contiguity relationships describes how the various channel types are affected by crossing zone boundaries. For instance, the capacity of Ethernet and CAN is not affected (attenuation= 1) while WiFi's capacity is halved. Regarding the cabling cost, Ethernet and CAN are more expensive than WiFi. The plant structural description is contained in the task section where task characteristics are reported, e.g., the computational complexity (first number in each row) and mobility (last number of each row). For instance, task `VS_MR` and `TS_MR` (green circles inside the mobile robot in Figure 8) are mobile with respect to the digital twin task in the office. Finally, the plant functional description is contained in the data flow section which reports the quality of service required by the various data flows, e.g., throughput and maximum tolerated delay. Monitoring data flows (green arrows in Figure 8) require high throughput (about 50 kb/s) without tight constraints on delay ($\leq$ 50 ms). Control data flows (red arrows) convey control loop information with low throughput (about 1 kb/s) but a strict constraint on delay ($\leq$ 2 ms). Coordination data flows (cyan arrows) require medium throughput (about 10 kb/s) and a moderate constraint on delay ($\leq$ 10 ms).

In this use case, the MILP formulation consists in minimizing the sum of costs of the allocated instances of the different network architectures subject to constraints on fulfilling data flow requirements on quality of service (e.g., data flow throughput should be compliant with channel capacity). The formulation takes into account also the quality loss and the cost increase due to zone crossing.

Listing 2 shows the resulting output file sent by the back-end tool to the user. The most relevant result is the allocation of data flows to the network architectures provided as catalog to the optimizer. Most of the data flows have been assigned to Ethernet, which provides high capacity without strong guarantees on delay. Data flows related to control loops have been assigned to CAN bus to fulfill delay constraint. Data flows involving mobile tasks have been assigned to WiFi to satisfy mobility constraint. The tool also provides statistics on relevant metrics for the infrastructure such as cost, energy consumption, sum of delay values and error rate values.

```
List of activated channels:
    Use    3 channels of type Ethernet
    Use    1 channels of type WiFi
    Use    2 channels of type CAN
Data-Flows allocation:
    Dataflow DF32 inside channel WiFi.1
    Dataflow DF33 inside channel WiFi.1
    Dataflow DF34 inside channel Ethernet.3
    Dataflow DF35 inside channel Ethernet.3
    Dataflow DF36 inside channel Ethernet.1
    Dataflow DF37 inside channel Ethernet.1
    Dataflow DF38 inside channel Ethernet.2
    Dataflow DF39 inside channel Ethernet.2
    Dataflow DF40 inside channel CAN.1
    Dataflow DF41 inside channel CAN.2
    Dataflow DF42 inside channel CAN.2
    Dataflow DF43 inside channel Ethernet.1
    Dataflow DF44 inside channel Ethernet.3
    Dataflow DF45 inside channel Ethernet.3
Economic Cost       : 9031
    2442 (Nodes) + 1201 (Wireless) + 5388 (Channels)
Energy Consumption :  581
    42 (Nodes) + 201 (Wireless) + 338 (Cable)
Total Delay         :  128
    80 (Wireless) + 48 (Cable)
Total Error         :   16
    4 (Wireless) + 12 (Cable)
Elapsed Time : 1.20 s
    File parsing          : 0.02 s
    Structure creation    : 0.12 s
    Constraints definition : 0.63 s
    Optimization          : 0.42 s
```

Listing 2. Network synthesis output for the ICELab.

6.2 Online monitoring and data collection

The ICE Laboratory is innovative because it combines the knowledge of many industrial partners, enabling the sharing of technologies and new ideas in the context of *Industry 4.0*. The combination of IIoT sensors and of a complete infrastructure for data collection allows online monitoring of the entire production process [14], [44]. The data collection architecture (shown in Figure 9) is built based on the network infrastructure described in Section 6.1. The IIoT sensors placed on the production line include sensors of vibration, temperature, position, and power. All these sensors are connected to a BOX-IO gateway [45]. The prerequisite to apply this data collection architecture is a plant equipped with OPC UA servers providing equipment status by native or custom OPC UA servers and environment data (IoT and Industrial IoT).

An OPC UA server provides secure access to industrial automation data using OPC UA information models, that specify how data is organized, stored, and collected. The proposed data collection architecture (Figure 9) is based on the OPC UA communication protocol relies on a Kubernetes infrastructure [46]. The goals of this infrastructure are:

- Monitoring through OPC UA servers;
- Data logging;
- Data analytics and filtering;
- Data upload to different cloud providers;
- Provide a unique interface to access data;
- Secure connections.

The data sources are the IoT sensors and the equipment (highlighted in yellow). Different OPC UA clients read data from the sources (OPC UA server application); each OPC UA client is reconfigurable because it allows to retrieve data from an OPC UA server through the OPC UA server URI, and it can subscribe to each variable. For each variable of the OPC UA information model, it is possible to set sampling

Author version

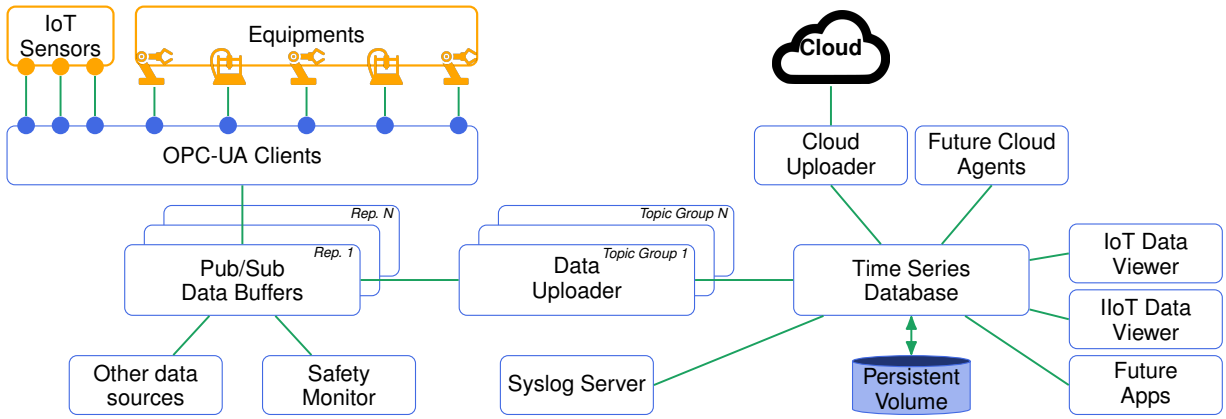

Fig. 9. Data collection architecture of the ICELab.

interval, datatype and unit, and other custom static fields. All data retrieved through the OPC UA clients is sent to a publish/subscribe data buffer, i.e., a large data stream handling data partitioned in different topics (our configuration is based on the Apache Kafka open-source application). This application is subdivided into multiple instances to guarantee fault tolerance and high performance. Moreover, the data buffer is extensible, as it allows to easily add data producers/consumers. Data from the data buffer allows to safely monitor parts of the production line before storing them in a time-series database through data uploader nodes. Each node of the data uploader sends chunks of data to the database (our configuration is based on the Telegraf, a plugin-driven agent that collects, processes, aggregates, and writes metrics into an InfluxDB time-series database).

The time-series database chosen for the ICE Laboratory is a NoSQL database, InfluxDB [47]. InfluxDB is an open-source time-series database that provides real-time visibility into stacks, sensors, and automation data for monitoring metrics and events. Data is stored in a persistent volume and optionally can filter incoming values and perform data aggregation. Data in the database is organized in buckets: each data saved in the time-series database is associated with a time-stamp, i.e., the time instant in which the data is read from the production line. The best configuration of the database is to subdivide the buckets for different purposes, *e.g.*, raw data, clean data, and cloud data. The database has multiple instances managed by the Kubernetes infrastructure. Data is analyzed through a filter, cleaned, and manipulated, and the results of such manipulations are usually stored in a different bucket inside the database. The data collection architecture's final steps consist of uploading part of the data stored in the time-series database to the cloud providers: only the necessary data for other analyses are uploaded to the cloud. The quantity of data on the cloud depends on the company policy. Moreover, through custom applications and viewers, it is possible to visualize the query results on the data stored in the database (our implementation of custom dashboards is based on the Grafana Dashboards). All the nodes of the data collection architecture are bundled in a container, an application abstraction that packages code and dependencies together (application, binaries, and libraries). In a physical plant, a system could handle thousands of containers: for that reason, clusters are managed by Kubernetes, a container orchestration tool from CNCF Foundation and automatically manages the containers without an IT manager's control.

6.3 Energy monitoring

This Section focuses on constructing a digital twin for power consumption of the portion of the production line that was equipped with sensors at the time of this experimental analysis: the quality check station, the robotic assembly station and the transportation system.

6.3.1 *Sensing infrastructure*

Accessing the power consumption of the devices of interest required the installation of sensors for monitoring the evolution of the AC power quantities [48]: active power (W), phase angle (degrees, i.e., the phase between the voltage and current sinusoidal curves), power factor (used to estimate the amount of dissipated power), and current (A).

The ABB Yumi and the Kuka LR are single-phase AC devices, and the quality station is a DC station converted to single-phase AC through an inverter. All such devices are thus monitored through single-phase Eastron SDM 230 power meters [49]. The transport line is made of 15 three-phase AC motors. To ease the monitoring of power consumption, all motors are monitored through a single three-phase Eastron SDM630 meter, exporting the accumulated AC power characteristics of all phases [50]. This sensing strategy allows to consider the transport line as a single-phase AC load, thus avoiding the burden of separating the power models of each single phase. The power sensors allow to retrieve the sensed data with a polling rate of 2 seconds in our Modbus configuration.

6.3.2 *Data processing*

All real-time data, including measurements made by the sensors and information about the production recipe, are made available to the power consumption digital twin through the MES interface (as explained in Section 6.2). Information about the device operating mode, exported from the PLC, consists of the macro-state of the device, i.e., whether it is idle or active, and in the latter case the action performed (e.g., in case of the robots, pick or place). More

Author version

detailed parameters (e.g., operating speed and acceleration) were not available at the time of the experimental analysis.

To model power consumption of the devices, we decided to adopt approaches that exploit both historical and real time data. This required a collection campaign, to store data relative to a number of production cycles, and a cleaning phase of such data, to extract timestamps, remove duplicated data and identify unavailable samples.

6.3.3 Models of power consumption

The goal of the digital twin is to predict active power consumption of each device, given information about the production recipe and the device operating mode received at run time: if the corresponding sensed value is different from the predicted one (given a certain tolerance threshold), an alert is notified. Data pre-processing has been applied on historical data of each device in order to determine the correlation of its settings with the corresponding power consumption of the machine. The developed model will strictly depend on the kind of monitored device and of its available information.

An analysis of the correlation ρ between the quantities monitored by power sensors proved that all of them are highly correlated with active power: e.g., correlation ρ w.r.t. active power for the Kuka robot is 0.76 for current, 0.89 for power factor and 0.84 for phase angle[1]. Thus, modeling power consumption is a good bias also for the other quantities. This allows reducing the amount of data that must be transferred from the sensors to the digital twin and the complexity of the models to be developed.

Quality of prediction w.r.t. the sensed power consumption is measured for all models in terms of *mean squared error (MSE)*, defined as: $MSE = \frac{1}{n}\sum_{t=1}^{n}(y_t - \hat{y}_t)^2$, where y is the actual power consumption received from the sensors at time t, $\hat{y}$ is the predicted power consumption, and n is the number of samples under analysis.

6.3.4 Power model of the quality check station

The QC station is always active, as the work station and the camera are always on. As a result, its power consumption is quite stable, and it is normally distributed (mean 110.69W) with a small standard deviation ($\sigma = 1.29W$).

To predict its power consumption, we thus adopted a mathematical model, that allows to identify any behavior not included in the range of the expected ones. The model relies a *moving average algorithm*, where the predicted value $\hat{y}$ is the mean of the previous n data points. The choice of the size n of the sliding window depends on the desired smoothing: an increased value of n enhances the smoothing at a cost of accuracy. The estimated power consumption at time t is therefore computed as:

$$\hat{y}[t] = \frac{y[t-1] + y[t-2] + ... + y[t-n]}{n} \quad (1)$$

where y is the measured power consumption in the previous time steps at time $t-1 ... t-n$, and $\hat{y}$ is the predicted power consumption. This model assumes that consecutive power demand samples have a similar consumption; as such, most of the estimate depends on the latest measurement with a defined sliding window ($n = 15$).

Historical data are then used to derive the typical *standard deviation* of the distribution σ. The resulting power model thus exploits both the moving average and the standard deviation: at any time instant t, moving average is used to predict power consumption $\hat{y}[t]$: if the sensed power consumption falls in the range $\hat{y}[t] \pm 3 \cdot \sigma$ then the sensed value is considered coherent with the estimation, else an alarm is set. This mechanism is exemplified in Figure 10, where the red line is $\hat{y}$, the shaded area highlights the allowed prediction interval over time, and the blue line are the measured values y. The achieved MSE is very low ($1.23W$), thus achieving a very good prediction accuracy.

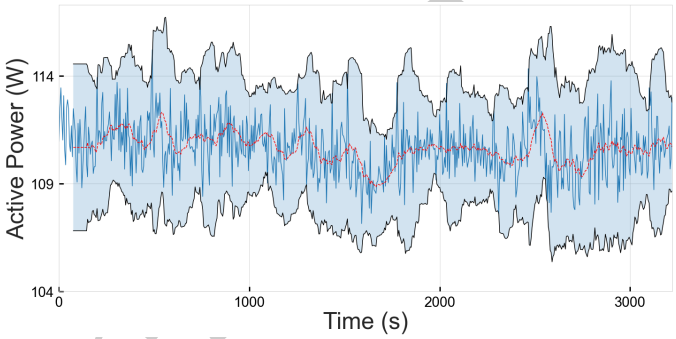

Fig. 10. Power model fo the quality check station: active power (blue line), estimated power consumption $\hat{y}[t]$ (Equation 1), and allowed prediction interval $\hat{y}[t] \pm 3 \cdot \sigma$ (light blue area).

6.3.5 Power models of the robot assembly station

The ABB Yumi and the Kuka LR robot arms are used to pick and place LEGO-like blocks from the pallet, so to assemble them. Their task is thus periodic: when the pallet enters the bay, each arm picks one block from the pallet, puts it in the correct position and then goes back to idle. The resulting power consumption curves have peaks in correspondence to each pick or place phase (Figures 11 and 12).

For both robots, the data infrastructure exports the status and the operating mode of the robot (idle, active and pick, active and place), plus the measured power consumption. The available historical data is used to construct a model of power consumption based on a *neural network* that creates a relationship between input parameters (i.e., the $Idle$, $Pick$, and $Place$ operating modes of the robot arms) and the output O, i.e., the estimated power consumption. The neural network has two hidden layers with 32 and 64 neurons respectively, and one output layer that predicts the power consumption with 300 epochs used to update the weights and the bias of the parameters by optimizing the error between the training sample with the predicted power. For each robot, the model has been trained using 80% of the available historical data by using the Levenberg–Marquardt back-propagation training algorithm (*training set*) and verified using the remaining 20% (*test set*). The estimated MSE is $13.21W$ for the Kuka LR and $2.11W$ for the ABB Yumi: the model thus proves to be accurate in the estimation of the power consumption of the robots. Figures 11 and 12 show a snapshot of the evolution of the models, by reporting

1. A value closed to 1 indicates that the variables tend to increase together, while a value closed to 0 indicates that they are independent.

Author Version

measured vs. predicted power consumption (top, solid and dashed, respectively). The power model generates an alarm if the measured power consumption does not fall in the range $\hat{y} \pm MSE$, thus allowing a tolerance threshold to take into account the possible prediction error.

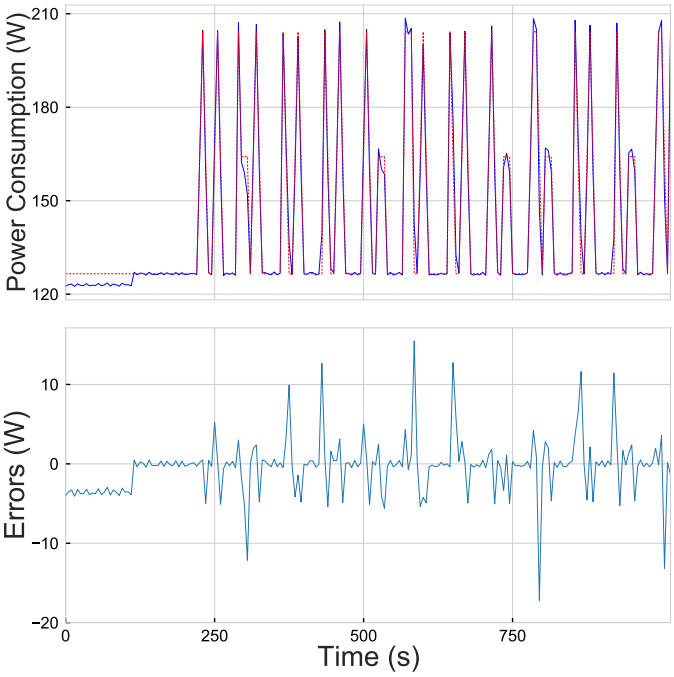

Fig. 11. Kuka measured (solid blue) and estimated (dashed red) power consumption (top) and error of the prediction model (bottom).

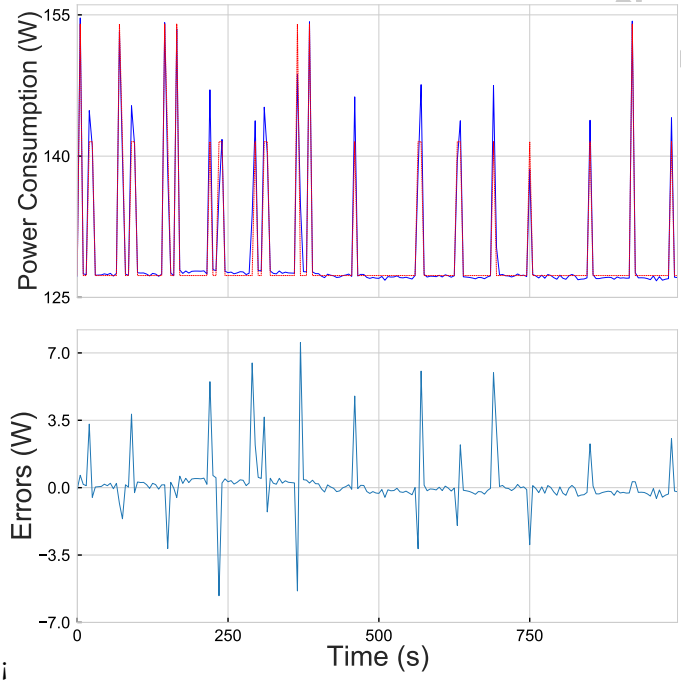

Fig. 12. ABB measured (solid blue) and estimated (dashed red) power consumption (top) and error of the prediction model (bottom).

6.3.6 Power model of the transport line

Transport line movement is managed by 15 motors, of which two control the main belts, while the others allow movement from the main belts to the bays (Figure 7 shows only the motors used in the following of this section). Sensor data monitors overall power consumption of the transport line, thus aggregating not only the three AC phases but also the power consumption demand of all motors. To derive an accurate model of power consumption it thus becomes crucial to know which motors are active at any time. For this experiments we activated only the belts of interest for the monitored devices, i.e., motors $M1$ and $M2$ that control the main belts, plus motors $M12$, $M7$ and $M8$ that move the belt connecting the bay with the robot assembly station. Reducing the number of motors active at any time allows indeed to reduce the noise and to better control power consumption, given that we get an aggregated power consumption curve for all motors.

Data received from the MES allows to know what motor is active at any time (green labels). We thus analyzed historical data of power consumption jointly with information about active motors over time, to separate the contribution of each motor to the overall power consumption. Then, we built a *moving average model* similar to the one in Section 6.3.4, that takes into account also which motors are active at any instant. Motors $M1$ and $M2$ are always active, thus the base predicted moving average is $\hat{y} = \mu_{M1} + \mu_{M2}$. When one motor Mi becomes active, the current overall moving average is increased by the mean power contribution of the motor μ_{Mi}, so to build the power prediction by considering also its contribution to power consumption; vice versa, when a motor Mi becomes idle, the mean of its contribution to power consumption μ_{Mi} is subtracted from the current overall moving average, to make a prediction that considers this motor as turned off.

To exemplify this process, Figure 13 shows the evolution of sensed power consumption (blue line), moving average $\hat{y}$ (red) and allowed prediction interval when turning on and off line motors alternatively (green labels). At the beginning, only motors $M1$ and $M2$ are active, thus $\hat{y} = \mu_{M1} + \mu_{M2}$. Then, motors $M7$ and $M8$ are activated to move the pallet to the robot assembly bay: $\hat{y}$ is increased with the estimation μ_{M7}, and then of μ_{M8}. When the motors are turned off, the estimation goes back to $\hat{y} = \mu_{M1}+\mu_{M2}$. Then, the prediction follows the subsequent activation of the motors. This model allowed to reach a MSE of $5.58W$, thus achieving a very good prediction accuracy.

6.3.7 Adoption of the digital twin for anomaly detection

The built models for power consumption concur in building a digital twin of the production line. The digital twin runs in parallel with line operation: it is fed with commands and sensed data, it estimates the expected power consumption of the single devices based on the developed models, and it compares the prediction $\hat{y}$ w.r.t. the sensed data. The data processing infrastructure can be used to build services, like graphical rendering of the real-time power consumption and of the predicted consumption. Additionally, the digital twin allows live detection of unexpected behaviors, identified as a misalignment between the sensed and the predicted power consumption: in this case, an alert is notified to the user, that can monitor the operation of the device of interest to identify any possible wear and tear effect, or to discard the alert as an anomalous sensed data.

Version

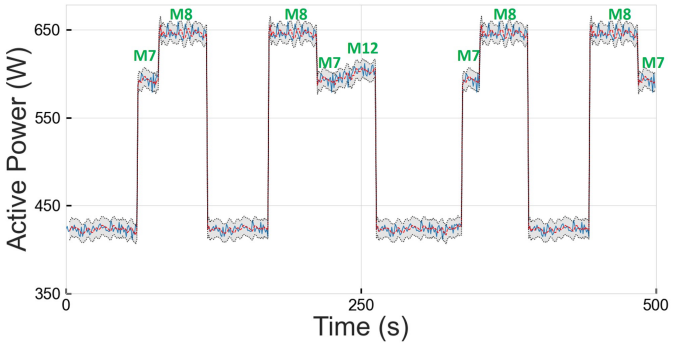

Fig. 13. Transport line Power Consumption (blue line) and application of the prediction model: the red line is the estimated power consumption $\hat{y}[t]$, while the light blue area represents the allowed prediction interval. Motors $M1$ and $M2$ are always active; green labels indicate the activation of additional motors.

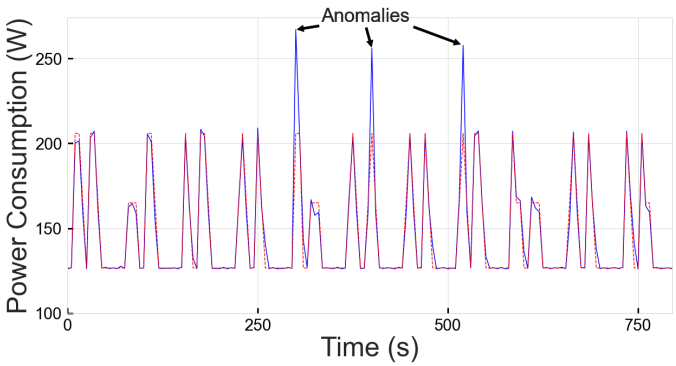

Fig. 14. Example of adoption of the digital twin of power consumption for anomaly detection: the unexpected power peaks of the Kuka robot are detected and used to fire an alert to the user and to the production line.

Figure 14 shows an example of this for the Kuka robot arm. The Figure is similar to Figure 11, in that it reports the real time power consumption of the robot arm (solid blue line) and the corresponding prediction of the model presented in Section 6.3.5 (dashed red line). In this case, the model detects three anomalous behaviors of the robot, i.e. three peaks that consume up to 30% more than the predicted value: these anomalies are detected by the power model, that sends an alert both to the plant and to the user interface. This allows the user to analyze robot behavior and identify any disturbance in data monitoring infrastructure or any malfunctioning of the robot arm. The subsequent step is the application of monitoring and maintenance actions to prevent further damage of the equipment [39], like compensation (e.g., slow down of Kuka operation to mitigate stress), raising an alert (either stopping line operation to restore normal operation, or continuing it to ensure effective production), or giving a feedback to the user, with the measured values and the Kuka operating conditions so to allow an informed maintenance decision. In our scenario, an alert is risen but line operation goes on, leaving further investigation to the user. As future work, the production line will be extended with control devices and a maintenance management system, that will apply different policies depending on user settings and on the level of severity of the risen alarm.

7 Conclusions

Smart manufacturing nowadays is mandatory to maintain competitiveness. This article has proposed techniques to support the digital transformation of a production line and the connection of these techniques to the digital twin, to show how it is possible to introduce digital content in a novel or existing production line, so to take full advantage of the Industry 4.0 methodologies. First, the *connecting network* of the production line has been built by using a network synthesis tool starting from the communication flows inside the plant. Second, the techniques to apply *online monitoring* based on standard and IIoT sensors and to *collect the data* have been presented. Finally, we exemplified how the digital data sensing and collection infrastructure can be used to monitor plant operation through the construction of a digital twin for energy monitoring, that highlights deviations with respect to the expected energy consumption models to witnessing aging and faults of machines. The presented concepts have been exemplified in the Industrial Computer Engineering laboratory of the University of Verona. As future work, we will work on the application of the edge computing paradigm to improve data management, and on the exploitation of the digital content to optimize and improve production effectiveness, e.g., by focusing on the impact of production recipes on communication and on energy consumption.

References

[1] P. F. Cunha and P. G. Maropoulos, *Digital Enterprise Technology - Perspectives and Future Challenges*. Springer, 2007.
[2] J. Morgan, M. Halton, Y. Qiao, and J. G. Breslin, "Industry 4.0 smart reconfigurable manufacturing machines," *Journal of Manufacturing Systems*, vol. 59, pp. 481–506, 2021.
[3] J. Schlechtendahl, M. Keinert, F. Kretschmer, A. Lechler, and A. Verl, "Making existing production systems industry 4.0-ready," *Production Engineering*, vol. 9, p. 143–148, 2015.
[4] F. Tao and M. Zhang, "Digital twin shop-floor: A new shop-floor paradigm towards smart manufacturing," *IEEE Access*, vol. 5, pp. 20 418–20 427, 2017.
[5] G. Tristo, G. Bissacco, A. Lebar, and J. Valentinčič, "Real time power consumption monitoring for energy efficiency analysis in micro EDM milling," *International Journal of Advanced Manufacturing Technology*, vol. 78, p. 1511–1521, 2015.
[6] D. Etz, T. Frühwirth, and W. Kastner, "Flexible safety systems for smart manufacturing," in *IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, vol. 1, 2020, pp. 1123–1126.
[7] A. Eckhardt, S. Müller, and L. Leurs, "An evaluation of the applicability of opc ua publish subscribe on factory automation use cases," in *IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, vol. 1, 2018, pp. 1071–1074.
[8] M. Sjarov, T. Lechler, J. Fuchs, M. Brossog, A. Selmaier, F. Faltus, T. Donhauser, and J. Franke, "The digital twin concept in industry – a review and systematization," in *IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, vol. 1, 2020, pp. 1789–1796.
[9] K. Xia, C. Sacco, M. Kirkpatrick, C. Saidy, L. Nguyen, A. Kircaliali, and R. Harik, "A digital twin to train deep reinforcement learning agent for smart manufacturing plants: Environment, interfaces and intelligence," *Journal of Manufacturing Systems*, vol. 58, pp. 210–230, 2021.
[10] F. Tao, M. Zhang, and A. Nee, Eds., *Digital Twin Driven Smart Manufacturing*. Academic Press, 2019.
[11] F. Biesinger and M. Weyrich, "The facets of digital twins in production and the automotive industry," in *International Conference on Mechatronics Technology (ICMT)*, 2019, pp. 1–6.

Author version

[12] Y. Gao, H. Lv, Y. Hou, J. Liu, and W. Xu, "Real-time modeling and simulation method of digital twin production line," in *IEEE International Information Technology and Artificial Intelligence Conference (ITAIC)*, 2019, pp. 1639–1642.
[13] Y. Ma, H. Zhou, H. He, G. Jiao, and S. Wei, "A digital twin-based approach for quality control and optimization of complex product assembly," in *International Conference on Artificial Intelligence and Advanced Manufacturing (AIAM)*, 2019, pp. 762–767.
[14] F. Tao, Q. Qi, A. Liu, and A. Kusiak, "Data-driven smart manufacturing," *Journal of Manufacturing Systems*, vol. 48, pp. 157–169, Jul. 2018.
[15] T. Lins, R. Augusto Rabelo Oliveira, L. H. A. Correia, and J. Sa Silva, "Industry 4.0 retrofitting," in *Proc. of Symposium on Computing Systems Engineering (SBESC)*, 2018, pp. 8–15.
[16] E. Fraccaroli, F. Stefanni, R. Rizzi, D. Quaglia, and F. Fummi, "Network synthesis for distributed embedded systems," *IEEE Transactions on Computers*, vol. 67, no. 9, pp. 1315–1330, Sep. 2018.
[17] K. Alamin, N. Dall'Ora, E. Fraccaroli, S. Vinco, D. Quaglia, and M. Poncino, "Digital twin extension with extra-functional properties," in *Design Automation Test in Europe Conference (DATE)*, 2021, pp. 1–6.
[18] "Gurobi optimizer," https://www.gurobi.com/products/gurobi-optimizer/, Gurobi Optimization, LLC, 2021.
[19] R. Rosen, G. von Wichert, G. Lo, and K. D. Bettenhausen, "About the importance of autonomy and digital twins for the future of manufacturing," *IFAC Symposium on Information Control Problems in Manufacturing*, vol. 48, no. 3, pp. 567–572, 2015.
[20] Y. Zhang, G. Zhang, J. Wang, S. Sun, S. Si, and T. Yang, "Real-time information capturing and integration framework of the internet of manufacturing things," *International Journal of Computer Integrated Manufacturing*, vol. 28, p. 811–22, 2015.
[21] D. Agrawal, A. E. Abbadi, S. Antony, and S. Das, "Data management challenges in cloud computing infrastructures," in *International workshop on databases in networked information systems*, 2010, p. 1–10.
[22] W. Ji and L. Wang, "Big data analytics based fault prediction for shop floor scheduling," *Journal of Manufacturing Systems*, vol. 43, pp. 187–194, 2017.
[23] E. Begoli and J. Horey, "Design principles for effective knowledge discovery from big data," in *Software Architecture and European Conference on Software Architecture*, 2012, p. 215–8.
[24] L. U. Khan, I.Yaqoob, N. H. Tran, S. M. A. Kazmi, T. N. Dang, and C. S. Hong, "Edge-computing-enabled smart cities: A comprehensive survey," *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 10 200–10 232, 2020.
[25] Q. Qi, D. Zhao, T. Liao, and F. Tao, "Modeling of cyber-physical systems and digital twin based on edge computing, fog computing and cloud computing towards smart manufacturing," in *ASME International Manufacturing Science and Engineering Conference*, 2018.
[26] L. Zhou, J. Li, F. Li, Q. Meng, J. Li, and X. Xu, "Energy consumption model and energy efficiency of machine tools: a comprehensive literature review," *Journal of Cleaner Production*, vol. 112, pp. 3721–3734, 2016.
[27] "Powertag wireless energy sensor," https://www.se.com/ww/en/product-range/63626-powertag, Schneider Electric, Tech. Rep., 2021.
[28] G. Tristo, G. Bissacco, A. Lebar, and J. Valentincic, "Real time power consumption monitoring for energy efficiency analysis in micro EDM milling," *International Journal of Advanced Manufacturing Technology*, vol. 78, p. 1511–1521, 2015.
[29] G. May, I. Barletta, B. Stahl, and M. Taisch, "Energy management in production: A novel method to develop key performance indicators for improving energy efficiency," *Applied Energy*, vol. 149, pp. 46–61, 2015.
[30] F. Liu, J. Xie, and S. Liu, "A method for predicting the energy consumption of the main driving system of a machine tool in a machining process," *Journal of Cleaner Production*, vol. 105, pp. 171–177, 2015.
[31] M. Zhang, Y. Zuo, and F. Tao, "Equipment energy consumption management in digital twin shop-floor: A framework and potential applications," in *International Conference on Networking, Sensing and Control (ICNSC)*, 2018, pp. 1–5.
[32] S. Emec, J. Kruger, and G. Seliger, "Online fault-monitoring in machine tools based on energy consumption analysis and non-invasive data acquisition for improved resource-efficiency," *Procedia CIRP*, vol. 40, pp. 236–243, 2016.
[33] J. Zou, Q. Chang, J. Arinez, and G. Xiao, "Data-driven modeling and real-time distributed control for energy efficient manufacturing systems," *Energy*, vol. 127, pp. 247–257, 2017.
[34] S. Jia, R. Tang, J. Lv, Q. Yuan, and T. Peng, "Energy consumption modeling of machining transient states based on finite state machine," *International Journal of Advanced Manufacturing Technology*, vol. 88, p. 2305–2320, 2017.
[35] I. Ullah, R. Ahmad, and D. Kim, "A prediction mechanism of energy consumption in residential buildings using hidden markov model," *Energies*, vol. 11, no. 2, 2018.
[36] N. Bhusal, R. M. Shukla, M. Gautam, M. Benidris, and S. Sengupta, "Deep ensemble learning-based approach to real-time power system state estimation," *International Journal of Electrical Power & Energy Systems*, vol. 129, p. 106806, 2021.
[37] Şükrü Özşahin and H. Singer, "Development of an artificial neural network model to minimize power consumption in the milling of heat-treated and untreated wood," Kastamonu University, Tech. Rep., 2019.
[38] A. Bakurova, O. Yuskiv, D. S. A. Riabenko, and E. Tereschenko, "Neural network forecasting of energy consumption of a metallurgical enterprise," *Innovative Technologies and Scientific Solutions for Industries*, vol. 1, 2021.
[39] Z. Li, Y. Wang, and K.-S. Wang, "Intelligent predictive maintenance for fault diagnosis and prognosis in machine centers: Industry 4.0 scenario," *Advances in Manufacturing*, vol. 5, p. 377–387, 2017.
[40] "RB-Kairos mobile manipulator," Robotnik, 2020, https://www.robotnik.eu/logistics/portfolio/rb-kairos/.
[41] "YuMi - IRB 14000 collaborative robot," ABB, 2020, https://new.abb.com/products/robotics/industrial-robots/irb-14000-yumi.
[42] "Sensitive robotics LBR IIWA," KUKA, 2020, https://www.kuka.com.
[43] P. Ferrari, A. Flammini, S. Rinaldi, E. Sisinni, D. Maffei, and M. Malara, "Impact of quality of service on cloud based industrial IoT applications with OPC UA," *MDPI Electronics*, vol. 7, 2018.
[44] N. Dall'Ora, S. Centomo, and F. Fummi, "Industrial-iot data analysis exploiting electronic design automation techniques," in *IEEE International Workshop on Advances in Sensors and Interfaces (IWASI)*, 2019, pp. 103–109.
[45] "BOX-IO gateway," https://www.box-io.com, EDALab s.r.l., 2021.
[46] "Kubernetes - production-grade container orchestration," https://kubernetes.io/, Cloud Native Computing Foundation, 2021.
[47] "Influxdb - act in time. build on influxdb." https://www.influxdata.com/, InfluxData Inc., 2021.
[48] P. Rajan and A. Sekar, "Linear circuit analysis," in *The Electrical Engineering Handbook*. Academic Press, 2005, pp. 3–29.
[49] "SDM230 series - single phase metering," https://www.eastroneurope.com, Eastron (Metering) Europe Limited, 2021.
[50] "SDM630 series - three phase metering," https://www.eastroneurope.com, Eastron (Metering) Europe Limited, 2021.



Nicola Dall'Ora (S'20) is currently a Ph.D. student at University of Verona. He received his M.Sc. degree in Computer Science and Engineering at University of Verona, with a thesis on Predictive Maintenance of Industrial Assets. His main researches focuses on predictive maintenance of Cyber Physical Production Systems (CPPS) and fault injection in multi-domain systems described at different abstraction levels.

Author version



Khaled Alamin is currently a Research Assistant at Politecnico di Torino. He received his M.Sc. degree in Mechatronics Engineering at Politecnico di Torino, with a thesis on Machine Learning Techniques to improve Die Casting Process. Since July 2019 he joined the Department of Control and Computer Engineering at Politecnico di Torino as a Research Assistant. His main research interest focuses on machine learning and deep learning techniques applied to energy optimization and industrial 4.0.



Enrico Fraccaroli (M'16) is a postdoctoral research fellow at the Department of Computer Science of the University of Verona since May 2019. He received his Ph.D. degree in computer science from the University of Verona, Italy, in May 2019. His research interests are the development of new methodologies for the efficient simulation and functional safety evaluation of embedded platforms composed of analog, digital and network components.



Massimo Poncino (SM'12-F'18) is a Full Professor of Computer Engineering with the Politecnico di Torino, Torino, Italy. His current research interests include several aspects of design automation of digital systems, with emphasis on the modeling and optimization of energy-efficient systems. He received a PhD in computer engineering and a Dr.Eng. in electrical engineering from Politecnico di Torino. He is a Fellow of the IEEE.



Davide Quaglia (M'03) received the PhD degree in computer engineering from Politecnico di Torino, Italy, in 2003. Currently he is an Associate Professor in the Computer Science Department, University of Verona, Italy. He is co-author of about 90 papers on Networked Embedded Systems, Networked Control Systems, Cyber-Physical Systems. He was co-founder of EDALab, a spin-off of University of Verona. He is member of the DATE Program Committee.



Sara Vinco (M'09) is Assistant Professor with Tenure Track at Politecnico di Torino, Turin, Italy. She received the Ph.D. degree in Computer Science from the University of Verona, Verona, Italy, in 2013. Her current research interests include digital twins, energy efficient electronic design automation, efficient simulation and optimization of energy systems, and techniques for simulation and validation of heterogeneous embedded systems.

Author version