Production Engineering (2021) 15:397–412
https://doi.org/10.1007/s11740-021-01037-3

PRODUCTION MANAGEMENT

A virtual commissioning based methodology to integrate digital twins into manufacturing systems

Giacomo Barbieri[1] · Alberto Bertuzzi[2] · Andrea Capriotti[2] · Lorenzo Ragazzini[2] · David Gutierrez[3] · Elisa Negri[2] · Luca Fumagalli[2]

Received: 11 January 2021 / Accepted: 17 February 2021 / Published online: 1 March 2021

Abstract
Digital Twin (DT) is considered a key approach to enhance the system reactivity to uncertain events due to its ability to getting data from the field and triggering actions on the physical asset. Given the modern technological and rapidly changing work environment, it is likely that in the next years companies will need to retrofit their manufacturing systems by integrating DTs. In this context, it is fundamental to define the necessary steps for the development of DTs and for their integration into manufacturing systems through a DT architecture. In response to this issue, a methodology based on Virtual Commissioning is proposed. A stepwise approach is illustrated in which the DT is designed, integrated and verified using a virtual environment. The methodology is validated through the integration of a DT into a flow shop for the implementation of a scheduling reactive to machine breakdown. By following the steps of the proposed methodology, a DT architecture able to improve the makespan of the studied flow shop is developed, suggesting the potential applicability of the approach to industrial manufacturing systems.

Keywords Manufacturing · Virtual commissioning · Digital twin · Architecture · Retrofitting · Design methodology · Production scheduling

1 Introduction

Mass customization and shortening product life cycles pose a heightened set of requirements on modern production systems [28]. Fast responses to changing conditions have been found to be a key to competitive advantage for manufacturing companies [36]. In this context, Digital Twin is considered a key approach to enhance the *system reactivity* to uncertain events [13].

Digital Twin (DT) represents the next wave in modelling, simulation, and optimization technology [35]. According to Kritzinger et al. [19] and Negri et al. [31], DT ”exploits sensed data, mathematical models and real-time data elaboration in order to forecast and optimise the behaviour of the production system at each life cycle phase, in real time”. DT has been applied in different areas of manufacturing with the common target to increase competitiveness, productivity, reactivity and efficiency [6]. Given the above, business advantage and added value can be generated for the enterprise in retrofitting their manufacturing systems with the integration of DTs.

✉ Elisa Negri
elisa.negri@polimi.it

Giacomo Barbieri
g.barbieri@uniandes.edu.co

Alberto Bertuzzi
alberto1.bertuzzi@mai.polimi.it

Andrea Capriotti
andrea1.capriotti@polimi.it

Lorenzo Ragazzini
lorenzo.ragazzini@polimi.it

David Gutierrez
dags@xcelgo.com

Luca Fumagalli
luca1.fumagalli@polimi.it

[1] Department of Mechanical Engineering, Universidad de los Andes, Bogotá, Colombia

[2] Department of Management, Economics and Industrial Engineering, Politecnico di Milano, Milan, Italy

[3] Xcelgo A/SXcelgo A/S, Ry, Denmark

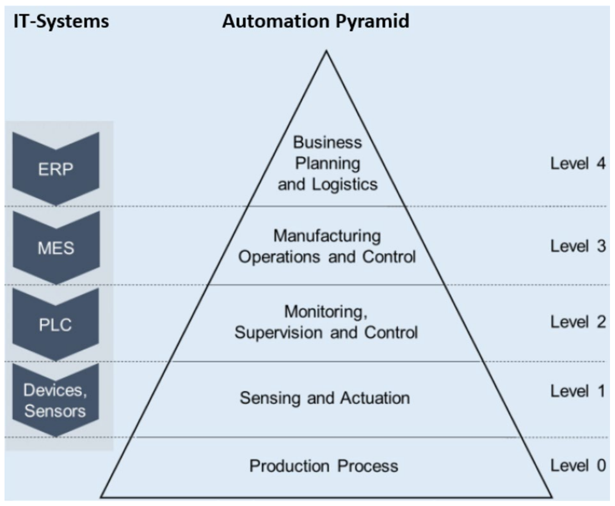

Fig. 1 Current perception of the automation pyramid, consisting of five (0 to 4) different hierarchy levels and corresponding IT-Systems [17]

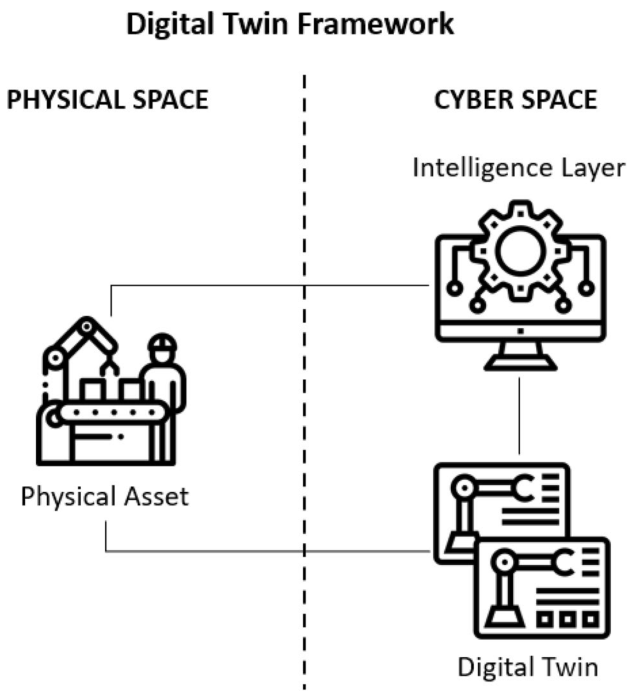

Fig. 2 Common actors within the DT frameworks presented in the literature

DTs are digital models enhanced with the bilateral communication between the physical and the cyber space [19]. In traditional simulation, the digital representation of an existing physical asset does not use any form of automated data exchange between the physical asset and the digital one. In a DT, the data flow between an existing physical asset and a digital one is fully integrated in both directions. In this way, the digital model is synchronized with the status of the physical asset and the results of the simulation can be directly implemented to optimize the physical asset, increasing system reactivity. To implement DTs, the digital model and its bilateral communication must be integrated into the control architecture of the manufacturing system.

Manufacturing systems operate in accordance with the classical Automation Pyramid [1, 14]. This normative separates a generic manufacturing company information and control system into five different hierarchical levels; see Fig. 1. To integrate DTs into manufacturing systems, a *DT architecture* must be built in which DTs, MES (Manufacturing Execution System) and PLCs (Programmable Logic Controllers) are interfaced and synchronized [3, 34, 42]. Furthermore, considering that the DT acts as a virtual test bed to evaluate the different 'what-if' scenarios that may optimize the physical asset, an *'intelligence' layer* that hosts the rules and the knowledge to choose among alternatives must be developed for the decision-making [30]. Given the complexity of the problem, it is desirable to perform these operations following a methodology that guides the user in the development of the DT model and architecture.

1.1 Digital twin: frameworks and architectures

Li et al. [22] define the term 'framework' as a conceptual layered structure of a system for a set of functionalities, while 'architecture' as the instantiation of the framework through implementation technologies. Given that, the literature concerning the design of DT applications has focused on the definition of *frameworks* and on the identification of available technologies for the instantiation of the frameworks into *architectures*. Some examples are next reported.

Lu et al. [24] propose a DT framework consisting of an information model, a communication mechanism and a data processing module. Tao et al. [40] introduce a five-dimension DT framework based on physical entities, virtual models, services, data, and connections. Lim et al. [23] list the main functionalities of a DT framework as communication, representation, computation and microservices. The illustrated works also indicate available technologies for the instantiation of the frameworks into architectures. The presented frameworks vary on the basis of the application but all have common elements (Fig. 2): the presence of a physical and a cyber space, the use of DTs and intelligence layers for supporting the decision-making, and the bilateral communication for the continuous interaction, synchronization and optimization between the DTs, their physical counterpart and the external, surrounding

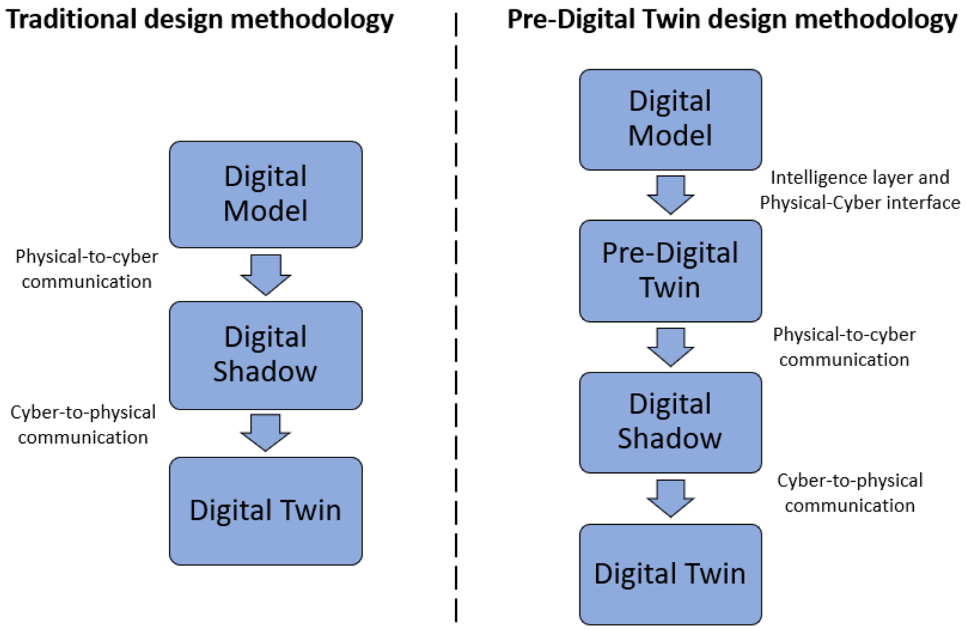
Traditional design methodology
Digital Model
Physical-to-cyber communication
Digital Shadow
Cyber-to-physical communication
Digital Twin
Pre-Digital Twin design methodology
Digital Model
Intelligence layer and Physical-Cyber interface
Pre-Digital Twin
Physical-to-cyber communication
Digital Shadow
Cyber-to-physical communication
Digital Twin

Fig. 3 'Traditional' and 'Pre-Digital Twin' methodology for the design and verification of DT architectures. The arrows indicate the operations to be performed for moving from one phase to the subsequent

environment [2]. Even if the functionalities and technologies for the generation of DT frameworks and architectures are illustrated, authors do not clarify the necessary steps to develop and integrate DTs into manufacturing systems.

1.2 Digital twin: design methodologies

A *methodology* to design DT models and architectures can be extrapolated from the contribution of Kritzinger et al. [19] starting from the definition of the different integration levels between the physical and the cyber space. The phases of the design methodology are depicted in the left-hand side of Fig. 3 and consist in:

1. Digital Model (DM): digital representation of an existing physical object that does not use any form of automated data exchange between the physical object and the digital one. This phase generates a digital copy of the physical object that does not vary its status in an autonomous way;
2. Digital Shadow (DS): DM with an additional automated one-way flow between the state of an existing physical object and a digital one. This phase enhances the DM by synchronizing it with the status of the physical object;
3. Digital Twin (DT): DM in which the data flow between an existing physical object and a digital one is fully integrated in both directions. This phase enhances the DS for exploiting its decision-making ability, since the information obtained from the digital model can be used to implement changes on the physical object.

These phases define a stepwise methodology for the implementation of different communication capabilities of the DT. However, the development of the intelligence layer is not considered. The intelligence layer is generally developed at the end of the design process and directly implemented in the physical asset [43]. Therefore, an intermediate virtual phase in between the DM and the DS would be desirable for developing the intelligence layer necessary for the decision-making, and for identifying the interface in between the physical and the cyber space before the implementation; see right-hand side of Fig. 3.

According to Madni et al. [26], *Pre-Digital Twin* is defined as "a virtual generic executable model of the envisioned system that is typically created before the physical prototype is built. Its primary purpose is to mitigate technical risks and uncover issues in upfront engineering". Since ages, Pre-Digital Twins – generally known as virtual prototypes – have been utilized for the development of physical assets. However, to the best of authors' knowledge they have not been used for the design and verification of DT architectures. In this work, Virtual Commissioning is embraced to support the development and integration of DTs within the 'Pre-Digital Twin' phase (Fig. 3).

1.3 Virtual commissioning within digital twin architectures

Virtual Commissioning (VC) is generally utilized for the design and verification of the control software of complex manufacturing systems due to its ability to speed-up the commissioning process [20]. In the context of DT, VC has been utilized within the robotics domain as a synchronized digital representation of the existing physical asset. For instance, Xia et al. [43] adopt the VC to represent manufacturing cells, simulate system behaviors, predict process faults, and adaptively control manipulated variables. Kousi et al. [18] utilize the VC to enable system reconfiguration through shared environment and process perception. Burghardt et al. [5] use an immersive robotics environment that integrates VC and Virtual Reality to achieve the automatic programming of industrial robots. In the domain of CNC machine tools, Shen et al. [39] utilize VC to tune the control parameters of the servomotors and to evaluate the kinematic performance of the physical asset. In domains different from the robotics and CNC machine tools, few works claim to use the VC as digital object within DT architectures [15, 33, 38]. However, in these works VC is utilized to design and verify the control software before its deployment. According to the paradigm proposed from Kritzinger et al. [19], these works should be classified as digital models and not as DTs, since they are not characterized by the bilateral communication between the physical and the cyber space.

1.4 Research objective

The illustrated literature review shows examples on the use of VC within DT architectures. However, there is a lack of applications of VC for the generation of a virtual environment to design, integrate and verify DT architectures before their physical implementation. This virtual environment would allow to interface and synchronize DTs, MES and PLCs, and to develop the intelligence layer before the implementation in the physical asset. Given the above, a Virtual Commissioning based methodology is proposed in this work to integrate DTs into manufacturing systems. The methodology is validated through a case study in which a DT for production planning and control is integrated into a flow shop. The article is structured as follows: the proposed VC-based methodology is introduced in Sect. 2. Sect. 3 applies the methodology to a case study for the implementation of an event-driven reactive scheduling through DT. Obtained results are discussed in Sect. 4 and finally, Sect. 5 presents the conclusions and sets the directions for future work.

2 Virtual commissioning based methodology

Concerning the design methodologies illustrated in Fig. 3, different implementation patterns have been developed for the 'Digital Shadow' and the 'Digital Twin' phases on the basis of the technologies selected within the DT architecture. For instance, Negri et al. [30] utilize the OPC communication through Level 2 Matlab S-Functions[1] to interface a manufacturing system with a DT simulated in Simulink[2]. With the objective to propose a universal approach, these two phases are not included within the presented methodology since these are technology-dependent and specific implementation patterns must be developed.

Given the above, the presented Virtual Commissioning based methodology is intended as a tool for the development of the 'Digital Model' and the 'Pre-Digital Twin' phases, and is meant to be applied to an already designed / operating manufacturing system in which the control architecture must be retrofitted with DTs. Therefore, its output is the definition of a DT architecture verified through a virtual environment.

The proposed methodology is depicted in Fig. 4. In the figure, the methodology is represented as a linear sequence of operations. However, iterations may be necessary throughout the process since all the different phases depend on each other. Next, each phase is described.

2.1 Framework

The conceptual DT layered structure and functionalities are identified without considering their implementation technologies. For instance, in the manufacturing domain this phase defines how to integrate the DT within the automation pyramid, along with the functionalities that the DT must fulfill; e.g. to classify the health state of the physical asset, to stop the production in case of breakdown, etc. The definition of the functionalities is separated from their implementation since this abstraction avoids the generation of 'biased' solutions and has been demonstrated to bring benefits to the design process such as enhanced reuse and traceability [37].

2.2 Technology

The technologies for instantiating the framework into an architecture are selected. Specific types of software and simulations are identified, based on the functionalities defined in the previous phase; e.g. Finite Element Analysis in Ansys[3] may be chosen for the DT, Python code written in Anaconda[4] for the intelligence layer, etc. Finally, the actors that will be interfaced within the architecture are specified. For instance, a Raspberry Pi controller responsible for the data acquisition may be chosen to communicate the physical asset with the enterprise cloud hosting the DTs. Within this phase, only the communicating actors are identified. A criterion for

[1] https://mathworks.com/help/simulink/sfg/writing-level-2-matlab-s-functions.html.

[2] https://mathworks.com/products/simulink.html.

[3] https://www.ansys.com/.

[4] https://www.anaconda.com/.

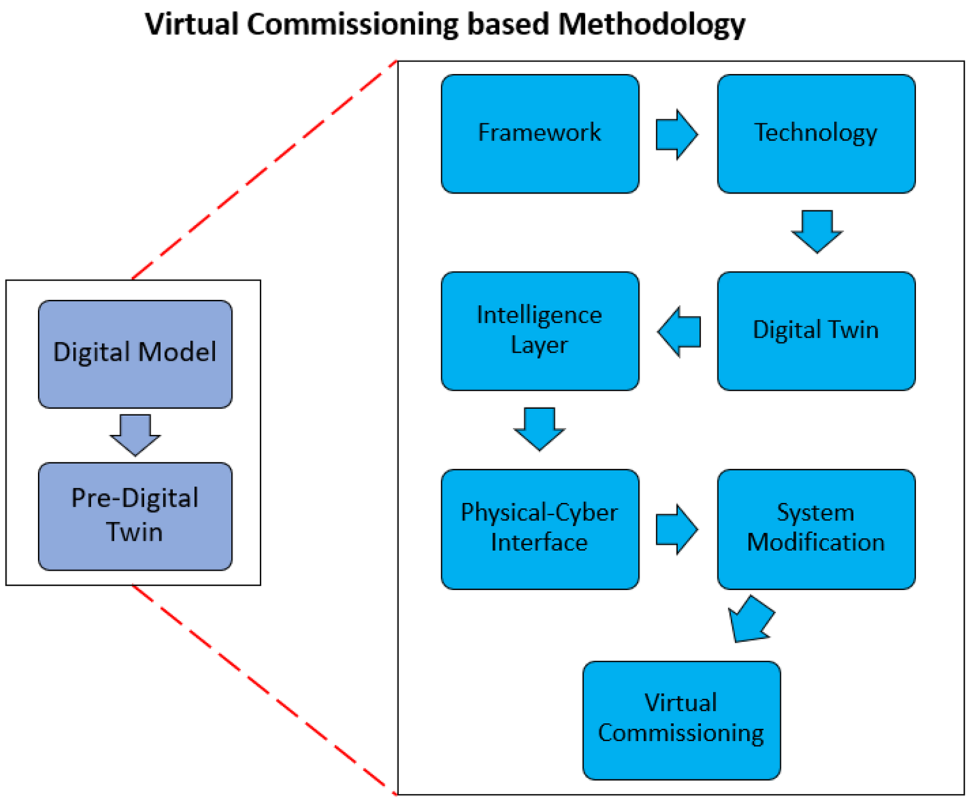

Fig. 4 Representation of the proposed VC-based methodology and its relation with the 'Pre-Digital Twin' design methodology (Fig. 3)

this selection process must be the capability to interface the identified actors. Whereas, the exchanged signals are defined within the 'Physical-Cyber Interface', and all the protocols and communication technologies are implemented within the 'Digital Shadow' and 'Digital Twin' phases, thus outside the focus of this work.

2.3 Digital twin

The DT models are utilized to forecast and optimize the behaviour of the physical asset [32]. These are developed using the software and types of simulation selected in the previous phase. Once the modeling process has been completed, the models are validated by comparing their behavior with the one of the physical asset. As any modeling process, the fidelity of the models is a necessary requirement for the DT implementation [8]. Furthermore, the models must be flexible enough to reproduce the behavior of the physical asset at each life cycle phase. For instance, the models may implement failure and repair functionalities, may be defined with physical parameters updated with the current state of the physical plant; e.g. friction coefficients, etc.

2.4 Intelligence layer

The intelligence layer is developed starting from the defined functionalities and the selected implementation technologies. The intelligence layer hosts the rules and the knowledge to choose among the different 'what-if' scenarios that may optimize the physical asset. Furthermore, it uses the DTs as virtual test beds to evaluate the evaluated alternatives. The algorithms utilized within this layer depend on the application domain and on the functionalities that the DT must fulfill. For instance, machine learning algorithms may be adopted to optimize the production [27], expert systems to detect faults [25], etc. Within this phase, the interaction between the DTs and the intelligence layer is exploited with the aim to compare different algorithms and to tune the specific parameters.

2.5 Physical-cyber interface

Once the DTs and the intelligence layer have been designed, all the information is available to define the signals to be exchanged among the different actors within the DT architecture. As shown in Fig. 2, signals are exchanged between:

1. Physical asset-intelligence layer: to monitor the physical asset and to implement the results of the decision-making process;
2. Physical asset-digital twin: to synchronize the DTs with the status of the physical asset;

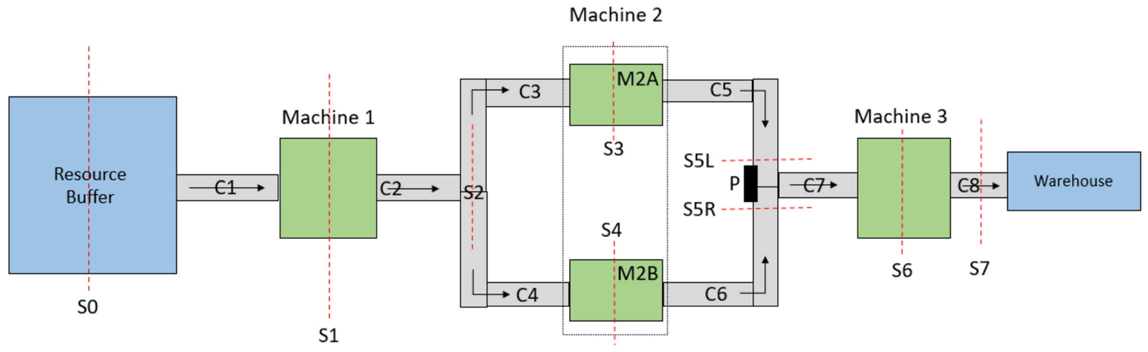

Fig. 5 Schematic representation of the flow shop utilized in the case study. Sensors and actuators are represented in the figure, along with machines. C_i indicates the i-th conveyor, S_j the j-th light barrier sensors, and P a pneumatic cylinder

3. Intelligence layer-digital twin: to evaluate alternatives and choose among them during the decision-making process.

This phase also establishes the order in which signals are exchanged and which sequence of operations are implemented. This information constitutes the specifications of the 'System Modification' phase. For instance, a chronological sequence of signals and operations may be: "once the intelligence layer receives a breakdown signal, a rescheduling operation must be implemented".

2.6 System modification

After the definition of the physical-cyber interface, changes must be implemented in the original physical asset following the identified specifications. For instance, the system control software may be modified to integrate the DTs and the intelligence layer, additional sensors may be installed, etc. In this phase, modifications are developed only at a virtual level. In fact, the objective of the methodology is to verify them through VC before their physical implementation.

2.7 Virtual commissioning

VC is used to verify the changes implemented to the original physical asset. A simulation model of the physical plant described at the level of sensors and actuators is developed and connected to the control software for the generation of the VC. Again, the VC model must be validated by comparing its behavior with the one of the physical asset to rely on the results provided from this verification process. Then, the VC is interfaced with the cyber space for the generation of a virtual environment that emulates the DT architecture. The obtained virtual environment is utilized to verify the built architecture. Different conditions that may occur in the physical asset (e.g. machine breakdown, etc.) are injected in the VC model to verify the response of the developed DT architecture.

3 Case Study

In this section, the proposed methodology is applied to a case study. The objective is to validate that the methodology enables the development of a DT architecture and not to retrofit a real manufacturing system and to select the best tools for its optimization; e.g. digital models, optimization algorithms, etc. Therefore, a simple case study is utilized and the identified tools are not the best ones for the considered application.

Job shop scheduling or the *job-shop problem* (JSP) is an optimization problem in which various manufacturing jobs are assigned to machines at particular times while trying to minimize the makespan [44]. In this context, the DT enables the dynamic scheduling and the reconfiguration of the manufacturing resources in response to the occurrence of uncertain events [41]. In this case study, a DT is integrated into a flow shop for the implementation of a scheduling reactive to machine breakdown.

The studied flow shop is illustrated in Fig. 5. It consists of three machines in series able to process nine different types of jobs. Each job is characterized by an identification number from 1 to 9. Machine 1 (M1) and machine 3 (M3) can manufacture any job, while M2A odd jobs and M2B pair jobs. Each job j is defined with a processing time on each i-th machine (p_{ij}). The resource buffer and the warehouse are assumed with infinite capacity. The flow shop consists in a Kanban Pull system since a new job is generated when the previous one enters machine M2 [7].

The control architecture of the flow shop is shown in Fig. 6. The operator inputs a production sequence in Microsoft Excel that acts as a MES. This information is sent to a CoDeSys PLC[5] which controls and monitors the actuators and the sensors of the production process. Light barrier sensors (S) are utilized to identify the position of the jobs within the production line. Sensors $S2$ and $S7$ also detect

[5] https://www.codesys.com/.

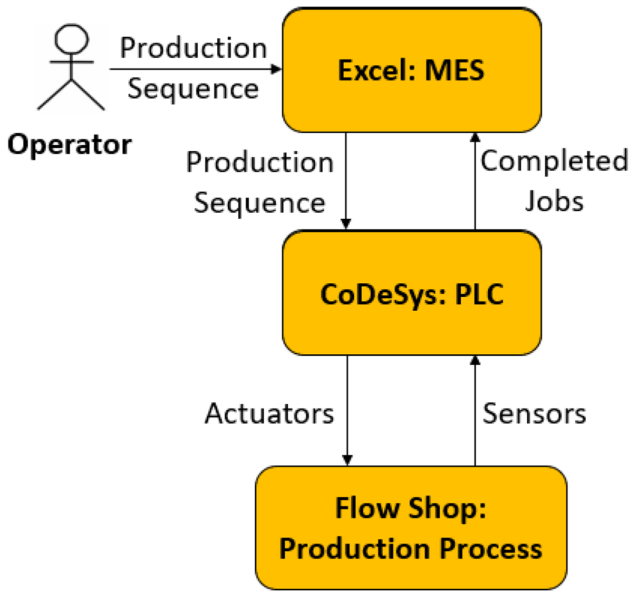

Fig. 6 Control architecture of the flow shop of Fig. 5

the job identification number to respectively bring the job to the proper M2 machine and to trace the number of the completed job. Conveyors (C) and pneumatic cylinder P are used to move the job within the production line in accordance with its identification number. Only machines M2A and M2B can have breakdowns, and these two machines are characterized with a constant repair time (r_i). The breakdown is immediately detected from the PLC that stops the production until the machine has been fixed. Finally, the information concerning the completed jobs is sent to Excel to generate a production report containing the time in which each job has been loaded and completed.

In this case study, a Digital Twin must be integrated to improve the system performance. The improvement is achieved through the information available from the DT by means of a scheduling algorithm that reschedules the remaining jobs to be produced once a machine breakdown occurs.

In this retrofitted flow shop, a production order is input in the MES and the sequence that minimizes the makespan is computed. Once a breakdown occurs, a rescheduling is implemented for the remaining jobs to be produced. Concerning the rescheduling, the scheduling algorithm knows the machine repair time (r_i), and a DT is utilized to test different sequences of jobs. The described event-driven reactive scheduling is designed and verified using the *VC-based methodology* illustrated in Sect. 2. Next, the implementation of each phase is illustrated.

3.1 Framework

In this phase, the functionalities that the DT architecture must fulfill are identified, along with the actors involved in the implementation of each functionality. This information is represented in Fig. 7 by means of a UML use case diagram [9]. The identified functionalities are:

- Rescheduling: the DT architecture must be able to reschedule the production after the occurrence of a breakdown. This functionality also includes the identification of the production sequence starting from the initial production order. The rescheduling operation is led by the intelligence layer which generates different production sequences and tests them in the DT to identify the one with the lower makespan. The information of a machine breakdown is sent to the intelligence layer by the PLC; see 'Breakdown Detection' functionality in Fig. 7. The MES is also participating in the realization of the functionality, since the intelligence layer reads from the MES the remaining jobs to be produced before starting the rescheduling operation;
- Model Synchronization: the synchronization of the DT with the production process is led by the PLC. The PLC sends to the DT the information concerning the current status of the production process. Then, the DT implements the necessary changes to mimic the condition of the physical asset;
- Breakdown Detection: M2A and M2B are machines able to detect and communicate breakdowns to external systems. Once a breakdown occurs, they send this information to the PLC. Then, the PLC transmits the breakdown information to the intelligence layer to trigger the rescheduling operation.

3.2 Technology

The technologies for instantiating the framework into an architecture are selected. Excel and CoDeSys are maintained respectively as MES and PLC since the DT must be integrated into the original flow shop. A *Discrete Event Simulation* (DES) running in Simulink is selected as DT, and Matlab as intelligence layer [10]. *Simulink* provides a SimEvents[6] library which enables the modeling of production systems through various blocks, such as generators, queues, and servers. DES is utilized in place of continuous time (CT) simulation since DES is computationally more efficient, allowing the quick test of different production sequences. *Matlab* is selected for the intelligence layer since

[6] https://mathworks.com/help/simevents.

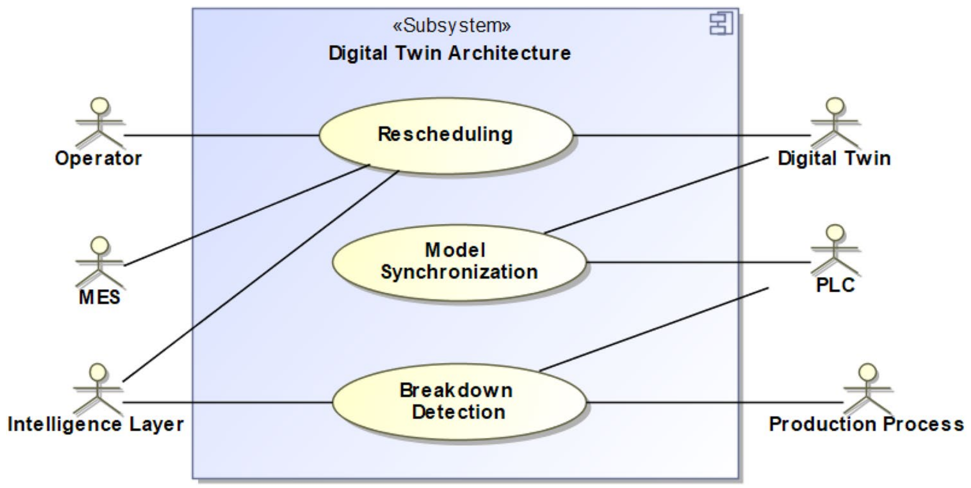

Fig. 7 UML use case diagram representing the functionalities that the DT must fulfill, and the actors involved in the implementation of each functionality

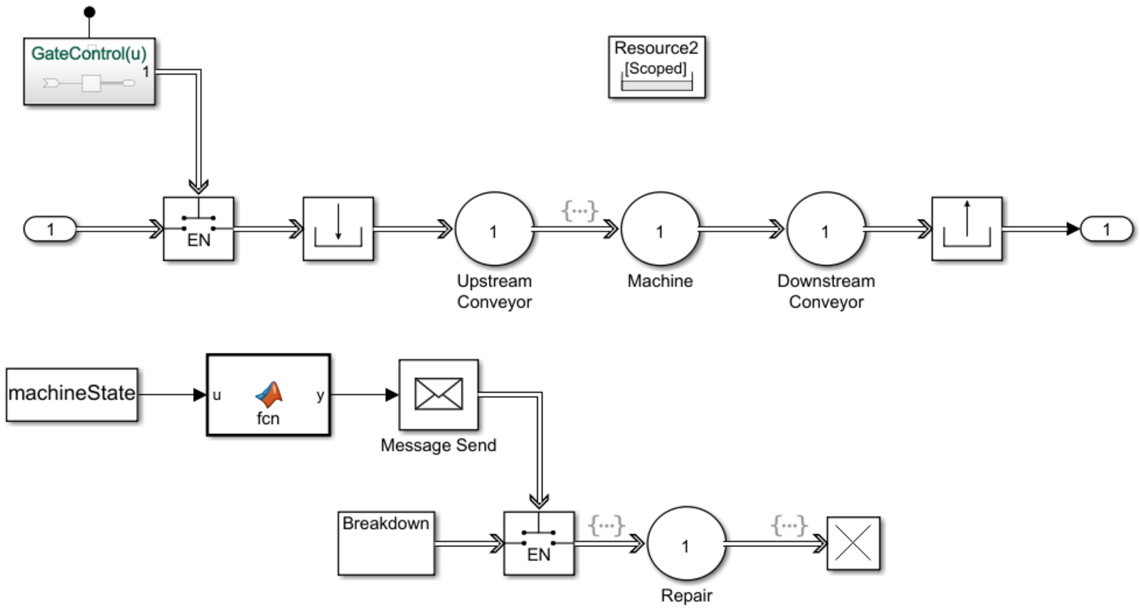

Fig. 8 DES model of a machine including the breakdown and repair functionalities

it enables the implementation of optimization algorithms and the control of the Simulink model. Finally, a CT simulation in Experior[7] is adopted as VC. In this case, CT simulation is utilized since the model must replicate the dynamics of the production process to be properly interfaced with the PLC. It must be noted that these actors have been selected also considering that they can be physically interfaced.

3.3 Digital Twin

In this phase, the DT model that will be utilized to forecast and optimize the behaviour of the manufacturing system is developed.

Each machine is modeled in Simulink using the pattern shown in Fig. 8. The machine has a constant processing time dependent on the processed job (p_{ij}), and is characterized by an upstream and downstream conveyor. These elements are modeled in Simulink as *'server'* actors. A machine can process only one job at a time. This constraint is implemented

[7] https://xcelgo.com/experior/.

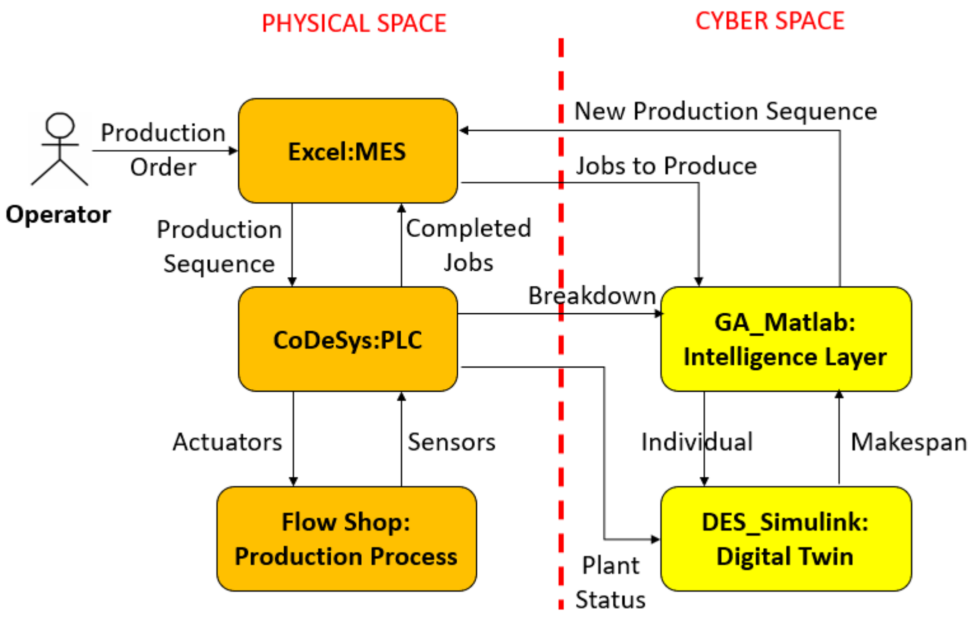

Fig. 9 Representation of the signals exchanged within the DT architecture

setting the *’resource pool’* actor to one resource. The *’entity gate’* actor is utilized to model the breakdown and repair functionalities. When the *’Machine State’* variable is set to 1, the lower entity gate is opened, while the upper one is closed. This operation prevents the input of jobs in the machine, and triggers the machine reparation (’*Repair*’ server). Once the machine has been fixed, the upper gate is opened and the machine can restart processing jobs. This pattern is applied to machines M2A and M2B, while machines M1 and M3 are only characterized by the nominal behaviour (upper part of Fig. 8), since breakdowns are assumed to never occur in machines M1 and M3. The machines are then connected following the configuration of the studied flow shop (Fig. 5).

Finally, the behavior of the DT should be validated with the one of the production process. In this case study, this operation is not implemented since the objective is to validate the methodology and not to apply it to a physical system. Therefore, a production process is not utilized. The application of the methodology to a physical manufacturing system is left as future work.

3.4 Intelligence layer

In this case study, the intelligence layer must be able to schedule the remaining jobs to be produced. A *Genetic Algorithm* (GA) is developed for this purpose. The objective of the GA is to identify a production sequence that minimizes the makespan. The GA is an iterative and stochastic population-based metaheuristic algorithm inspired to some of the processes which characterize natural evolution [11]. The solutions are directly encoded as permutations representing the sequence in which the jobs must be executed. The algorithm uses a two-points ordered crossover (2OX) operator and a swap operator for the mutation functionality [29]. Finally, the tournament criterion is adopted for the selection functionality [12]. After each iteration of the GA, children and parents are joined and sorted according to their fitness value, and only the ones associated with the lowest makespan are moved to the subsequent generation. The algorithm has been implemented in Matlab to be integrated within the developed DT architecture.

3.5 Physical-cyber interface

After the design of the scheduling algorithm, the *signals* to be exchanged among the different actors of the DT architecture are identified. Therefore, a cyber space – containing the intelligence layer and the DT – is integrated to the control architecture of the original manufacturing system.

Figure 9 indicates a static representation of the exchanged signals. However, the chronological sequence in which signals are exchanged and which sequence of operations are implemented must be defined before modifying the original manufacturing system. The *UML sequence diagram* can be utilized for this purpose since it enables the representation of the chronological sequence of events and the involved actors for the implementation of the specified functionality [9]. The UML sequence diagram of Fig. 10 specifies the rescheduling operation due to the occurrence of a breakdown. Here, the intelligence layer receives the information of a breakdown from the PLC, and the remaining jobs to be produced from the MES. The PLC also sends to the DT the information concerning the failed machine. Then, the DT implements the necessary changes to mimic the plant status (’*Synchronize*’ action in Fig. 10), and the intelligence layer generates the

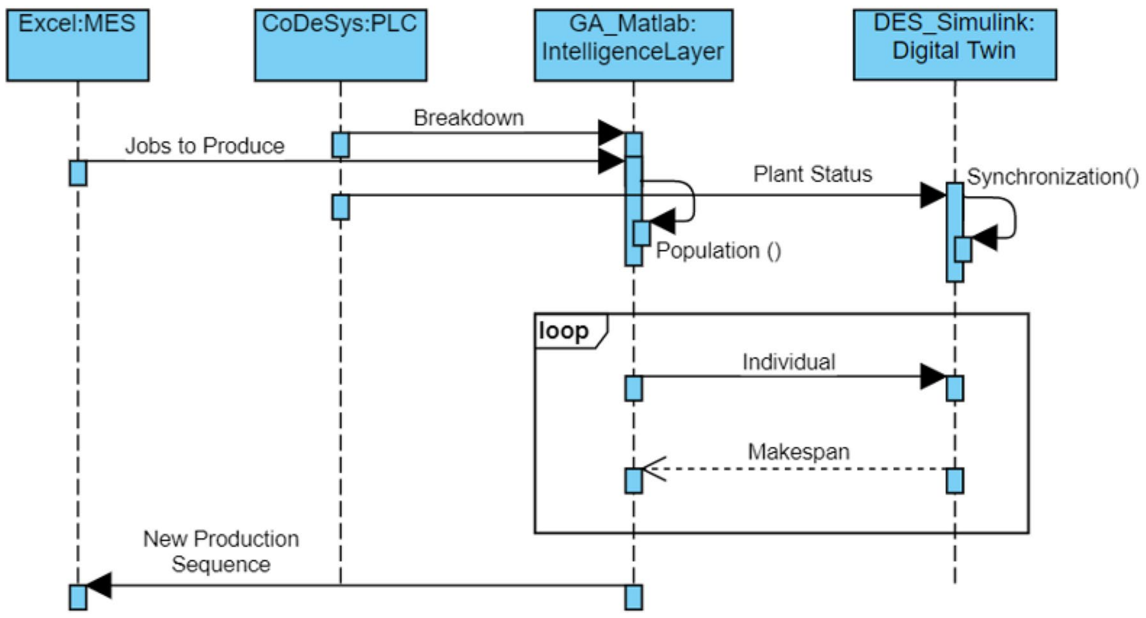

Fig. 10 UML sequence diagram representing the chronological sequence of exchanged signals and implemented operations during the rescheduling

production sequences that will be tested in the DT ('*Population*' action). Each production sequence ('*Individual*') is run in the DT to calculate its makespan. Finally, the production sequence with the lowest makespan is written in the MES.

3.6 System modification

The information developed in the 'Physical-Cyber Interface' constitutes the specifications for the changes that are implemented in this phase – at a virtual level – into the original manufacturing system.

In the original manufacturing system, the breakdown is immediately detected from the PLC that stops the production line until the machine has been fixed. Whereas, the DT is meant to improve the system performance by rescheduling the remaining jobs to be produced and continuing the production, even if a machine is being repaired. This behavior implies modifications to the *control software* of each machine.

The implemented control software for machine M2A is illustrated in Fig. 11 with a UML state machine diagram [9]. In the original manufacturing system, machine M2A is only characterized by the *'Stop'* and *'Normal Functioning'* states, since the production line is stopped in case of a breakdown. Here, the *'Wait'* and *'Breakdown'* states are integrated. The *'Wait'* state is reached when a breakdown occurs in machine M2B. Production is temporarily stopped waiting for the intelligence layer to calculate the new production sequence. Once the rescheduling operation has been completed, machine M2A can restart processing jobs. The *'Breakdown'* state is entered when a breakdown occurs in machine M2A. The repair operation is started after the calculation of the new production sequence. This choice was taken in order to avoid the repair time being affected by the computation time of the scheduling algorithm. In fact, the scheduling algorithm takes a no negligible time to calculate the new production sequence. If the repair operation were immediately started, the repair time considered into the rescheduling operation would be different from the actual repair time.

Finally, the specifications of the control software of each machine are converted into *PLC code* using the design pattern proposed in Bonfé et al. [4]. This pattern is selected since it implements a one-to-one translation of UML state machines into PLC code, enhancing the traceability of the process.

3.7 Virtual commissioning

In this phase, a VC is developed to verify the changes implemented to the original production process. Then, the VC is interfaced with the cyber space to test the defined DT architecture.

A *CT model* of the flow shop is developed in Experior (Fig. 12). The model also includes an HMI (Human Machine Interface) through which breakdowns are injected. To rely on the subsequent verification process,

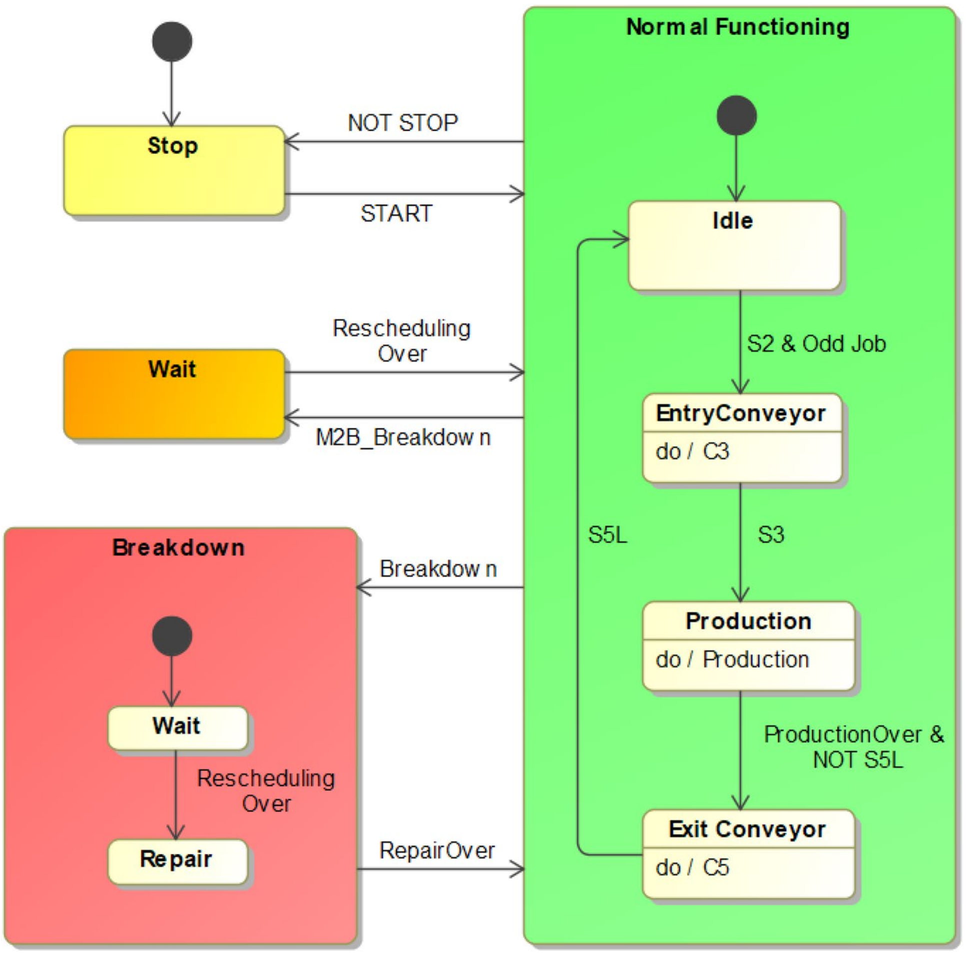

Fig. 11 UML state machine diagram representing the control software of the machine M2A for the implementation of the DT architecture. The nomenclature of the sensors and actuators follows the flow shop schematic shown in Fig. 5

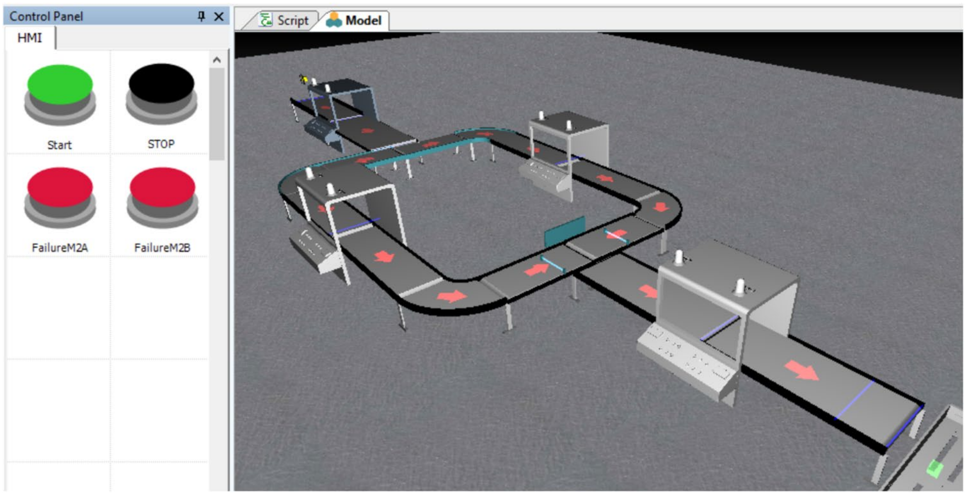

Fig. 12 Experior model of the studied flow shop. Breakdowns are injected using the two buttons on the HMI

the model is validated by comparing the behavior of the CT model with the DES one; i.e. DT model.

A 'traditional' Virtual Commissioning simulation is implemented to verify the modified control software. CoDeSys offers an integrated Soft-PLC – named CoDeSys Win Control V3 – that emulates an industrial controller under Windows. The Experior model is connected to CoDeSys Win Control V3 through OPC communication

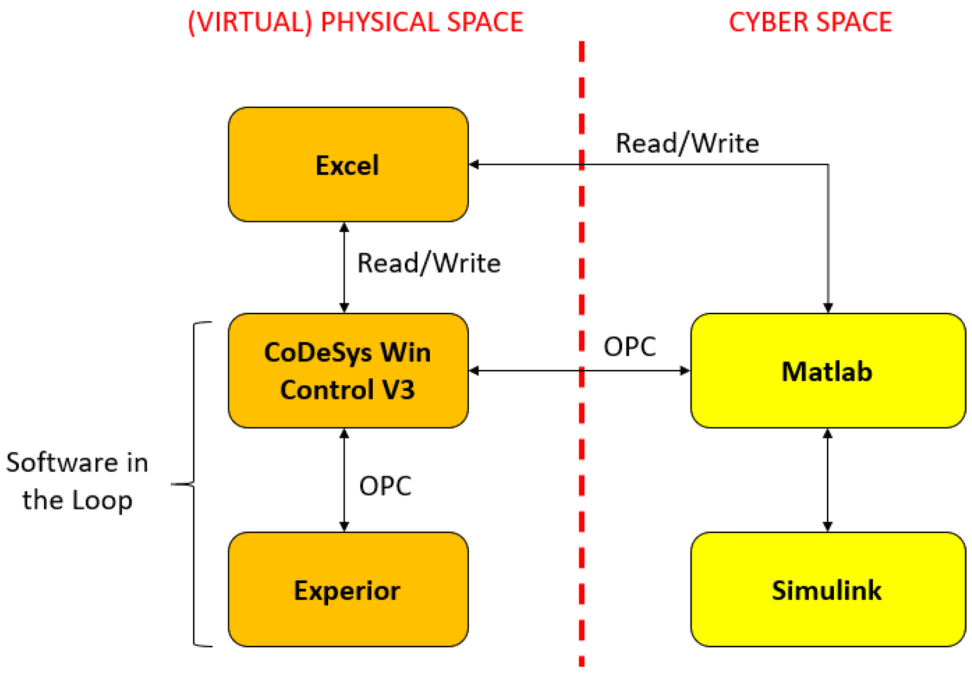

Fig. 13 Emulated DT architecture built for the verification of the DT integrated to the manufacturing system

for the generation of a Software in the Loop VC simulation [21]. Furthermore, the capability of CoDeSys to read and write Excel files is utilized to achieve the communication with the MES. Then, the system is run under different scenarios (i.e. normal functioning and breakdown) to debug the developed control software.

Finally, the emulated *DT architecture* is built by integrating to the VC simulation, the interface between CoDeSys-Matlab (OPC communication), and Matlab-Excel (read/write). Then, the DT architecture is verified. It can be noticed that a completely virtual environment is generated to verify the DT architecture by interfacing Experior, CoDeSys, Excel and Matlab (Fig. 13).

4 Results and discussions

The objective that guided the design of the DT architecture was to improve the system performance through the information available from the DT model. A scheduling algorithm was developed to schedule the production sequence starting from the production order. Furthermore, the algorithm was also utilized to reschedule the remaining jobs to be produced once a machine breakdown occurs.

A '*breakdown*' scenario was implemented to verify that the developed DT architecture achieves better makespans than the original manufacturing system. In this scenario, production orders are randomly generated containing 50 jobs, and the same production sequence is assigned to the two flow shops. Then, a breakdown is injected in machine M2B after 250s of production, and the machine repair time is set to 300s. The original flow shop stops the whole production line until the machine has been fixed. Within the retrofitted flow shop, the DT model is updated with the status of the flow shop and it is utilized from the intelligent layer to test different scheduling sequences. Therefore, a rescheduling is implemented with the remaining jobs to be produced, and production continues. A video is made available to the reader for clarifying the operations implemented from the retrofitted flow shop.[8] The resulting production for the original and the retrofitted flow shop are respectively illustrated in the Gantt charts of Figs. 14 and 15. In the figures, each job is represented as a rectangle representing the Time in System; i.e. the left side corresponds to the loading time into the system and the right side to the completion time. Since the original flow shop stops the production until the machine has been fixed, the loaded jobs remain idle during machine reparation; see Fig. 14. Whereas, the DT architecture implemented within the retrofitted flow shop enables the rescheduling of the remaining jobs to be produced, and the flow shop continues to process odd jobs in machine M2A; see Fig. 15. This results in a reduced makespan for the retrofitted with respect to the original flow shop. Given the stochasticity of the experiment, 10 repetitions are run for this scenario, after the verification that further repetitions did not modify significantly the results. In each repetition, the same jobs are processed with the original and the retrofitted flow shop. The obtained results are illustrated in Table 1. Results obtained for the breakdown scenario. From left to right, the

8 https://youtu.be/kRcVmyT-NI8.

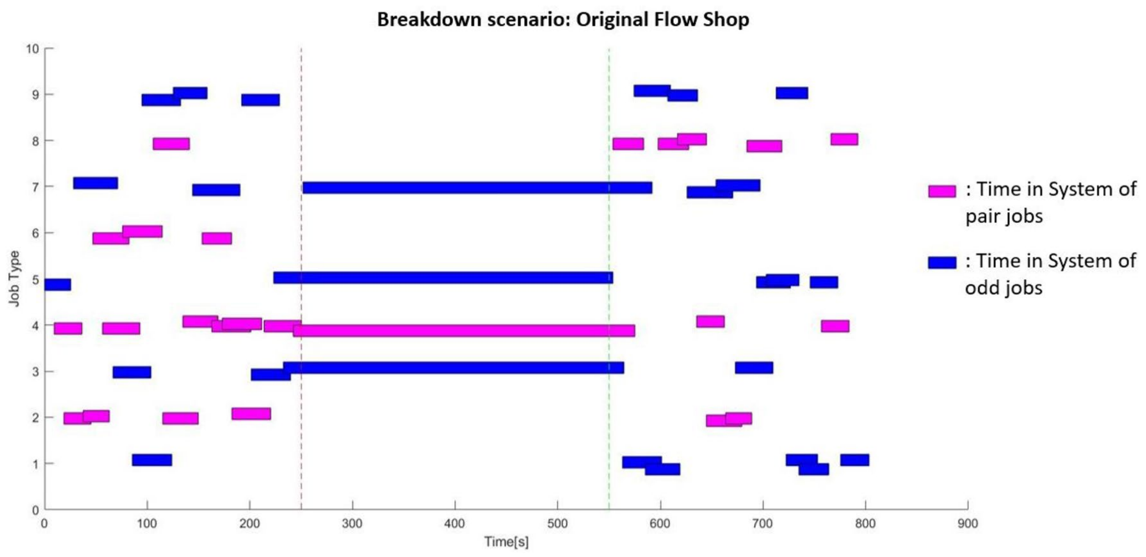

Fig. 14 Breakdown scenario: Gantt chart with the production of the original flow shop. Production is stopped while M2B is being repaired

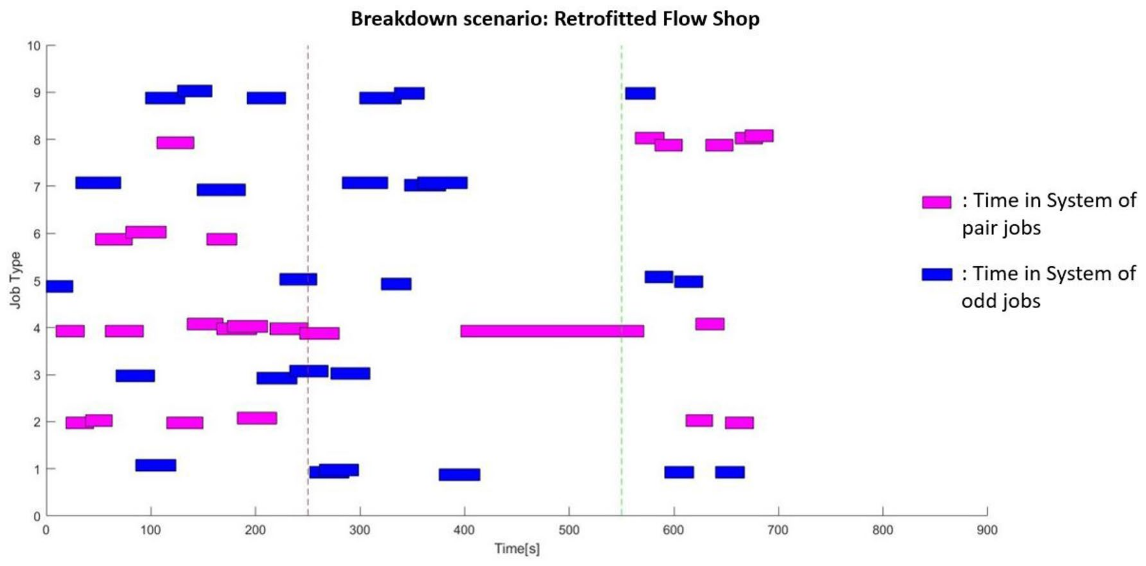

Fig. 15 Breakdown scenario: Gantt chart with the production of the retrofitted flow shop. Odd jobs are processed while M2B is being repaired

columns respectively indicates: (i) # of the experiment repetition; (ii) makespan of the original flow shop; (iii) makespan of the retrofitted flow shop; (iv) makespan improvement of the retrofitted with respect to the original flow shop. In this scenario, the makespan of the retrofitted flow shop is reduced on average by 9.1%. The computation time of the GA algorithm is not considered in the experiment.

It can be noticed that the integration of the DT within the original manufacturing system generates an improvement in the makespan, achieving the objective for which the DT was designed. With respect to a traditional rescheduling, the synchronization of the digital model with the status of the physical plant – typical of a DT – enables the identification of an optimal production sequence for the current status of the production process. In this way, the DT architecture brings additional benefits with respect to traditional rescheduling since it allows the system optimization in response to uncertain events as breakdowns.

Finally, we review the proposed methodology and its implementation with respect to its future applicability in industrial manufacturing systems, and to the contributions that may generate to the research in the DT field. By following the steps of the proposed methodology, we were able to design and verify the DT architecture before

Table 1 Breakdown scenario: makespan obtained with the original and the retrofitted flow shop

# Repetition	Original (s)	Retrofitted (s)	Δ(%)
1	803.3	715.0	11.0
2	813.8	768.2	5.6
3	833.0	786.0	5.6
4	815.4	752.8	7.7
5	833.2	730.5	12.3
6	830.6	770.6	7.2
7	807.3	721.1	10.7
8	818.1	751.5	8.1
9	840.9	735.7	12.5
10	838.0	756.2	9.7
Mean	823.4	748.8	**9.1**

its implementation – using a virtual environment. With respect to the current practice of directly implementing the DT in the manufacturing system, the application of the methodology generated three main benefits: (i) the virtual interface between the intelligence layer and the DT enabled the selection and tuning of the algorithm utilized within the intelligence layer; (ii) the utilization of the VC simulation enabled the verification of the changes implemented in the original manufacturing system; (iii) the virtual environment – generated by virtualizing and interfacing all the actors of the DT architecture – allowed the simulation of the DT architecture and the identification of possible issues that would occur in the physical implementation; e.g. the no negligible effect of the computation time taken from the scheduling algorithm.

Considering the modern technological and rapidly changing work environment, it is likely that in the next years companies will need to retrofit their manufacturing systems by integrating DTs. The approach proposed in this article provides a stepwise methodology for the DT implementation. Furthermore, the utilization of a virtual environment provides the advantage for the enterprise that is not compelled to stop the production and waste products while designing and verifying the DT architecture. From an academic point of view, the generated virtual environment may enhance the investigation in the field of DT, since novel algorithms and approaches can be tested in a virtual environment before their implementation in the production process.

The only drawback of the methodology is that can be implemented only if the actors selected in the DT architecture can be interfaced and their behavior can be simulated. For instance, commercial MES should be tested to validate the applicability of the methodology in industrial manufacturing systems.

5 Conclusion and future work

In the next years, it is likely that companies will need to retrofit their manufacturing systems by integrating *Digital Twins*. Different frameworks and architectures have been defined for the implementation of DTs. However, it is fundamental to define the necessary steps for the development of DTs and for their integration into manufacturing systems through a DT architecture. In this context, the objective of this research work was to identify a *methodology* for the design and verification of the Digital Twin architecture before the implementation in the manufacturing system. The objective has been reached by defining a methodology based on the *Virtual Commissioning* simulation. The methodology proposes a stepwise approach in which the DT is designed, integrated and verified using a virtual environment. Finally, the methodology has been validated through a case study consisting in the integration of a DT into a flow shop for implementing a scheduling reactive to machine breakdown.

The proposed methodology provides three main *benefits* with respect to the current practice of directly implementing the DT into the manufacturing system:

- Algorithms for the decision-making: the virtual interface between the intelligence layer and the DT enables the selection and tuning of the algorithms utilized for the decision-making;
- Changes in the original manufacturing system: the VC simulation enables the verification of the changes implemented in the original manufacturing system to integrate the DT;
- DT architecture: the virtualization and interface of all the actors allow the generation of a virtual environment to simulate the DT architecture and to identify possible issues that would occur in the physical implementation.

This work contributes to the research on DT by clarifying the necessary steps to develop a DT architecture, and by providing a virtual environment for its design, integration and verification before the implementation in the production process. The methodological approach and the identified tools can be utilized from *companies* for retrofitting their manufacturing systems, and from *universities* for testing novel algorithms and approaches in a virtual environment before their implementation.

Notably, the proposed methodology constitutes a preliminary concept that in the future should be further validated and improved. Some future works identified are:

– Industrial validation: in this work, the methodology has been validated with a prototype flow shop. Its applicability in industrial manufacturing systems should be investigated;
– Learning Factory: since the VC enables to digitize manufacturing systems, the developed virtual environment may be utilize as a learning factory to teach DT. Learning activities should be developed to assess its potentiality as learning factory;
– Condition-based maintenance: in this work, the machine breakdown was immediately detected from the PLC. In the context of Condition-based Maintenance [16], failures are detected by processing the data acquired from the sensors placed in the machines, and sent to the cyber space by means of IoT technologies. Along with algorithms for improving the system performance, the methodology should integrate the development of algorithms for failure detection.

Funding Open access funding provided by Politecnico di Milano within the CRUI-CARE Agreement.

References

1. ANSI/ISA (2000) ISA95: enterprise-control system integration. Technical report, International Society of Automation.
2. Barricelli BR, Casiraghi E, Fogli D (2019) A survey on digital twin: definitions, characteristics, applications, and design implications. IEEE Access 7:167653–167671
3. Biesinger F, Weyrich M (2019) The facets of digital twins in production and the automotive industry. In: 2019 23rd International Conference on Mechatronics Technology (ICMT), IEEE, pp 1–6
4. Bonfe M, Fantuzzi C, Secchi C (2013) Design patterns for model-based automation software design and implementation. Control Eng Pract 21(11):1608–1619
5. Burghardt A, Szybicki D, Gierlak P, Kurc K, Pietruś P, Cygan R (2020) Programming of industrial robots using virtual reality and digital twins. Appl Sci 10(2):486
6. Cimino C, Negri E, Fumagalli L (2019) Review of digital twin applications in manufacturing. Comput Ind 113:103130
7. Deleersnyder JL, Hodgson TJ, Muller-Malek H, O'Grady PJ (1989) Kanban controlled pull systems: an analytic approach. Manag Sci 35(9):1079–1091
8. Durão LFC, Haag S, Anderl R, Schützer K, Zancul E (2018) Digital twin requirements in the context of industry 40. IFIP international conference on product lifecycle management. Springer, Berlin, pp 204–214
9. Friedenthal S, Moore A, Steiner R (2014) A practical guide to SysML: the systems modeling language. Morgan Kaufmann, London
10. Fumagalli L, Polenghi A, Negri E, Roda I (2019) Framework for simulation software selection. J Simul 13(4):286–303
11. Goldberg DE (1989) Genetic algorithms in search, optimization, and machine learning. Addison-Wesley Pub, Co, London
12. Goldberg DE, Deb K (1991) A comparative analysis of selection schemes used in genetic algorithms. Foundations of genetic algorithms, vol 1. Elsevier, Amsterdam, pp 69–93
13. Hasan-ul-Shoaib S, Macchi M, Pozzetti A, Carrasco-Gallego R (2018) A routine-based framework implementing workload control to address recurring disturbances. Prod Plan Control 29(11):943–957
14. IEC62264 (2002) Enterprise-control system integration. Technical report.www.iso.org/standard/57308.html
15. Janda P, Hajicek Z, Bernardin P (2019) Implementation of the digital twin methodology. In: book: Proceedings of the 30th International DAAAM Symposium, Intelligent Manufacturing and Automation, pp 533–538
16. Jardine AK, Lin D, Banjevic D (2006) A review on machinery diagnostics and prognostics implementing condition-based maintenance. Mech Syst Signal Process 20(7):1483–1510
17. Körner MF, Bauer D, Keller R, Rösch M, Schlereth A, Simon P, Bauernhansl T, Fridgen G, Reinhart G (2019) Extending the automation pyramid for industrial demand response. Proced CIRP 81:998–1003
18. Kousi N, Gkournelos C, Aivaliotis S, Giannoulis C, Michalos G, Makris S (2019) Digital twin for adaptation of robots ' behavior in flexible robotic assembly lines. Proced Manuf 28:121–126
19. Kritzinger W, Karner M, Traar G, Henjes J, Sihn W (2018) Digital twin in manufacturing: a categorical literature review and classification. IFAC Pap 51(11):1016–1022
20. Lechler T, Fischer E, Metzner M, Mayr A, Franke J (2019) Virtual commissioning-scientific review and exploratory use cases in advanced production systems. Proced CIRP 81:1125–1130
21. Lee CG, Park SC (2014) Survey on the virtual commissioning of manufacturing systems. J Comput Des Eng 1(3):213–222
22. Li R, Verhagen WJ, Curran R (2020) A systematic methodology for prognostic and health management system architecture definition. Reliab Eng System Saf 193:106598
23. Lim K, Zheng P, Chen C (2020) A state-of-the-art survey of digital twin: techniques, engineering product lifecycle management and business innovation perspectives. J Intell Manuf 31:1313–1337
24. Lu Y, Liu C, Wang KIK, Huang H, Xu X (2020) Digital twin-driven smart manufacturing: connotation, reference model, applications and research issues. Robot Comput Integr Manuf 61:101837
25. Luo W, Hu T, Zhang C, Wei Y (2019) Digital twin for cnc machine tool: modeling and using strategy. J Ambient Intell Hum Comput 10(3):1129–1140
26. Madni AM, Madni CC, Lucero SD (2019) Leveraging digital twin technology in model-based systems engineering. Systems 7(1):7
27. Min Q, Lu Y, Liu Z, Su C, Wang B (2019) Machine learning based digital twin framework for production optimization in petrochemical industry. Int J Inform Manag 49:502–519
28. Mindas M, Bednar S (2016) Mass customization in the context of industry 4.0: implications of variety-induced complexity. Adv Ind Eng 21–39
29. Murata T, Ishibuchi H, Tanaka H (1996) Genetic algorithms for flowshop scheduling problems. Comput Ind Eng 30(4):1061–1071

30. Negri E, Berardi S, Fumagalli L, Macchi M (2020) Mes-integrated digital twin frameworks. J Manuf Syst 56:58–71
31. Negri E, Fumagalli L, Macchi M (2017) A review of the roles of digital twin in cps-based production systems. Proced Manuf 11:939–948
32. Negri E, Pandhare V, Cattaneo L, Singh J, Macchi M, Lee J (2020) Field-synchronized digital twin framework for production scheduling with uncertainty. J Intell Manuf. https://doi.org/10.1007/s10845-020-01685-9
33. Orive D, Iriondo N, Burgos A, Saráchaga I, Álvarez ML, Marcos M (2019) Fault injection in digital twin as a means to test the response to process faults at virtual commissioning. In: 2019 24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), IEEE, pp 1230–1234
34. Răileanu S, Borangiu T, Ivănescu N, Morariu O, Anton F (2015) Integrating the digital twin of a shop floor conveyor in the manufacturing control system. International workshop on service orientation in holonic and multi-agent manufacturing. Springer, Berlin, pp 134–145
35. Rosen R, Von Wichert G, Lo G, Bettenhausen KD (2015) About the importance of autonomy and digital twins for the future of manufacturing. IFAC Pap 48(3):567–572
36. Samir K, Maffei A, Onori MA (2019) Real-time asset tracking; a starting point for digital twin implementation in manufacturing. Proced CIRP 81:719–723
37. Sangiovanni-Vincentelli A (2007) Quo vadis, sld? reasoning about the trends and challenges of system level design. Proc IEEE 95(3):467–506
38. Scheifele C, Verl A, Riedel O (2019) Real-time co-simulation for the virtual commissioning of production systems. Proced CIRP 79:397–402
39. Shen W, Hu T, Yin Y, He J, Tao F, Nee A (2020) Digital twin based virtual commissioning for computerized numerical control machine tools. Digital twin driven smart design. Elsevier, Amsterdam, pp 289–307
40. Tao F, Zhang M, Liu Y, Nee A (2018) Digital twin driven prognostics and health management for complex equipment. CIRP Ann 67(1):169–172
41. Wang C, Jiang P, Ding K (2017) A hybrid-data-on-tag-enabled decentralized control system for flexible smart workpiece manufacturing shop floors. Proc Inst Mech Eng Part C 231(4):764–782
42. Wang Y, Wu Z (2020) Digital twin-based production scheduling system for heavy truck frame shop. Proc Inst Mech Eng Part C. https://doi.org/10.1177/0954406220913306
43. Xia K, Sacco C, Kirkpatrick M, Saidy C, Nguyen L, Kircaliali A, Harik R (2020) A digital twin to train deep reinforcement learning agent for smart manufacturing plants: environment, interfaces and intelligence. J Manuf Syst
44. Zhang J, Ding G, Zou Y, Qin S, Fu J (2019) Review of job shop scheduling research and its new perspectives under industry 4.0. J Intell Manuf 30(4):1809–1830