

**UNIVERSITÀ DEGLI STUDI
DI MODENA E REGGIO EMILIA**

Dottorato di ricerca in Ingegneria dell'Innovazione Industriale

Ciclo XXXVI

Smart Digital Twins in Industry 4.0

Candidato: Matteo Martinelli

Relatore (Tutor): Prof. Marco Lippi

Correlatore (Co-Tutor): Prof. Prof. Marco Picone

Coordinatore del Corso di Dottorato: Prof. Franco Zambonelli

Contents

Abstract	iii
1 Introduction	1
2 State of the Art	7
2.1 Industrial Evolution	7
2.2 Industrial Architectures	11
2.3 Digital Twins: Birth and Rising	15
2.4 Digital Twins Applications	24
2.5 Intelligence in Industry	32
2.6 AI and DTs: Synergies Over Industrial Scenarios	38
2.7 Industry 4.0: Not Yet Enough	41
3 Digital Twins: a Solution for Industrial Complexity?	45
3.1 Complexity in the Industrial Environment	46
3.2 Modelling DT in the Industrial Context	60
3.2.1 Data Ingestion & Augmentation	61
3.2.2 Physical World Actionability	63
3.2.3 Cyber-Physical Relationships	64
3.2.4 Composition & Hierarchical Views	65
3.2.5 Application Interaction	68

3.3	Organical Application of DTs	70
3.3.1	A Hierarchical Architecture for Smart Industries	71
3.3.2	Hierarchical Abstraction in Industrial Digitisation	72
3.4	DT Contribution Over Complexity	75
3.5	Resulting Advantages	78
4	Industrial Digital Twins and AI	85
4.1	The Role of AI	85
4.2	Observer AI	94
4.3	Advisor AI	97
4.4	Controller AI	99
4.5	Embedded AI	103
5	Experimental Evaluation	109
5.1	Informational AI and Causality	111
5.1.1	Production System Model: Description & Terminology	114
5.1.2	Industrial Simulation Setting	117
5.1.3	Learning Causality of Failures	120
5.1.4	Impact of Digital Twins on Digital Complexity	124
5.2	Experimental Environment & DT Implementation	128
5.2.1	Experimental Setup	131
5.2.2	Machine-Layer Digitisation: Adapters Structure	134
5.2.3	Digital Twins Cores: Augmentation Functions	136
5.2.4	Digital Twins Enabled Actionability	139
5.2.5	Digital Twins Hierarchy Navigability	140
5.2.6	Architecture Performance & Complexity	143
5.3	External AI & Industrial Behaviour	146
5.4	Embedded DT Intelligent Augmentation	150

Abstract

Industries have become increasingly complex production entities. They involve technologies of various types, of new and old generations, with realities where machines have not replaced man and where their transversal integration is increasingly complex. Furthermore, the evolving market demands and the speed of their changes make it necessary to have production systems capable of responding promptly to new requests, making the complexity of the systems even higher. Modern advances in computer science have generated great expectations for improvement in the industrial production world. However, the distributed nature of these systems, their heterogeneity and inherent complexity make the application of these modern solutions critical. In response to industrial requirements, their heterogeneity and the rapid response to changes required, a level of abstraction capable of intelligently “disconnecting” physical complexity from digital complexity is therefore necessary. This dissertation presents a possible abstraction of the physical industrial world implemented through Digital Twins and their interaction with modern Artificial Intelligence techniques, defining four possible interaction patterns between the two technologies, named *Observer AI*, *Advisor AI*, *Controller AI* and *Embedded AI*. Extensive experimentation is conducted by implementing the envisioned architectures on both simulated and physical environments, also providing a practical measurement of the impact of Digital Twins on

reducing *digital complexity* in the considered domain. The resulting architectures enable an organic representation of the production context in the digital world, improving both its processes and its global management.

Chapter 1

Introduction

Industrial systems foster our economics, being the backbone of world markets. They are the main contributor to the production of goods necessary to carry on human everyday life, ranging from basic commodities to more complex solutions, even as one-off products. The best economic leverage of the production system through industrial evolution has always been simplification, standardisation and economy of scale [15] [5]. Indeed, over time, the process of simplification as well as the standardisation of products, made possible the scientific and systematic approach to their production process. The direct consequence was the ability to saturate industrial resources in an environment positively prone to activities organisation [31].

Approach to standardisation permeated the entire industrial entity, from actual activities needed to be carried out on the shop floor, to the resulting industrial organisation at middle and higher levels. It is not a surprise to find in the market organisational structures most of the time similar, with an overall approach guided by hierarchies, domain expertise in the area of operations as well as very divided and well-defined responsibilities. Indeed, a production system specialised in food transformation, for example, can be

compared with a system crafting automotive pistons in how operational activities are organised. The approach *abstracting* the actual technical activity nature carried out in the system is another flavour of the standardisation approach researched by manufacturing system transversal to different sectors. Nevertheless, across different sectors arose different technical solutions and organisational approaches, with the will to somehow better specialise the production system concerning the market environment. This can be the case for industrial architectures, that from the first long standardised production lines evolved in cellular manufacturing, job-shops or project-shops layout, adopting different Customer Orders Decoupling Points to better fit actual sector requirements [137] [40] [92] [50].

The described standardisation strategies, combined with others, permitted industrial managers to handle the trade-off between the growing market customisation demand, and the actual implementation and complexity costs. Nevertheless, actual standardisation strategies are becoming less and less effective over time, due to an ever-growing demand for customisation associated with big production volumes, which led to coining the term of *mass-customisation*. Facing new market standards is becoming more and more difficult for industrial systems, and the associated growing complexity, driven by the growth in product variants and the associated production techniques needed to satisfy market requests, is putting pressure on them. As a direct response, the organisation process control framework arose as *Lean Manufacturing*. Indeed, a lot of expectations emerged towards modern Information Technology (IT) solutions, such as Artificial Intelligence (AI) or Internet of Things (IoT), to make production systems *smarter*, by handling the growing complexity of meeting market demands. Moreover, the need for *intelligence* crosses also the need for managing the actual physi-

cal complexity [33]. Having the actual production system *mirrored* in the digital domain enables, indeed, the opportunity to manage the mirrored representation and the associated data generation with the same Information Technology tools expected to deeply change industrial production systems. Building a Cyber-Physical Production System (CPPS) brings opportunities, challenges and risks. In this domain, risks fall under the umbrella of complexity, as the rising level of production systems complexity should be mitigated or at least managed by future solutions, and not promoted. The best tool that researchers agree to be the solution for organic digitisation of physical assets in the industrial context is the Digital Twin (DT).

The depicted scenario brings a set of problems that need to be considered and carried out as a whole. Indeed, future solutions must have the ability to abstract the physical setup, transposing into the virtual environment only the set of abilities that are interesting from a physical asset management “perspective”, keeping a rational order concerning data associated with each physical asset. Considering standardisation needs, it is also necessary for the solution to accept a defined level of customisation, to leave to industrial managers the ability to represent the production system *as-is*, and not to adapt it to the tools expected to be implemented. And, finally, the digital representation of the system needs the ability to be integrated with Artificial Intelligence and associated data analysis tools to add *intelligence* in industrial key points, boosting the actual system when and where needed, in the expectation to manage and eventually to overcome growing production complexities.

To support the vision and propose a contribution, it will be first discussed the two main concepts of industrial complexity (*static* and *dynamic*) that are typically found in literature [33], and then investigated a third type

of complexity named *technological*. The influence of Digital Twins in managing the considered levels of complexities is then highlighted, and the concept is then carried out in experimental sections with a first simple proposal of *measuring* the practical impact of introducing Digital Twins in the industrial receipt. Objectively measuring how the introduction of Digital Twins eventually lowers the complexity of an intricate distributed system as a shop floor is crucial both for the research community, to have a common ground of discussion for gains and pains of Digital Twins, and for practitioners, to understand whether and when the technological improvement is worth the implementation effort.

The dissertation topics then move towards modelling Digital Twins and their associated features to effectively digitise Cyber-Physical-Production-System (CPPS). Domain experts intuitively understand how a Digital Twin can represent an associated physical asset *organising* and *exposing* organically its information. However, how it is useful in the industrial domain and why it is worth considering some Digital Twin characteristics for shop floors needs to be pointed out. For example, the potential composition of Digital Twins can be very useful in such a domain, to represent not only physical assets but also to move towards *physical concepts* that can be found in the considered context.

After the definition of this common ground about Digital Twins and how they can be applied in the shop-floor scenario, the introduction of intelligence is considered, to obtain a *smart* Digital Twin ecosystem to mirror the underlying physical state-of-the-affairs. Towards the depicted direction, four possible patterns of interaction between Artificial Intelligence and a Digital Twin ecosystem are introduced. Their description is made by keeping in mind possible applications in the reference context, and how they can impact in

moving from a *descriptive* towards a *prescriptive* and eventually *autonomous* Digital Twin ecosystem [27] which are named *Observer AI*, *Advisor AI*, *Controller AI*, and *Embedded AI*.

The envisioned relationships between physical assets, digitised counterparts via Digital Twins and the insertion of intelligence have been experimentally evaluated in 3 different experiments. The purpose of the described activities was to validate the contribution under different scopes. System feasibility has been considered in measuring the Digital Twins ecosystem introduced overhead, while complexity measures have been computed to evaluate the impact of the digitisation activity in improving its management. Finally, Artificial Intelligence models have been added in the receipt, to practically demonstrate considered interactions with a Digital Twin ecosystem.

The dissertation outline is the following:

- Chapter 2 reviews background concepts and state-of-the-art industrial architectures, Artificial Intelligence and Digital Twins applied to the industrial domain, as well as their mutual usage in the considered scenario;
- Chapter 3 describes industrial complexity highlighting how Digital Twins can help in mitigating issues related to the management of such a domain;
- Chapter 4 presents the four emerging interaction patterns between Artificial Intelligence tools and Digital Twins applied to industrial contexts;
- Chapter 5 further develops such scenarios in practical applications to validate the proposed contribution. Experimental environments consider a virtual simulation of a production facility, then an implementation in an actual physical scenario, taking into account hardware

controlling industrial machines at the process level, a Digital Twin Industrial Ecosystem, and different interaction patterns with Artificial Intelligence models;

- Finally, Chapter 6 concludes the dissertation, proposing future paths to make the proposed vision effective and useful for application in the considered domain.

Chapter 2

State of the Art

The industrial sector, as well as technologies enabling the next evolving step in production systems, made a lot of steps forward over time. In the next paragraph, different topics of results achieved until now by industries, AI and DT are going to be explored and discussed. After a brief overview of past industrial revolutions that brought us to the nowadays setting, contributions to industry architectures and AI applications in the domain are going to be presented. Then, a description of the concept of DT is provided, to provide foundational definitions about what is and why it is important in the considered scenario. A final overview of research findings in the application of DTs, and DTs mixed with AI techniques is provided, before a conclusive section summarising the actual State of the art.

2.1 Industrial Evolution

The industry is not a concept that has been always around during economic history. Indeed, its advent arrived only in the seventeenth century, after the so-called “*first industrial revolution*”. Industrial history and main milestones

have been widely discussed and there is a general agreement upon foundational ideas characterising their revolutionary aspects. Industrial revolutions have always been characterised by strong technological innovations, that lead to big changes in the global market structure. The first one has been guided by the invention of the steam-powered engine, and other innovations in the textile sector. The second industrial revolution, instead, has been pushed by the findings made in the chemical sector, the start in the usage of electricity, as well as by the emergence of the scientific organisation of industrial production on the very first assembly lines. At each step of the industrial history, big changes were reflected both at the industrial level, with factories that changed in their form and organisation, both at the societal level, with the birth of new social classes and several impacts on human society. That was also the case of the third industrial revolution, guided this time by innovations in the electronics, informatics and telecommunications sectors. This revolution was characterised by the very first introduction to shop-floor environments of computers and related electronic equipment. Moreover, the first applications of IT and mechanic technologies lead to the birth of the first industrial robots, pushing industrial operations effectiveness to its boundaries.

The widely known historic trajectories of industrial revolutions brought to us, nowadays a tremendous amount of disruptive and incremental innovations that are now part of our day-to-day life. The petrol engine, aeroplanes, trains, computers as we know them, modern internet. Some of the evolving technologies of our epoch brought experts to asses that we probably are at the beginning of a new industrial disruption guided, again, by the matching of rising technologies. Therefore, a potential “fourth industrial revolution” is upon us, guided especially by a set of 3 IT pillars [145]:

- Internet-of-Things - IoT (and Industrial Internet of Things - IIoT):

sensors and actuators are starting to be connected to the Internet, enabling the monitoring of distributed scenarios, collecting data, and interacting with the environment. Therefore, the widely tested and reliable infrastructure of the internet is starting to be considered the backbone of the next generation of connected things. By the way, IoT-generated data by themselves support limited decisions in time and value. Nevertheless, the big stream of data generated by the equipment enables the second pillar of the industrial revolution, that of Big Data.

- Big Data: there is not a general definition for Big Data. In this context, is possible to consider them as a big volume of information collected in time. Big Data are characterised by the “5 Vs”: Volume, Velocity, Variety, Veracity, Value. Value is the most important V for industry, and the most difficult to achieve. The V of Value represents the capacity to translate the big volume of data enabled by IoT and other existing technologies into a concrete value or outcome for the company. The Value can impact, among others, 3 industrial aspects: process optimisation, with a consequent reduction of costs; increase of the desirability of an existing product, thus augmenting the accepted market price; and creation of new businesses through new business models or products.
- Artificial Intelligence - AI: AI is the last industry 4.0 pillar. It represents the possibility to extract knowledge and wisdom from data and information and to build smart systems capable of interacting with the environment smartly, such as humans do. Typical AI tasks include data classification, pattern recognition, forecasting, data analysis, and causal learning across observed variables. It can be centralised if it resides only in one machine, or distributed if it involves multiple machines

or multiple independent software entities (as agents).

These three pillars are strongly interconnected and positive feedback can be induced amongst them: distributed sensors deployed in the IoT arena continuously produce Big Data; Big Data in turn feeds AI systems; finally, AI and Big Data influence the real-world environment using distributed actuators enabled by IoT, in a closed-loop fashion [145].

The fourth industrial revolution brings to the experts big expectations, as enables a set of new scenarios and applications at the shop-floor level not possible before. For example, a new wave of automation guided by AI application support is expected to change shop-floor solutions and designs. Nevertheless, a lot of challenges are still in front of the scientific community, to bring the fourth industrial revolution to reality.

2.2 Industrial Architectures

Ref.	PS	SPA	LM	Description
[38]	x			Model T first standardisation effort.
[137]	x			First definition of scales for commonly accepted competitive priorities.
[40]	x			Priorities improvement through variability reduction and technological enhancements.
[50]	x			Competitive priorities focus with respect the CODP.
[92]	x			Competitive CODP/OPP positioning strategies and industrial architecture.
[18]		x		Energy-efficient scheduling problem definition for industrial production and shipping.
[140]		x		Two-stage three-machine assembly flow shop scheduling with two learning agents.
[129]		x		Cloud decentralised scheduling approach for a flow-shop CONWIP production line.
[71]		x		IoT based operation control.
[52]		x		Q-Learning algorithm with jobs permutation for flow-shop scheduling.
[135]		x		Weighted Q-Learning algorithm for job-shop scheduling.
[113]			x	Toyota Production System (TPS) lean manufacturing principles.
[91]			x	TPS Lean manufacturing rise and settling.
[3]			x	Lean manufacturing adoption in SMEs.
[90]			x	Lean manufacturing and production performance review.

Acronyms:

- PS: Production Streamlining
- SPA: Scheduling of Production Activities
- LM: Lean Manufacturing

Table 2.1: Summary of related works in the context of industrial architectures.

Complexity management has been the first major problem to restrain the rise of industrial assembly lines. Its nature can embrace different aspects, from the variety of management introduced by a wide product portfolio, passing through the number of different machines operating in a shop floor, to their configuration combinations, considering also the management issues of all possible negative happenings of a production environment, and so on. The major historical example comes from the first automotive assembly lines born in the first of 1900, and characterised by a series of solutions aimed to increase production efficiency, lowering as a consequence associated costs, to enable the ability to propose products to a mass public. One of those solutions, for example, came from Henry Ford and the Model T assembly line: to lower production complexity and associated resources, the car has

been proposed only black for the first years of production [38]. Assembly lines themselves are moreover the very first example of production complexity management, rising from the need to pass from a low-volume production typical of artisans to a high-volume production.

The rationalisation of production systems brought to business focus based on operations priorities [137][40] and layout differentiation, orders decoupling points and more in general, to standardisation of modern industrial architectures [92] [50]. Industrial architecture standardisation came from the need to differentiate the most suitable system concerning market requests and the nature of the crafted product. Assembly lines of the first of the '900 can be considered flow-shops: a long, highly specialised system with fixed automation aimed at producing only one highly standardised product, in high volumes. Then, usually, the lower the volumes, the higher the production portfolio variety. Following this pattern, *cellular manufacturing*, *job shops* and *project shops* emerged as responses to different production needs, following the idea of specialising the production system to obtain higher competitiveness in the market and better results in quality terms.

Nevertheless, specialised layouts came with drawbacks, therefore the *Customer Order Decoupling Point* came to mitigate them. CODP was at first defined concerning the Production Delivery lead time ratio (P/D ratio). Then, the P/D ratio has been intersected with Relative Demand Volatility or RDV [92]. Make-to-stock (MTS) environments have been theorised to cope with highly standardised products produced in high volume, leveraging the fact that the higher the production volume, the higher the level of predictability of the demand. Indeed, this architecture does not consider product orders, as it is produced based on market forecasts and then directly brought as close as possible to potential customers, as customers are generally not available

to wait for a commodity. *Make To Order* instead respond to market requests for more customised products in lower volumes concerning MTS. Customers are usually more inclined to wait for a customised product. Therefore, in the MTO, the good production is scheduled only when an order is placed. In this way, is possible to change the production set-up to enable the ability to respond to different product requests, utilising a single multi-purpose production system. Nevertheless, can happen that the customer availability in waiting for a customised product can not be handled by Make To Order (MTO) architectures. So, to be able to gain responsiveness to customer orders, a hybrid solution between MTS and MTO emerged, called Assembly To Order (ATO). The result is a Customer Order Decoupling Point placed in a convenient point, in the middle of the production system, where the production in the upstream is managed following a production forecast as MTS, thanks to higher predictability obtained by grouping sub-assemblies expected demand, while the downstream is managed by customer orders, gaining the responsiveness of a short production line.

When instead the demand is for unique (and usually big) products crafted in one slightly more piece, the most suitable architecture is the Engineer To Order (ETO). This architecture is characterised by considering in the lead time also the engineering phase of the product (usually not considered in previous systems), as usually the product does not exist or needs to be highly customised and its level of customisation does not permit to start from an already existing design.

Over time, the market evolution pushed for higher levels of customisation, leading to hybrid industrial architectures, specialised to cope with growing challenges requests. This evolution is testified by the high number of research articles in the area of production scheduling [18] [140] [129] [71] [52] [135].

Nevertheless, the level of complexity highlights the need to drive challenges connected to mass customisation with a new wave of standardised management tools. Taking for granted the necessity of customising production to acquire the ability to craft a very specific product, it is also useful to have a set of tools capable of abstracting a production system and creating a methodical approach to addressing the growing complexity and management challenges of manufacturing. The most known approach in that direction comes from Lean Manufacturing, whose roots come from Japan and the TPS (Toyota Production System) [113] [91]. The generalised interest of the market in Lean Manufacturing and its techniques. proves its usefulness. However, the lean approach has several limitations, especially among SMEs [3] [90], as its applications mostly rely on manual tracking of information, therefore on the impact of the human factor across the entire organisation. The standardisation effort brought by Lean manufacturing is therefore valuable because of the creation of a general framework for managing a manufacturing organisation. Nevertheless, a further step in developing new generation management tools capable of integrating existing architecture representations and metrics as well as opening new opportunities is desired.

2.3 Digital Twins: Birth and Rising

Ref.	DA	D	S	R	Description
[123]	x				DT for PLM integration.
[46]		x			First DT definition.
[99]	x				DT flight system with diagnosis prognostics capabilities.
[126]	x		x		Simulation of an aircraft based in its DT.
[13]	x		x		Cutting tool DT via IoT for productivity analysis and process planning.
[120]				x	DT review, difficulties, challenges and proposed paths.
[121]				x	Review of DT state-of-the-art and applications in industry.
[108]				x	DT description, vision, challenges, impacts, risks.
[103]		x			Vision of a Web of DT, abstract model and architecture.
[80]		x			DT purpose, trust, and function working principles.
[45]		x			Mirroring physical world in the digital domain.
[84]		x		x	DT principles in the IoT context and four application scenarios.
[9]	x	x			DT requirements for I4.0.
[81]		x			Challenges over DTs composition.
[49]	x				DT POC of a bending beam test bench.

Acronyms:

- DA: Domain Application
- D: Definition
- S.: Simulation
- R: Review

Table 2.2: Summary of related works considering the first conceptualisation and use of DTs.

The DT concept was first introduced by Michael Grieves between 1999 and 2002 in a presentation held at the University of Michigan under the scope of a Product Life-cycle Management (PLM) tool centre [123] [46]. The initial idea, although missing the explicit name of “Digital Twin”, had already all the elements of the DT concept, i.e. the definition of virtual and physical spaces, the link between the two spaces, and the associated bidirectional information flow. The following concept of “twinning” arose since the idea envisioned consisted of two different systems, i.e. the physical and the digital, where the virtual is fed by the physical information, mirroring in the digital domain. The context where the idea was proposed first mentioned also the PLM concept, communicating a vision where the DT follows the associated physical entity across its entire life cycle.

The concept of DT has then been strongly considered in the aerospace area by NASA. Indeed, it has been used by the agency in its technological road-maps, and for next-generation fighter aircraft and vehicles [99] [126].

In the initial Grieves' vision, the DT is a set of virtual information constructs that fully describe a potential or actual physical manufactured product from the micro-atomic level to the macro-geometrical level. Grieves categorises base-level DTs into two types: DT Prototype (DTP) and DT Instance (DTI). The DT Prototype (DTP) represents the prototypical physical artefact and contains the necessary information sets to describe and produce a physical version that duplicates the virtual version. In Grieves' proposal, physical information is strongly correlated, but not limited, to its 3D design, associated bill of materials (with material specifications), bill of processes, bill of services, and bill of disposal. The DT Instance (DTI), instead, describes a specific corresponding physical product linked to an individual DT throughout its life. Therefore, a single DT instance can be considered as bonded to one and only one physical counterpart. Nevertheless, Grieves does not specify if a single physical object may be associated with multiple DTs at the same time. In Grieves' perspective, the DTI may contain information regarding a fully annotated 3D model with General Dimensioning and Tolerances (GD&T) of a physical object, bill of materials of the asset listing current and past components, the bill of processes of the asset detailing operations performed, along with measurements and test results, the service record of the asset describing past services and component replacements, and operational states captured from actual sensor data, both current and predicted future states. Then, Grieves expands the idea of DTI under the concept of DT Aggregate (DTA), representing the aggregation of different "base-level" DTIs, having access to all associated DTIs. It can query DTIs

ad-hoc or proactively, asking questions such as the Mean Time Between Failure (MTBF) of a specific component or conducting proactive analyses based on sensor readings. Finally, Grieves identifies a DT Environment (DTE), as an integrated, multi-domain physics application space for operating on DT for various purposes. These purposes include predictive uses, where the DT is employed to predict future behaviours and performance of physical products, and interrogative uses, where DTIs are queried for current and past histories. The aggregation of data from multiple instances allows for correlation and prediction of future states, providing valuable insights for maintenance and decision-making.

In more recent times, DTs have been classified as one of the top-ten strategic technological trends of the last years in manufacturing [13] and some argue that half of all corporations worldwide might be using them in the near future [120]. DT definition and application purposes have quickly evolved from its original research field and are emerging as a new cross-domain paradigm to design and implement cyber-physical applications through the creation of synchronised software replicas of physical devices, products, and whole organisations [121]. Research is moving fast towards the definition of a general way to bridge the physical and the digital spaces [108], also thanks to a new pervasive vision that aims to create a distributed ecosystem of interconnected DTs [103], fostering initial visions proposed in [80] [45], and Grieves. Among all the technologies that had an impact on the DT vision evolution, IoT can be considered the most relevant. Indeed, thanks to IoT a quick migration to a technological ecosystem where the effective collaboration between *cyber and physical layers* represents a fundamental enabler for the next generation of applications, where DTs can and will play a crucial role. The development of the IoT itself represented a big step forward in the evolution of IT,

enabling a highly pervasive level of devices (i.e., objects other than computers or servers) equipped with sensors and actuators able to communicate through the common means of the internet. Nevertheless, its evolution was limited to the "connectivity" aspect of objects, taking care only to bring the first four layers of the ISO/OSI model on them. But how to represent those objects? How to integrate them? How to make them communicate, even if they are of different types or manufacturers? This led to a multiplicity of connection solutions, with consequent fragmentation and difficulties in having a homogeneous layer of communication across different hardware and software - characteristics that gave us the Internet as we know it today. DTs try to overcome those limitations, bringing an abstraction layer that decouples the real and the digital world with the responsibility to properly represent real-world objects in the digital domain. This abstraction gives the possibility to uniformly represent any real-world domain, standardising the approach to the development phase, cutting developing times, and enabling smoother integration and interoperability across different devices. In this way, developers and engineers can focus only on the valuable activities expected by industrial stakeholders. As a matter of fact, DTs enhance IoT applications, overcome their limitations and bring additional, useful features into the domain of connected things.

An interesting point of view in this regard is given by Minerva in [84]. The article target is to identify DTs key characteristics as a general concept applicable in the IoT context, keeping the idea of somehow transposing what is happening in the physical context into the digital one, and (eventually), vice-versa, through the mean of IoT capabilities. Among the identified characteristics, the most interesting are *augmentation*, *composability*, *representativeness*, *contextualisation*, and *entanglement*. *Augmentation* is

considered as the capability of *modify, update and improve* physical object functions over time through the mean of the associated DT leveraging the software dematerialisation. Interestingly, the *augmentation* property proposed in [84] finds another definition in [9], as the ability enabled by the DT of augmenting information ingested from the physical counterpart through different data manipulation techniques (where the most powerful belongs to AI), as well as augmenting available actions that can be operated by the DT in the digital domain as well as in the physical one, e.g. the possibility to expose APIs following different patterns through the DT in scenarios where the physical counterpart is implemented only with one communication protocol. Despite different details in the definition of augmentation capabilities, the opportunity of having a digital replica *following* its physical counterpart *before, during, and after* its lifetime opens up to a set of opportunities that can bring new abilities to the physical object through the mean of its digital representation.

Composability is a concept somehow already present in Grieves' definitions in the form of the DTA (DT Aggregate), and that is better explored in Minerva's proposal. In Minerva, composability refers to the ability to imitate human experience with physical object assemblies. Indeed, PAs can be seen as groupings of sub-parts (e.g., a robot with multiple joints and a gripper) or as the combination of several individual entities operating together within the same environment (e.g., a production line composed of multiple interconnected machines). Therefore, *composability* property of DTs represents the capability to abstract the complexity of a large system made up of multiple sub-objects and to eventually focus on a few relevant properties and behaviours without having to consider the functioning of all the aggregated sub-components. Through composability, DTs can be in charge of

abstracting a complex environment exposing only relevant information and functionalities to the application layer. This property is proposed as a way to efficiently represent sub-systems and their integration into a larger *system-of-systems*, as happens in the real world. Composability capabilities, and more in general, horizontal and vertical integration challenges, are also depicted in [81]. The proposed article has an industrial background, a typical scenario where multiple heterogeneous components are grouped to form a very specific “system entity”. The horizontal integration allows not only to representation of the single components of the composition but also an advantageous way to abstract different *views* of the same system, concerning the designed use case. While the vertical composition allows the creation of specific views starting from lower level components (physical as well as other DTs), the composition of different sub-system components into its higher level entity, as well as the mix between the two.

Representativeness deals with the capacity of the DT to be uniquely identified, directly associated with its physical counterpart, and being in charge of representing it as much as possible in terms of attributes (e.g. telemetry data, configurations, etc.), behaviours (e.g. actions that can be performed by the physical device or on it by external entities) and relationships (e.g. a link between two assets operating in the same logical space, or two sub-parts of the same device). The *representativeness* should be supported by a model defining how the DT maps the physical world concerning a target context in which to operate (denoted as *contextualisation* property, more about it later). In case the usage scenario of the DT is a specific environment (e.g., a production line), most likely only a subset of all the features, properties and information of the physical object are relevant to qualify the twin in the target digital world. Multiple representations and DT instances of the same

Physical Asset can be defined in the same environment focusing on mapping different physical aspects in relationship to the context and application goals. For example, a robot manipulator can be digitalised at the same time by two distinct DT instances, one focusing only on robot joints coordinates to build a synchronised 3D replica and the other one responsible for monitoring task execution, collecting sensor telemetry data and detecting anomalies. The physical and digital counterparts mutually cooperate through a *bidirectional synchronisation* (also named shadowing or mirroring) meant to support the original capabilities of the mirrored device, while enabling and augmenting features and functionalities directly on the digital replica, both for monitoring and *control*. DTs consequently allow external services to design new cyber-physical behaviours and to execute high-level policies without directly handling the complexity of end devices.

Connected to the concept of *representativeness*, [84] describes the *entanglement*. Entanglement refers to *the level of “adhesion” digital replica towards its physical counterpart*. In other words, it can be considered as the level of *fidelity* owned by the digital replica of what is going on in the physical world. In Minerva’s perspective, entanglement can be characterised by 3 properties: *connectivity*, *promptness*, and *association*. Connectivity refers to the ability of the DT to directly or indirectly communicate with the physical object. The more “direct” is the communication, the higher the entanglement. Promptness considers the time frame necessary for the DT to receive physical data updates and reflect eventual actions in the physical domain. DTs are considered entities that, being *living twins* of physical objects, must reflect their changes and have to influence physical state-of-affairs in a close-to-real-time matter. Association regards the *information communication flow* between the DT and the physical entity. Indeed, communication directly between the

physical and the logical entity is always necessary to have a digital representation of a physical object. Nevertheless, the DT can also send actions (or their requests) towards the physical domain, making the communication bi-directional and obtaining, therefore, a higher level of entanglement.

Given the reported characteristics, they gain more value when the concept of *DT model* is introduced. The concept of model is mentioned in [84], as well as other articles, as [103] or [49]. The *model* can be considered as the implementation of the DT behaviour. The behaviour can affect only DT ingested data, offering a manipulated, or *augmented* version of them, can somehow interact with other physical entities software entities, or can actively affect the physical state-of-affairs through its physical object or collaboration with other DTs. The role of the model is crucial in the DT definition, as characterises the DT as an *active* entity, capable of *reasoning*, and eventually *decision making*. Since the DT must reflect its physical counterpart nature to some extent, eventually through its entire life-cycle as envisioned by Grieves, each model equipping a DT is specifically designed for its purpose in the context of the physical object that the DT represents, concerning its life-cycle stage.

The introduced concept of the model related to some or all aspects of the digitised physical counterparts introduces the last feature expected by DTs that can be found in the research literature, that is *simulation* capabilities. Simulation can be considered as a model *specialised in simulation tasks*, whose is to give some insights about the trajectory that the system will take over time. Simulation purposes can range from anticipating a future physical state or happening that will affect the physical counterpart, to sharing the gained information based on the DT history with other DTs, software systems or stakeholders. Connected to the concept of simulation there is the

concept of prediction, which can be also achieved through the application of AI techniques, that will be, as a consequence, part of the models available to DTs to achieve the set of targets it was designed to in its particular context.

Over time, especially in the last years, the DT has become a hot topic and different approaches have been proposed in the literature. In the tentative of simplify and order its definition, given the major contributions reported here in the paragraph, in this dissertation DTs are considered as an approach to digitising physical assets in a cross-collaborative manner. Digitisation methods can range different approaches and areas of applications. The one considered in this thesis regards IoT-oriented DTs, in the area of industrial and manufacturing systems. In particular, DTs digitise associated physical assets state across their entire life-cycle in real-time or close-to-real-time. Physical assets state reflected in the associated DTs includes assets properties, conditions, relationships and behaviour (s), through the use of assets data and associated models. Considered models can range from identity functions, simply mirroring the underlying physical sensor state in the digital domain, to more complex sets of operations aimed to augment data into higher-level information characterising the DT state. DT models can also be complex AI or simulation models, with the target of enabling predictions or what-if analysis based on DT historical data. Moreover, DT models are not limited to incoming data manipulation, but can also have an impact on the physical context state sending a single or a set of action requests to obtain a desired state of affairs. A single DT can be finally characterised by a single or a set of models, concerning its initial design purpose, its state at the given moment and with respect to its entire life cycle.

2.4 Digital Twins Applications

Ref.	GA	R	VR	I	SC	V	PLM	L	Description
[25]		x		x					DT implementation in a MES equipped lab.
[7]	x								Open source DT web based server.
[36]	x								Conceptual modelling framework for DTs.
[58]		x		x					Review of DT for digital factories, research paths.
[84]		x	x	x	x			x	DTs in the IoT context, features and application scenarios.
[106]		x		x					Cognitive industrial DT with ontology and Knowledge Graphs.
[32]				x					Enhanced Cognitive Twin for Smart Factory.
[1]	x								Architecture for Hybrid and Cognitive Twin.
[104]	x								Real-time batch processing pipelines for DT models.
[115]	x								DT for IoT large scale behaviour analysis.
[117]	x								Ontologies to represent DTs for CPS and embedded systems.
[28]	x	x							DTs expected properties, realisation, operations, evaluation.
[63]	x		x						Robot fish DT for tests in VR environment.
[20]					x	x			Shared twinned model of learned driving style of car drivers.
[101]	x								DTs applied with new 5G and MEC technologies.
[105]					x				Modern infrastructure architectures for Smart Cities.
[64]	x			x		x			DT model for ships twinning and FOC predictions.
[118]				x			x		DTs for integrating products data across their entire life cycle.
[122]	x			x					DT shop-floor key components models.
[134]				x				x	Production DT predicting end of job for logistics optimisation.
[127]				x					Multi-modal data acquisition approaches for DTs in CPPS.
[39]				x					Data-driven industrial DT simulation models generation.
[128]				x					Wearable with edge computing synchronising factory DTs.
[12]	x			x					Modelling framework with CBR for DTs models generalisation.

Acronyms:

- GA: General architecture
- R: Review
- VR: Virtual Reality
- I: Industrial
- SC: Smart City
- V: Vehicles
- PLM: Product Life-cycle Management
- L: Logistic

Table 2.3: Summary of related works about application of DTs in industrial scenarios.

Recently, different applications and frameworks have been proposed to support DTs in the context of product design, simulation, manufacturing and augmented reality. The proposed architectures have been reasonably designed targeting a specific application domain and without providing an interoperable approach where multiple DTs can coexist and cooperate [25].

During the last few years, new platforms have been introduced (e.g., Microsoft Azure DTs¹ and Eclipse Ditto²) to try to effectively connect the physical and the cyber world through DTs and a shared set of “as-a-service” APIs and functionalities. Unfortunately, in such references and also within some of the recent research activities [7] [36] DTs have been mainly exploited as data structure repositories, rather than active software components with an internal behaviour and a set of core responsibilities to handle the interaction with the physical world. The responsibility to communicate with the physical counterpart to collect data and send commands is often delegated to external applications and typically through platform-specific services (e.g., APIs or SDKs) instead of being a specific core responsibility of each DT. This aspect impacts the digitisation process, which in state-of-the-art platforms results in somewhat fragmented across several modules and strongly limits the envisioned and desired autonomy of DTs [58].

This challenging evolution calls for new software architectures and approaches featuring levels of flexibility beyond the ones provided by the solutions available in current state-of-the-art approaches. As reviewed and pointed out also in [84], the literature is conceptually aligned on the idea of adopting DTs in multiple fields but each model is still built from scratch without common methodologies and with the concrete risk to generate a strong vendor lock-in and to block the creation of a real open ecosystem where physical assets (PAs), DTs and applications can cooperate through an effective service continuum.

In this context, intending to integrate DTs with cognitive, intelligent and analytical solutions, some works in the literature propose the adoption of

¹<https://azure.microsoft.com/en-us/products/digital-twins/>

²<https://www.eclipse.org/ditto/>

semantic models and technologies to extract knowledge from data to enable twins to autonomously perform some intelligent tasks within the context of the PA [106] [32] [1]. Furthermore, DTs become interesting for a plethora of specific approaches related to data analytics [104], behavioural modelling [115] or ontology definition [117].

Moving towards more specific applications, DTs are nowadays a high-interest topic in different scenarios [28]. They are employed in many areas, e.g., biology, automotive, smart cities and manufacturing. On the biology side, the works proposed in [63], VR is proposed as a solution for robotic systems where testing scenarios are difficult to set up and control. In particular, the article proposed a virtual representation of a physical robotic fish made for monitoring real schooling fishes, because of the difficulties encountered in testing some features of the physical counterpart immersed in its future environment, due to the lack of fish to work with. Practical benefits in system setups made on the fly have been gained, overcoming as a consequence practical barriers. Among the automotive industry, an example of a DT-based solution is the one proposed in [20]. Authors argue that DT models can be used to obtain driver's behavioural models, to gain the ability to make predictions about driver's near-to-future actions. Set the model, the driver's DT characterised by its internal behavioural model can then be shared in the geographically close cars community, to then implement a prevision system of neighbouring drivers' actions and, as a consequence, car operations. Solutions like the one described in the article can impact drivers' activities minimising hazardous situations in a close-to-real-time manner. Smart city applications are also starting to emerge, as preliminary DTs, 5G, and Multi-Access Edge Computing technologies tests carried out in [101]. The reported architecture gains various communication improvements concerning

traditional solutions, in terms of ultra-low latency, and reliable and responsive connectivity. Such a solution enables a set of possible applications in the area of DT applied to the city mobility previously impossible, as the one described in [105].

In the vehicular context, cars are not the only type of vehicles targeted by DTs and their capabilities. Indeed, a role is also filled by resource-hungry vehicles as ships and associated operations [64]. In the mentioned work, a data acquisition and analysis pipeline is proposed and described, tagging the resulting platform as a “prototype version of a virtual replica of the en-route vessel”. The platform is characterised by a series of ML and AI aimed to estimate fuel and oil consumption utilising a reduced-sized feature set, which allows for predicting the vessel’s main engine rotational speed. The whole system has the final goal of mitigating emissions from an environmentally friendly perspective.

In the context of production-product interaction, DTs open up new opportunities, as already envisioned by the initial definitions made by Grieves. On the *product* side, they enable the opportunity for interaction between the product and the associated production system through the active mean of their twins. Such an interaction can start even before the actual crafting of the actual product, with design and prototype feasibility studies enabled by DT-based simulations as well as production facility capacity simulations, to understand the actual ability of the existing system to correctly respond to actual (or predicted) market requests. Looking at the enabled opportunities under a more general scope of Product Life-cycle Management, an interesting proposal is made in [118]. The article, in particular, proposes a DT-driven method for product design, manufacturing, operation and disposal

DTs application also opens new opportunities on the *product* side, en-

abling the possibility of interaction between the product and the production system twins. The work in [118] presents a method of product design, manufacturing and services driven by DTs. The reason for the study lies in the fragmented nature of data associated with products through their life-cycle, a problem that can be strongly mitigated through the application of digitised entities of product counterparts, containing data and models of the associated physical object. In this view, the role of the manufacturing process is mostly focused on the first part of the product life-cycle during the actual production process, from the product concepts to its manufacturing and delivery, strengthening also the after-sales services. Nevertheless, the enabled ability can be applied both to products that are delivered to *final customers*, as well as products that take part in other *production systems* or that are *sub-components* of a bigger product. Under this perspective, the PLM point of view gains a high-value level, as both producers, as well as customers in the role of business owners, can leverage a data-driven-based control of their sold or used assets.

Within smart factories, DTs have already been envisioned as a key brick of shop-floor environments, and several works have been carried out in different specific research areas. Concepts proposed in [122] describe the role that DTs can have in a smart factory: among their responsibility, there is data collection and ordering from existing equipment, data sharing through the means of the Internet infrastructure, data analysis and implementation for realistic planning provisions that take into account, among other things, customer orders, order priority, and work centres average performance. Moreover, the article also envisions the use of DTs as a support for planning validation through its expected simulation and prediction capabilities, to obtain affordable planning outputs.

An application in the logistics area of a production system is the topic proposed in [134]. The target of the work is to allocate logistic equipment optimally. In this context, the proposed solutions are based on a DT of the shop floor which *mirrors* the physical world state-of-affairs. The digital mirror is represented by shop-floor Key Performance Indicators (KPIs), that are extracted through a data analysis pipeline. The pipeline cleans and extracts the intended KPIs representing the state from shop-floor data. On top of this representation, based on extracted KPIs, a task-end prediction system based on a time-weighted multiple linear regression method is built to optimally allocate logistic equipment. With this solution, the system gains the ability to proactively allocate material handling missions, exploiting a forecasting module to anticipate the logistic service call and minimise bottlenecks generated by full output buffers.

[127] instead underlines the importance of multi-modal data acquisition, as data is still a requirement to properly represent the digital counterpart like its physical twin. The paper describes difficulties in implementing data acquisition in existing industries, thus proposing a solution suitable for SMEs. The solution exploits sensor-based tracking and machine vision data acquisition, to track different process parameters as well as operators and forklifts. Another proposal from [39] proposes a data-driven approach to infer the shop-floor composition from the data it generates. The need of having such a system is raised by the increasing customer demands challenges and shorter product life-cycles which guide a continuous factory transition in the shop-floor organisation, while the opportunity is provided by the abundance of data usually present in a production environment, recently fostered by the recent high availability of modern sensor solutions. Data extracted from the shop-floor logs are collected by DTs and analysed via ML and process

mining techniques to build a Petri-Net, i.e. a structured representation of the actual production process. For demonstration purposes, a Petri-Net of a quad-copter sub-assembly is first set, and then extracted from data with the aforementioned techniques. Finally, the extracted Petri-net is used as a base for a simulation to study the assembly system reliability, studying its "fail" and "repair" transitions.

In the industrial context, the DT assumes different roles concerning the actual problem to solve. AR is a technology expected to mitigate some future challenges that blue collars will face due to the trend of mass customisation [128]. Nevertheless, authors recognise also the limited computational power offered by such devices, needing therefore an optimised approach to carry out properly their tasks. The solution proposed considers wearable devices working in parallel with edge devices on the shop floor, supported by modular software powered by shop-floor equipment DTs combined with ML algorithms for object recognition activities. In this context, for example, DTs act as representers of the real-world composition, enabling the described opportunity in the application of AR on manufacturing shop floors.

Authors in [12] consider DTs as digital entities able to represent, control and predict CPPSs. Considering the long-living nature of considered systems, mixed with the different environments they can be surrounded, system behaviours might diverge from designed targets, relying on domain experts for tricks and tweaks necessary to gain expected system behavioural results. To boost up predicting capabilities of CPPSs, DTs have been employed to learn from human experts the necessary CPPSs setups to address unforeseen challenges, facilitating as a consequence system self-adaptation to different surroundings. To enable the described self-adaptation of the physical system, its twin has been equipped with a model based on the Case-Based Reasoning

(CBR) technique, with the target of learning highly specific sparse domain expertise and reacting to upcoming, never-seen problems.

Looking at the actual literature shows that DT research application suffers a little from the siloed approach mentioned in the first section of this Paragraph. Examples of that are the works mentioned in [63] [64] [134] [128]. By the way, more generally applicable systems are also studied, as proposals in [118] [122] [127] [39] [12]. To this regard, the approach described in the following dissertation considers the manufacturing contexts as IoT-oriented, studying therefore applications principles of DT in the associated IoT scenario. Such an approach is considered as DTs are envisioned to be not only a way to digitise physical assets but also as an abstraction layer to pose between physical objects and digital applications that has the added function of *generalise* the physical complexity of the underlying system, i.e. to *abstract* it. Indeed, in a manufacturing environment (but rationally, in any kind of environment), it does not matter if the milling machine of brand A has a certain type of software concerning the same kind of machine of brand B. What matters is that they can be described with a set of *attributes* and *capabilities*, that will be reflected in the digital domain and abstracted by associated DTs. In this way, only needed physical asset characteristics will be transposed, leaving to the DT structure and model the responsibility to describe the associated physical object peculiarities.

2.5 Intelligence in Industry

Ref.	FP	OP	M	CL	SPA	I4.0	Description
[138]	x						Oil production forecast.
[2]	x				x		Future spare parts demand forecast for production planning.
[37]	x	x				x	WIP time left prediction in a job-shop environment.
[114]		x				x	Flow control WIP levelling through RL.
[73]		x				x	Integration welding systems through IoT.
[16]	x		x			x	Review of ML methods in the field of Predictive Maintenance.
[29]	x		x			x	Review of ML and reasoning for Predictive maintenance.
[148]	x		x			x	Time to Failure prediction for industry with automatic ML pipeline.
[47]	x		x		x	x	Prediction of time window opportunities for maintenance activities.
[34]			x			x	TPM AI-based Semi-Markow-Decision-Processes solving.
[94]			x			x	Prescription for Maintenance activities.
[131]				x		x	Review of manufacturing causal discovery.
[6]				x			xAI basic concepts, taxonomy, review.
[69]					x	x	AI-based rescheduling optimisation algorithm for job-shop.
[52]					x	x	Q-Learning based flow shop scheduling.
[135]					x	x	MAS Q-Learning adaptive scheduling.
[70]					x	x	DQN edge scheduling.
[133]					x	x	MAS real-time scheduling.
[41]					x	x	Two-agents deteriorating scheduling.
[60]						x	Review about AI applications in Industry 4.0.

Acronyms:

- FP: Forecast/Prediction
- OP: Operations Control
- M.: Maintenance
- CL: Causality Learning
- SPA.: Scheduling of Production Activities
- I4.0.: Industry 4.0

Table 2.4: Summary of works related to the application of AI in industrial scenarios.

AI finds a fertile application ground in industrial scenarios. The abundance of data, necessary information, and metrics that need to be mapped and mixed to extract highly valuable insights makes AI a powerful tool for doing so. Areas of application are multiple, with different targets. Among them, the most straightforward regards forecast activities. Indeed, predicting the future puts industrial systems in a very powerful position, with the ability to plan activities in function of the prediction. A quiet old example, proof of the long-lasting application of AI techniques in the industrial do-

main, is the system described in [138], which applied fuzzy logic and neural networks to forecast oil production.

Moving towards more recent applications, demand forecast for planning activities has been studied in [2]. The application scenario involves a construction machinery company, whose problem is to predict the future spare part requests for each customer minimising the prediction error. To implement the system, several AI tools have been applied as multiple linear regression, multiple non-linear regression, artificial neural networks and support vector regression.

Future forecasting is a problem applied not only to expected market requests but also to the operations management area. Is possible to consider this area as the mix of tools and techniques aimed at having control of actual activities at the shop-floor level. A job remaining time forecast is proposed in [37], where the project result is then applied to a 44-machine shop floor producing 13 different types of parts. The value of the reported work lies in the intrinsic difference between production scheduling outputs and their effective actualisation, which encounter very often external disturbances. Gaining the ability to predict the remaining time of a job analysing data gained directly from the shop floor enables the opportunity for smart-decision making close-to-real time, as triggering new online activity scheduling or taking other decisions on the go.

Operations control is another key activity in the area of operations management, typically involving the choice of the best control strategy concerning the actual scenario. Common strategies share the idea of minimising the circulating WIP material, capping its maximum value for the considered system. By the way, considered limits can change over time, changing production conditions of the system, making the WIP limits no more suitable

for the upcoming production context. Making WIP limits dynamic overtime periods is what is proposed in [114] through the use of a Reinforcement Learning approach. Results show the benefit of the introduced system, as the WIP level has been reduced up to 43% without any performance degradation.

Managing operations in robotised systems through the help of AI is a research topic too, as reported in [73]. The article analyses the potential application of modern IT pillars of the fourth industrial revolution (i.e. IoT, Big Data and AI) to modern welding systems, with the target of improving quality predictions, control methods, and overall production improvement. Experiments involve Multi-Agent Systems to gain a distributed intelligent environment with coordination capabilities, solving the problem of integrating different sub-apparatus composing the overall production tool, and putting intelligence inside of each of the analysed components.

Another common area of application of AI in industries is Maintenance. The reason for such interest is to gain control of a problem that has stochastic characteristics both in its occurrence and associated repairing activities, minimising production downtimes, planning related activities with a smart approach, and optimising the overall maintenance costs [16] [29]. In this regard, time-to-failure prediction is the very first activity needed to be carried out by practitioners to organise consequently associated activities [148]. In scenarios that involve costly equipment, where there is the need to run them as much as possible to maximise the extracted value from them, minimising maintenance-associated activities and where human monitoring is difficult or even impossible, time-to-failure prediction based on machine sensors can be an effective solution. Gained the ability to predict future machine failures, organising associated activities becomes the next step to solve. A proposal in that direction is made in [47], leveraging the forecast of future machine starv-

ing or blocking conditions due to upstream or downstream not-correlated problems.

A more holistic approach considers existing production maintenance frameworks as Total Productive Maintenance or TPM [34]. In this regard, the Reinforcement Learning techniques become useful to optimise TPM-associated optimisation models, which are effective in small case scenarios, but become problematic when they grow in size.

State-of-the-art approaches try to make a further step in maintenance activities, building AI-based tools that not only have the ability to predict future problems but also to suggest possible solutions [94]. This is the field of *prescriptive* maintenance, which can close the circle of needed solutions in the maintenance area, making prescription and associated predicted time necessary to complete maintenance tasks useful information to maximise maintenance activities.

However, even if different research areas using AI in industry exist, some authors also highlight that an extensive application of AI techniques finds some barriers, due to a lack of fairness, accountability and transparency of state-of-the-art AI tools. Causal discovery is recognised as the future candidate AI technology, both in the industrial field [131] as well as more in general in the AI domain. Causal discovery is considered as part of eXplainable AI, a research topic aimed to overcome actual AI limitations such as the lack of transparency, with the target to make AI models *transparent*, i.e. to let users understand *why* a certain AI model returned a certain output instead of another [6]. Application of such concepts in the industrial domain can bring lots of benefits, from effectively understanding reasons why a certain happening is affecting the maintenance of production equipment, to cutting down general root-cause-analysis tasks. More about that topic will

be presented in Section 5.1.

The final worth mentioning in the industrial research area where AI is extensively applied is scheduling. Scheduling regards the activity of assigning operational tasks to production nodes on the shop floor according to market production requests. This kind of activity can be considered easy for scenarios involving simple production systems, e.g. when there are only a few products to craft and the system is mainly automated. Nevertheless, systems can easily become tough to manage under this lens, due to the ever-growing product customisation requests brought to complex production systems (more on the topic is provided in Paragraph 3.1) as well as for the np-hard nature of the problem. The trade-off between rescheduling activities and their frequency has been studied in [69], where Machine Learning (ML) techniques have been utilised in conjunction with optimisation techniques in a flexible-job-shop scenario. Other scheduling-related problems have been tackled with the reinforcement learning approach, training smart agents to take the best scheduling choice concerning the actual production situation [52] [135] [70]. Agents and Multi-Agent Systems have also been considered, with different modelling approaches. A bargaining-game-based negotiation mechanism has been applied in [133] as a coordination system, while a deteriorating scheduling approach has been proposed in [41], with two agents in the system. The first target of the agent is to minimise the make-span of the solution, while the second has the target of minimising the total tardiness.

The abundance of data as well as the very challenging problems presented in this paragraph, make production environments a very good application field for AI and associated techniques, which are extensively researched with a plethora of approaches. Nevertheless, lots of challenges are still present, most of which are due to a lack of a comprehensive representation of indus-

trial systems, making them fragmented, and difficult to manage from an IT perspective, given the multitude of hardware, associated software, data structure and consequent difficulties in standardise data representations. Looks like those challenges can not be completely solved with AI, instead, AI can strongly benefit from their resolution. Among candidate solutions, the one that researchers mostly agree upon is DTs [60].

2.6 AI and DTs: Synergies Over Industrial Scenarios

Ref.	DT	AI	I	L	V	Description
[134]	x	x	x	x		DT model of a production predicting end of job for logistics optimisation.
[64]	x	x	x		x	DT model for ships twinning and FOC predictions.
[102]	x	x	x			Production control with DT, CONWIP, and Q-learning based predictions.
[39]	x	x	x			Generic data-driven simulation models generation for industrial DTs.
[8]	x	x	x			DTs for real-time optimisation of robot motion planning.
[142]	x	x	x			DT for emissions predictions for an industrial system.
[125]	x		x			DT driven dynamic scheduling of an hybrid flow shop.
[79]	x					Agents and DTs similarities and differences exploration.
[60]		x	x			AI applications for Industry 4.0.

Acronyms:
- DT: Digital Twin
- AI: Artificial Intelligence
- I: Industrial
- L: Logistic
- V: Vehicles

Table 2.5: Summary of related works considering both AI and DTs in the industrial domain.

In the previous sections (Section 2.5 and 2.4) has been depicted the importance of AI as well as the role of DTs in the industrial domain. Several researches and applications have been proposed, to solve specific problems as well as in the tentative of building a generalised framework to obtain a further step in the technology application. Because of the importance of both technologies in the industrial setting, mixing in their application is highly expected in the domain, and some articles proposed already applied both of them to obtain targeted results. Some examples are [134] or [64]: in the former, AI has been used leveraging information gained from a digital representation of the shop-floor to perform time prediction activities; in the latter, the whole data ingestion and analysis pipeline for ship-representation constitutes the DT of the physical asset, with AI tools embedded in it for

prediction activities. A similar proposal is the one described in [102]: the target of the article is to improve existing standard Constant Work-In-Process (CONWIP) workload control models through the usage of DTs containing Reinforcement Learning models. The model goal is to perform short-term forecasts about the behaviour of the system, enabling what-if analysis for the employment of different numbers of cards for workload balance. AI has been also used to perform the actual DT creation, as reported in the following already mentioned article [39]. Other two areas of application in the industrial domain can greatly benefit from the mix of the two technologies: energy consumption reduction and optimisation of automated equipment [8] as well as carbon emission prediction [142]. Production scheduling tasks have been studied in a lot of different manufacturing scenarios, providing value at all types of levels. It is a very important activity both for production systems with broad product portfolios as well as for commodities producers, because of the ever greater importance of activities functional to the production as logistics or maintenance. To this regard, the application of AI already made its course, but adding DTs to the mix can open up new application scenarios and scheduling strategies, based on standard tasks activities as well as unpredictable happenings [125]. In this regard, the conjunction use of DT and AI enables both prediction and simulation activities, bringing new value and research challenges in the field of scheduling and optimisation. The massive use of MAS systems in the scheduling research area can also open new research options, given the existing affinity between Agents systems and DTs [79].

Is clear how the mutual usage of AI and DT technologies are bringing great expediencies in the sector, and how they are deeply studied to bring industrial improvements at all levels. The particular importance of DTs in the

application of AI technology in manufacturing is explained in [60]. Indeed, the article describes how DTs are considered by most actors as the foundation technology for effective industry digitisation, integration and consequent smart application of AI. Moreover, the article states that the adoption of DTs in the industry will also accelerate the transition towards the Industry 4.0 paradigm, as it makes easier and seamless the integration of various genres of hardware without any kind of operations interruption. In particular, the AI-DT relationship can be considered also in the following way: DTs, being a way for digitising physical objects, become the logical constitution element for industrial shop-floor, with the ability to represent and contain all the information of each shop-floor physical asset, following the logic representation present in the real world. It means that all the information about a defined machine, for example, will be collected, stored and therefore found in that machine DT. If instead DTs are specialised concerning a certain aspect of a physical asset, therefore information about that aspect of the associated machine will be contained in its associated DT. In that regard, DTs not only represent physical objects but also associated physical concepts, e.g., shop-floor department information. The result can be achieved by composing DTs of interest in a bigger DT entity, which is fed by the composing DTs. In this scenario, representation capabilities are still respected, offering therefore an interesting and convenient abstraction of the physical world in the digital domain. Abstraction capabilities of DT also enable the horizontal integration of assets of different constructors, because of the generalised way of communication of DTs through the use of different communication adapters, and because of the uniform representation of physical assets in the digital domain characterised by its properties, conditions, relationships and behaviour(s), through the use of assets data and associated models (as de-

scribed at the end of 2.3). Nevertheless, the uniform representation does not limit the adherence of the twin to its physical counterpart, thanks to the usage of its associated models which characterise each twin concerning its physical counterpart and the design purpose. All described ingredients pave the ability of DTs and their designers to organise and extract all the necessary information of associated physical assets most conveniently, also in the eventual need of employing AI models. DT and AI relationship lays therefore in the clear division of responsibilities between the two technologies: the former communicates, represents, abstracts and orders data associated with single and collections of physical counterparts, while the latter brings intelligence as a tool at different levels and for different purposes starting from the world state-of-affairs represented by contextual DTs. In the investigated vision, AI techniques can serve different purposes: as intelligence internal to DTs, as prediction models of DTs' actual state, as simulation enablers for what-if analysis of DTs, etc etc. Nevertheless, AI abilities can also be placed *outside* of DTs, acting as external entities that somehow manipulate or push action requests towards the DT architecture system.

As the reader can understand, given the proposed set and division of responsibilities, a clear division of tasks emerges autonomously from the context, opening up a set of different opportunities in the industrial sector. Possible interaction patterns between the AI and DTs are one of the key topics of this dissertation and will be deeply discussed in Chapter 4

2.7 Industry 4.0: Not Yet Enough

Industrial management tools and management techniques have deep roots in industrial history. First of them arose with the will to give a rational

and scientific organisation to industrial production complexity, starting from forced simplification choices, as the one made by Henry Ford for the Model-T production, passing through the implementation of the best industrial architecture concerning the nature of the product to craft, terminating with the modern Lean Manufacturing production modern management frameworks. Nevertheless, the rising competitive pressure of modern markets as well as the increasing demand for customised products produced in massive volumes brought out the actual limitations of existing approaches, demanding a smart implementation of systems capable of transposing and abstracting existing production systems with the will of leverage, when necessary the power of modern IT instruments, as AI.

In this regard, several AI applications have already been made in the sector, most of which the tentative to solve very specific problems. A lack of a large adoption of modern IT techniques in the industry that fall under the umbrella of Industry 4.0 highlights the need for a more general and standardised approach to the problem. Indeed, standardisation and scale application of production has always been the power of the industrial sector, and nowadays a highly customised approach to industry standardisation is not bringing desired results.

Many experts point out DTs as the foundation technology that the industry needs to adopt to overcome existing limitations. Indeed, DTs can abstract and transpose existing production systems into the digital domain without losing associated key information. DTs can therefore be considered as a way of digitising existing physical objects actively, i.e. integrating into the digital counterpart one or more behavioural models that give the DT the ability to augment physical counterpart abilities and accept action requests to pass to their associated objects.

In this vision, DTs solve and manage the difficulties of digitising physical assets, coordinate over the communication protocol, structure a representation of the physical counterpart, store its behavioural model, manage the exposition of resulting information externally and administrate incoming action requests.

The division of responsibilities between DTs and AI becomes then clear, opening up new application opportunities in the context. In the following Chapter a deeper description of existing relationships between DTs and the industrial domain is given, describing problems tackled by DTs in the sector, their role, useful features, employable architectures and associated advantages.

Chapter 3

Digital Twins: a Solution for Industrial Complexity?

During industrial history, production complexity has always been a challenge to face. From the first implementation of assembly lines to modern lean manufacturing frameworks and Industry 4.0 implementations, one of the targets was to lower and/or manage the complexity to gain control of the production system and its happenings. The mass production approach as well as the first major rationalisation of activities have been enough until a new season of mass customisation emerged from the market, posing new challenges to the sector. In the following an overview about the theme of complexity in the manufacturing domain is given, proposing a contribution about a new source of industrial complexity impacting the complexity in the target domain. Then, the DT modelling section is provided, where a set of DTs useful features in abstracting the industrial physical and dynamical complexity towards the digital domain are proposed.

3.1 Complexity in the Industrial Environment

Industries have become increasingly complex production entities. It is widely accepted among the scientific community how complexity is tackling more and more manufacturing and associated activities. Authors are generally aligned about reasons of this growing complexity, as the globalised market, the consequent rise in competitiveness, requests of the market for high-value, highly customised products at competitive prices, consequent demand for a wider product variety, and shorter products life-cycles [33][85] [87] [48] [96] [107]. As a consequence, production systems involve technologies of various types, of new and old generations, with realities where machines have not replaced humans and where their transversal integration is increasingly intricate. Therefore, it is possible to observe that to be able to respond to depicted challenges, the complexity of the market demand has been reflected in complex production systems at the layout and equipment level [107]. As a result, practitioners can conclude that a certain level of industrial complexity is needed to respond to market requirements, and managing such a complex system has become an important aspect to cope with, which moreover can lead to a competitive advantage in the global market challenge [66].

In [33] the concept of complexity is analysed at 360 degrees. Different complexity descriptions and associated contexts have been given, e.g. computational complexity in computer science, complexity in engineering, product design and life-cycle management, manufacturing, information theory and others. Nevertheless, the authors recognise that the word “complexity” is tricky to pin down. Indeed, they report that there is no single, clear-cut definition that everyone agrees on the original Latin word “complexus” means “intertwined” or “joined together”, while the reported definition from the Oxford Dictionary depicts complex as something made of (usually several)

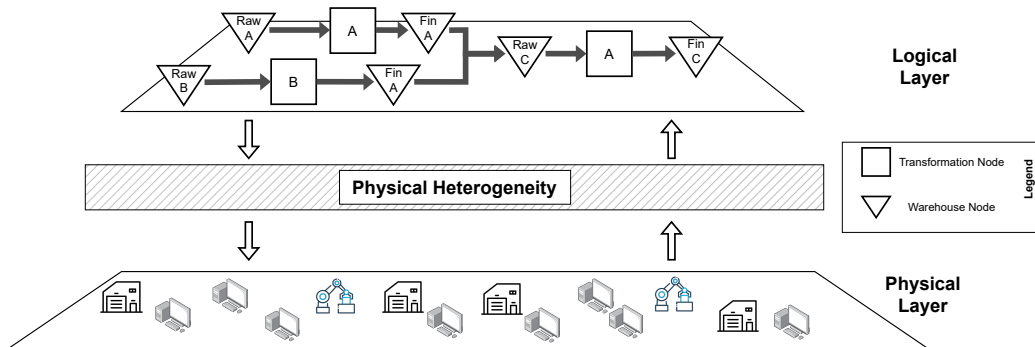


Figure 3.1: Given the physical complexity and heterogeneity, when it is reflected in the logical domain, the entire set of complications is transposed, making the logical representation intricate as well.

closely connected parts. A first conclusion that is possible to take considering reported definitions is that *the more parts and connections a system has, the more complex it is*. Contributors of [33] dive a little more into the complexity arena, distinguishing other two concepts closed to complexity: *complicatedness* and *chaos*. A system differentiates itself from being simple or complicated by the ease of knowing it: a simple system is easily knowable, while a complicated one is not. Instead, a system is highlighted to become complex when uncertainty takes part in it, making it somehow not fully predictable. By the way, complexity and complicatedness are also affected by the capacity of a single individual of *understand* it, even with the help of technology. Chaotic systems, instead, differentiate themselves from complex, complicated and simple systems because of their very different outcomes given small changes in initial conditions. They are very difficult to manage and also to predict, because of their chaotic nature. The growing complexity in manufacturing is also introduced and studied. Among different complexity definitions, authors of [33] report that the real (or perceived)

level of complexity of engineered products and the associated design and manufacturing process, is connected to the amount of information needed to be processed. Therefore higher customisation requests of the market lead to a higher number of variants of products, leading to a growing amount of information that needs to be processed, indeed. Interestingly, the article reports how highly automated environments result in a lower amount of needed operators, but in a complex highly integrated and intricate system. Costs associated with designing, implementing, controlling and maintaining the whole business system grow accordingly. Therefore, given the economic advantage of being able to operate with a lower amount of personnel, managing complexity emerges as a need to mitigate associated costs and confirm the gained competitive advantage.

Static complexity

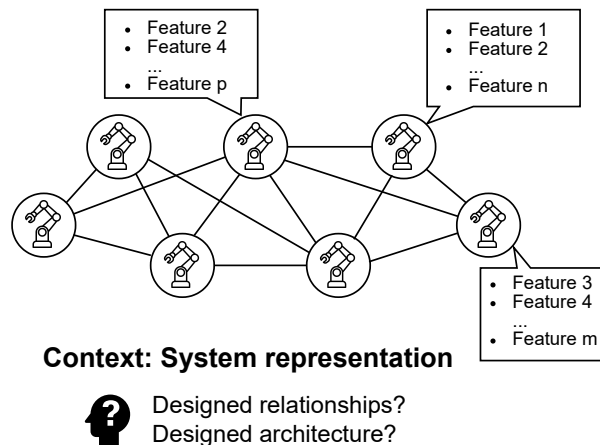


Figure 3.2: Static complexity considers the complexity of the system in a time-independent manner. Complexity level is therefore affected only by the system structure and associated features

Diving into the nature of industrial complexity, 2 types of complexities

Dynamic complexity

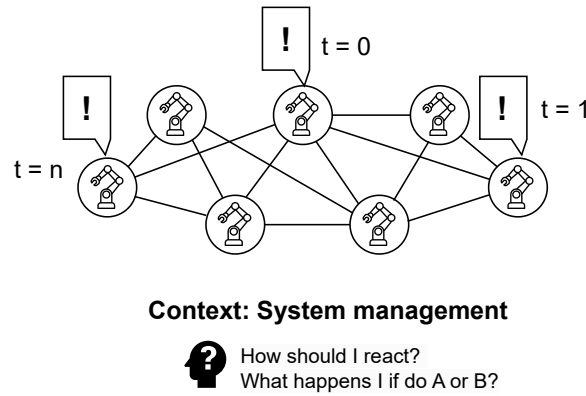
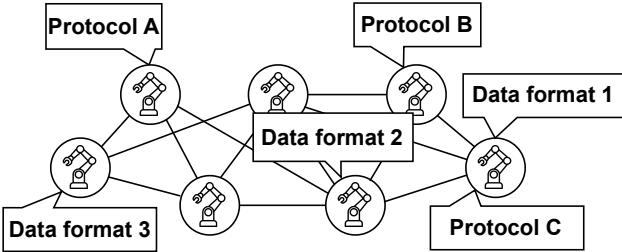


Figure 3.3: Dynamic complexity describes the complexity of the system at runtime. The complexity level is time-dependent and is affected by the amount of different expected and unexpected events emerging when the system is running.

have been recognised in considered environments [33]: *static complexity* and *dynamic complexity*. Static complexity, depicted in Figure 3.2, deals with the product and system structure in a time-independent manner, considering the amount of information necessary to describe the state of a system. The static complexity may be reduced by simplifying the product design and/or the associated production process. Dynamic complexity is defined as time-dependent and is also related to real-time operations activities and deviations from the normal/steady-state, comprising uncertainties, unpredictable events, and adaptive responses (depicted in Figure 3.3,). Drivers for this kind of complexity may be internal (e.g. breakdown occurrences, maintenance policies, scheduling activities) or external (e.g. defects in received raw material or delays due to supplier unpredictable occurrence). The static and dynamic complexity of a manufacturing system are particularly important

Digital complexity



Context: System communication and data format


 How can I manage many protocols?
How can I simplify many data formats?

Figure 3.4: Digital complexity

describes the information complexity of the system. In particular, focuses on the amount of information that the system needs to properly run.

because, static complexity mainly sets the overall cost level of the production system, while dynamic complexity draws the system towards undesired states and associated costs.

Different measurement options are available to measure the described characteristics. In [85] several of them are explored and then selected as the most representative. The latter is based on the conceptual link between the number of parts, machines, and operations. The relation between these 3 elements and the difficulty in determining the economic impact of a product variant introduction, as well as the emerging uneven distribution of costs across the product variants lead to induced complexity, prompted the authors of [87] to draw up a list of Variety-induced Complexity Cost Factors (VCCFs). Based on the scientific literature, the VCCFs have been designed considering the impact of product variation both at component and end-product levels and classifying research results into 4 categories following the industrial

standard of the American Productivity and Quality Centre, to guarantee the applicability of the list among different manufacturing industries. Resulting categories of VCCFs are “Procure materials”, “Produce/manufacture/deliver product”, “Manage logistics and warehousing”, “Develop and manage sales plans”. Then, the tool has been tested in real manufacturing realities. Logistics and warehousing resources resulted in the highest value of identified VCCFs, followed by “Produce/manufacture/deliver”. Material procurement and sales process instead were classes with the lower VCCFs. In 50% of the cases, VCCFs were also quantified, mostly among logistics and material procurement processes. Instead, in other classes quantification was lower, due to a lack of available data.

Configurations resulting from a growing demand for flexibility and customisation are studied by [48], analysing the resulting complexity under an operational lens. The complexity metric adopted is the information entropy, providing a numerical description of the complexity relationship between operations and stations, considered as sub-lines and parallel stations simultaneously.

The impact of complexity on performance metrics under a wide scope is evaluated in [96]. Static complexity is evaluated in an MTO and an MTS environment, comparing the relative experiments results. About MTO, the authors demonstrate that complexity is a good predictor for the lead time, but is not the only impacting it: indeed, demand level is another variable for the lead time as well and, in conjunction with complexity, leads to fast degradation of performance. In the MTS scenario, instead, there is not a direct link between static complexity, demand levels and performance degradation, which may be due to the customer orders independent scheduling activities. However, the low values obtained in the power analysis for MTS suggest that

collected information is not enough to draw valid conclusions.

A relationship between equipment and layout complexity is highlighted in [107]. The article argues that those two features need to be balanced to obtain the system with the lowest complexity possible. A granularity complexity index is modelled accordingly, considering the impact of the layout and the equipment characteristics on overall system complexity. Results demonstrate a direct relation between layout and equipment complexity, as the simpler the equipment, the more complex the layout; the more complex the equipment, the simpler the layout. Methods to achieve the desired balance are then explored.

It is arguable that there is a direct link between static and dynamic complexity in the industrial domain, since, the more complex the system statically, the more it is prone to unexpected events and a general drift towards undesired states. The literature review proposed in [55] supports the idea toward this direction, mentioning why manufacturing systems are characterised by instability [43]: interdependence, considered as the direct dependence between one work station and another, is depicted as one of the four factors leading to production system possible unpredictability. The associated relationship between events and consequent state changes is the second resulting factor. Sources of industrial complexity are more generally reviewed in [55], and can be grouped in the structure of the product, the structure of the plant, the production planning, the flow of information between agents, departments and workstations, uncertainty bonded with variations in resources and regulations. A similar cause-effect connection is described in [85], where a conceptual link between static complexity and the number of parts, machines, and operations is reported. Instead, a profound effort towards the definition of sources of static complexity has been made in [42]. Indeed, the authors

has studied 8 types of potential static complexity sources based on *product line complexity*, *product structure* and *process complexity* components, and verified their correlation with system performance, i.e. the variation on production performance of the candidate static complexity sources. Despite the great effort, the complexity contribution given by the layout structure was neglected, due to high difficulties in comprising it in the study. Nevertheless, progress has been made, and also the layout as a source of studied complexity has been considered in the literature [107].

Looking at the sources of industrial complexity, it is possible to highlight that static and dynamic complexity sources are connected to the structure of the industrial system, and more specifically:

- static complexity depends on time-independent components of the industrial system, i.e., its product complexity and mix, its architecture, defined by the number of machines involved and their layout (or, in other words, the logic that interrelates one machine with the other), and its process, defined as the set of rules to follow to achieve the efficient production operations;
- dynamic complexity, considered only in its internal component, deals with the statically defined system at runtime, comprising uncertainties, deviations and adaptive responses. The more the system is statically complex, the more difficulties in managing it dynamically, and the higher the probability of drifting to undesired or unforeseen states.

In this dissertation, the static and the dynamic complexity of industrial systems are grouped under the more general definition of *physical complexity*, as they are considered by the community as the representation of the complexity of industrial systems strictly in the *physical domain*.

Generally speaking, three are the possible ways to deal with complexity: avoid it, reduce it or control it, as reported in [55]. To this regard, some responses towards complexity management emerged, such as lean manufacturing: in [55], lean tools are considered as complexity management instruments; [33] recognises that production systems that adopt lean manufacturing experience lower complexity levels concerning mass production equivalent systems; [42] reports that the focus of lean manufacturing is indeed to lower the system complexity, as it is considered an important management factor.

On one hand, state-of-the-art literature witnesses that definitions of industrial complexity settled in the researchers' community. On the other hand, is argued that modern technology enhancements need to be considered in the industrial complexity definitions as it is true that they may help in improving considered systems, but they may also become a new tool that needs to be managed. In this regard, this work proposes a new intersection between the complexity typical of IoT systems and its impact on industrial environments. The consideration emerges due to the new generation of connected equipment, that needs to take into account the needed mix of programming information, communication protocols, control data types and control patterns that a single machine needs to correctly perform its operations. For that reason, *digital complexity* is proposed as a new source of complexity for considered environments. In the realm of IoT and Industrial IoT, the concept of digital complexity emerges as a formidable challenge, predominantly due to the pervasive fragmentation and heterogeneity within the landscape, as depicted in Figure 3.4. This fragmentation is evidenced by the multitude of protocols and data formats existing across various IoT deployments. Such diversity not only complicates the development and deployment processes but also presents significant hurdles for achieving seamless interaction and

interoperability among disparate components [51].

The intricate web of protocols and data formats acts as a concrete barrier, impeding effective cyber-physical interaction and hindering the realisation of collaborative, distributed intelligence systems. Intelligent applications, which seek to harness the potential of the physical layer to derive value, face formidable challenges in navigating through this complexity. Without standardised and interoperable digital layers, the potential for synergistic interactions across systems remains severely limited.

Moreover, the abundance of approaches in dealing with diverse scenarios exacerbates the issue by leading to further fragmentation. Integrating these disparate approaches into a cohesive system becomes non-trivial, both conceptually and technically. The lack of clear criteria for determining the placement of intelligent functionalities within IoT systems adds another layer of complexity. The distributed nature of IoT architectures offers numerous possibilities for embedding intelligence across various computational nodes, ranging from the edge to the cloud, further complicating the decision-making process.

Digital complexity differs from industrial physical complexity because it deals with the management of the software and generally ICT technologies, representing the physical context and the associated complexity in the digital domain. Indeed, if physical complexity faces increasingly stringent market demands to have a physical production system capable of responding to requests in the simplest possible way (and therefore the least expensive possible), the digital system instead deals with abstracting the complexity and specialisation of every single piece of equipment of the associated industrial context, to obtain a system that is easy to maintain, scalable, easily expandable and with reusable elements, enabling cross-collaboration between the

physical elements of the system and promoting the respective coordination. It could be considered that industrial physical complexity affects the complexity of the digital system to some extent, for example in the multitude of data formats and protocols used by individual machines, as well as the resulting foreseen or unexpected states towards which the system can slip. Nonetheless, if the digital complexity linked to the specific concepts of the IoT domain is not managed or mitigated, the intelligent representation of the physical complexity fails, thus leading to a complex physical system linked to an even more complex digital representation, expensive and difficult to update and maintain.

Depicted challenges of modern CPPS are nowadays promoted by IoT systems closed in silos, whose integration both horizontally (i.e. with other physical objects) and vertically (i.e. with applications that must interact with the physical world) becomes difficult and, as the complexity of the physical system grows, impossible. Therefore, transferring already engineered solutions from one industrial system to another is not practicable, with software that is therefore hyper-specialised with respect to the particular mix of IoT systems and specific requirements of the application context, limiting, among other things, the general application of intelligence [60]. Finally, the lack of tools that promote standardisation in the representation of physical objects leads to the so-called “information proliferation” typical of IT systems [33] and specific to each industrial context, thus promoting a further factor of digital complexity and feeding the negative spiral of the digitally induced complexity [11].

The resulting frame of the three described industrial complexities (static, dynamic, and digital), interact one with the other with the following pattern, as depicted in Figure 3.5:

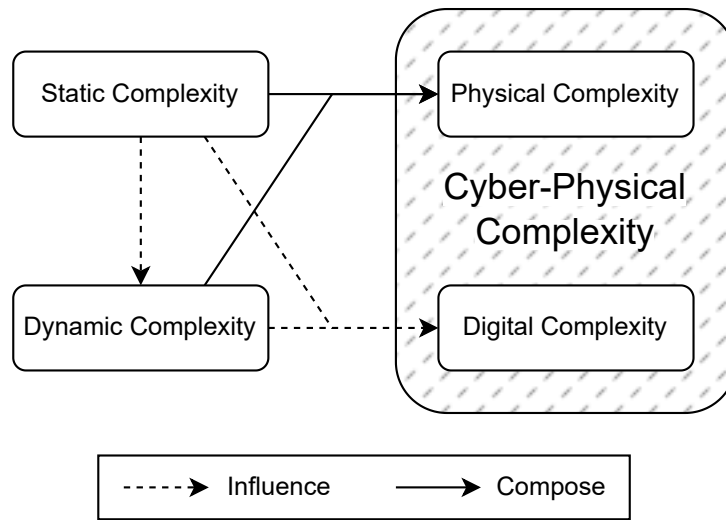


Figure 3.5: Influences between static, dynamic, and digital complexity, and the resulting definition of Cyber-Physical Complexity.

- the static complexity influences the dynamic, as the more statically complex is the system, the more it is prone to unforeseen states, uncertainties, and more in general critical events to manage;
- the static complexity influences the digital complexity as, the more the system is statically complex, the more is likely to have a heterogeneity of equipment characterised by its technological peculiarities involved in the industrial scenario, with a wider mix of products in the portfolio and a higher variety of processes and associated descriptions;
- the higher the static complexity the higher the dynamic one, which in turn affects the digital complexity because of the higher volume of events to model, catch, eventually manage and predict, comprised the unforeseen and critical ones.

Complexity concepts explored until here contribute to the overall complexity of industrial Cyber-Physical Systems (CPSs) in the physical domain,

through static and dynamic complexity, and in the digital domain, through digital complexity. It is therefore possible to say, taking into consideration an environment composed of multiple physical entities connected one with another through IoT hardware, that the physical and the digital complexity can be grouped under the concept of the more general *Cyber-Physical Complexity*, which then represents the overall complexity level of a CPS.

To address these challenges, advocating for the decoupling of intelligent applications from the inherent physical complexity is strategically imperative. By decoupling applications from the intricacies of fragmentation and heterogeneity, developers can create a continuum environment wherein applications seamlessly discover and exploit physical capabilities. This abstraction shields applications from the complexities of diverse communication protocols, enabling them to adapt and scale more efficiently.

In this challenging context, DTs emerged also as an effective solution and approach to handle and solve the digital complexity of the physical layers promising the creation of a strategic, homogeneous, and interoperable abstraction on top of the physical layer. This digital layer provides the necessary decoupling effect to handle the available heterogeneity of protocols, data formats, and interacting patterns to support different forms of intelligence coherently and offer guidance on distributing intelligence to maximise synergy. Proposing a domain-agnostic criterion for the separation of concerns facilitates informed decision-making regarding the placement of intelligence within the system. Additionally, conceptual architectures and abstract frameworks provide a unified lens for representing and modelling heterogeneous intelligent functionalities, transcending specific development approaches.

Addressing digital complexity in the context of IoT and IIoT necessitates strategic decoupling of intelligent applications from fragmentation and het-

erogeneity. By leveraging appropriate abstractions and design paradigms, with DTs, and establishing clear criteria for the distribution of intelligence, it becomes possible to navigate the intricate landscape of IoT deployments and unlock the full potential of cyber-physical systems.

The actual lack of standardisation in machine implementation added on top of existing complexity leads therefore to a never-so-high complexity level in trying to integrate different subsystems, with the unexpected paradox of not gaining competitiveness with the mass application of automation in production [33]. The three complexity components, i.e. static, dynamic and the introduced digital, contribute all together to the overall system sophistication which, as a consequence, can be framed under the concept of *Hierarchical and Ecosystem Complexity*. So, this wider level of complexity can be considered as the level of complexity obtained from:

- the level of the static complexity that a single piece of equipment can/must handle to obtain the process results,
- the associated digital complexity in terms of the amount of information needed for the single piece of equipment to properly operate, the complexity in the generated data types, the complexity in the associated communication protocols and techniques to enable cross-equipment coordination, coordination with the application level and other actors, and available operative actions it can perform,
- the dynamic complexity, as the sum of the scheduling and operations effort, the ability to promptly respond to unexpected events at a single equipment and layout level, the ability to produce expected quality goods meeting customer lead time agreements,
- the resulting difficulties in extracting data and information from a

group of machines and resources at each hierarchical level, such as a department, business unit or an entire production plant and eventually react to gained information, due to the components' complexity at a static, dynamic and technological level,

- the added management for products portfolio management, understanding the impact of adding a new product in the production loop or dismissing an existing product from production.

3.2 Modelling DT in the Industrial Context

A DT represents the *digitised software replica* of a physical asset with the responsibility to clone available resources and functionalities and to extend existing behaviours with new capabilities. For example, concerning an industrial robot a DT can mirror joint position and sensor telemetry; simplify the control of tasks and mission execution through dedicated exposed interfaces; and augment original capabilities by introducing anomaly detection functionalities to anticipate potential malfunctioning. In this context, DTs represent a fundamental architectural component to build a privileged abstraction layer responsible for *decouple* digital services and applications from the complexity and heterogeneity of interacting and managing deployed PAs. They allow observers and connected services to easily integrate cyber-physical behaviours in their application logic, and to design and execute high-level policies and functionalities without directly handling the complexity of end devices.

In the following the modelling contribution of a DT-based system in the industrial setting is described with higher specificity, exploring the characteristics that cyber-physical systems need to maintain in the industrial domain.

3.2.1 Data Ingestion & Augmentation

Each machine DT need to have an interface facing the physical world whose responsibility is to ingest information received by the physical world. The interface facing the physical world has to be flexible for the communication needs of each scenario, i.e. should be possible to use different protocols and interaction patterns in the interface to flexibly adapt to the physical world system implementation, gaining, therefore, an advantage over the *digital complexity*. A second aspect to consider in the interface is that it doesn't realistically know the information structure received by the physical object. Therefore, is necessary to consider in the interface some description of the physical entity *received* or *built* at the DT start. The description can be grouped into abstracted fields as *properties*, *events*, *actions*, and *relationships* [100]. After getting all machine information, they have to be processed by the DT. Then, information obtained by the physical world has to be passed to the DT core, where they are manipulated following some model or function, and then written as the DT state. Data manipulation is needed to extract some valuable information from the underlying physical entities.

This is the case, for example, of *Overall Equipment Effectiveness* [44], a performance metric for industrial equipment representing how efficient is the system in utilising the production capacity of production equipment in the time domain. OEE can be tracked by a production node DT that monitors its sensors and events and manipulates the received data to understand if the machine produces fast enough if the machine isn't working due to a breakdown or some other reasons, and if the machine production rate is meeting the expected speed. This metric does not depend on the underlying machine and usually is not tracked by the machines themselves. In real-world production systems, the OEE metric is inferred by tracking how much time a production

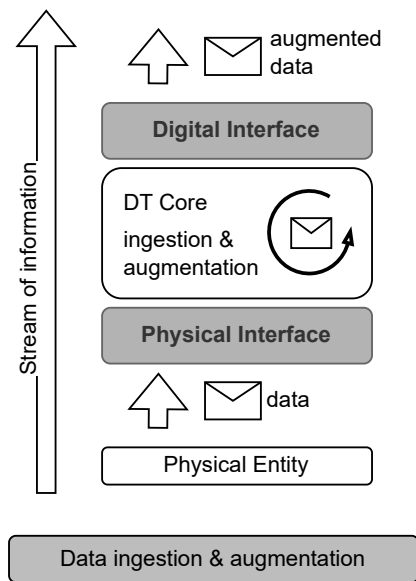


Figure 3.6: In DT data ingestion and augmentation, information received by the Physical Entity and ingested by the Physical Interface is augmented in the DT Core; the result is then exposed through the DT Digital Interface.

node passes in a certain state and what events happen in it, concerning production targets of the business unit. In a DT environment, production node information can thus be collected by its input interface and processed in the DT core to extract the OEE valuable information associated with the underlying production node state. After the core manipulation and updates, information has to be exposed externally to other applications or entities (another DT or a digital domain application) flexibly concerning the protocol used. The entity responsible for exposing information outside the DT on the digital side is another interface, following the pattern of the interface facing the physical side, therefore contributing to lowering the related *digital complexity*. Described process is depicted in Figure 3.6

3.2.2 Physical World Actionability

In the industrial scenario is also needed a pattern of interaction from the digital domain towards the physical world [21]. For example, this can be the case for machine setups, that must follow some specific information usually shared by industrialisation and scheduling offices, maintenance activities, where operators must interact with physical objects and manoeuvre them, or logistics, that must be triggered to coordinate with the production node buffers. Action capabilities have to be exposed by the physical objects

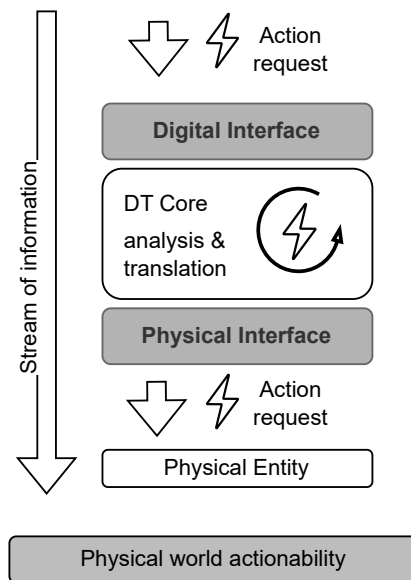


Figure 3.7: Actions are triggered by external entities through the DT Digital Interface; the request is analysed by the DT Core, and then passed to DT Digital Interface, which propagates the request to the associated physical asset.

through their description as reported in 3.2.1, and then are expected to be received from the digital domain (e.g. from another twin or another piece of software used by a system actor). As a consequence, the interaction pattern

can be considered as the one depicted for ingesting information 3.2.1, but with the opposite flow: the action request comes from the digital interface, which has the responsibility to correctly handle the request with a suitable communication protocol. After that, the request is ingested by the twin core that eventually analyses it, augments it, or translates it into a set of physical actions. Lastly, the twin core output is given to the physical interface, which has to communicate the result to the underlying physical world in the correct fashion (process depicted in Figure 3.7).

3.2.3 Cyber-Physical Relationships

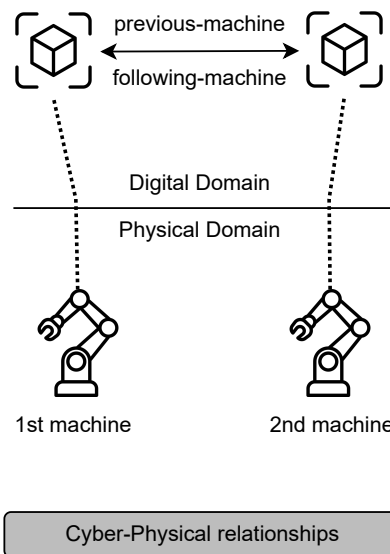


Figure 3.8: Each DT description is characterised by the existing relationships with other existing DTs.

Physical entities are usually related one to another and between actors in physical world systems. This is also the case for industrial environments, where relationship constraints exist to obtain a certain outcome or logic. The very practical example is represented by industrial layouts, where equipment,

production nodes and supportive pillars are grouped and related one to another. A very specific sequence can exist between production nodes, having therefore nodes that come *before* and *after* a given one. Moreover, production nodes can be grouped into clusters (i.e. departments) and clusters can express relations in turn. Operators and equipment can be related to industrial systems, *being part* of one area as well as another. Industrial layouts pose constraints also in KPI and system monitoring: throughput, for example, is a metric that needs to monitor only the *first and last* machines in a grouped system, being thus based on a relationship existing between them. Hence, DTs need to model also relationships between them, concerning the represented physical object (see Figure 3.8). It is indeed argued that such relationships contribute to the growing of static complexity, following the definition given in 3.1. Therefore, the work proposes relationship representation carried by DTs of the equipment as an additional tool for static complexity management. Moreover, relationship modelling enables also the possibility of navigating them if the relationship itself stores a pointer to the related digital entity. This characteristic is crucial to have the ability for a DT or external software to retrieve the needed information about related DTs.

3.2.4 Composition & Hierarchical Views

In the real world that we experience as humans, objects and tools are usually the results of complex compositions and interactions between parts, that need to cooperate to fulfil a goal. Cars are one perfect example, being the composition of different elements cooperating to obtain movement and controlling capabilities. Industrial architectures follow a similar pattern when it comes to information and interactions: resources are grouped into departments, de-

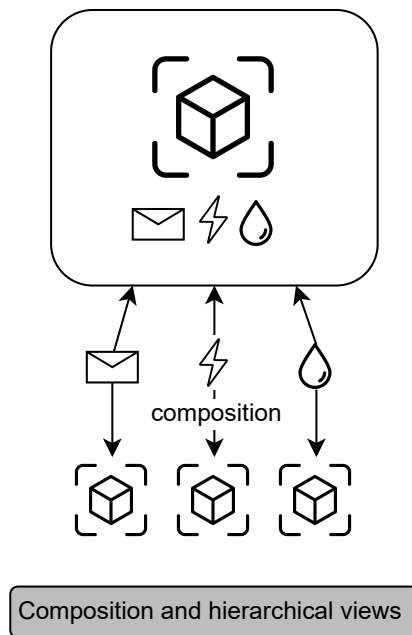


Figure 3.9: Outputs provided by different DTs and their Digital Interface can be the input of a new DT, through its Physical Interface; the obtained architecture is hierarchical.

partments into business units, and business units into plants. Talking about industrial metrics, a KPI involving composition abstraction is the composed OEE, also called Weighted OEE. The composition can be also used to create specialised views of the same shop floors, letting the high-level composition ingest only a subset of data from component DTs. As a consequence, a composed DT tracking the department performance through Weighted OEE can exist “in parallel” with another DT composition tracking the overall energy consumption of the same department. This mechanism can be implemented by exposing the digital-side interfaces of a group of DTs to the interface of the physical side of another DT: in this way, DTs exposing their digital-side interfaces act as components, while the DT ingesting data from its physical-side interface is the composed twin. Composed DT interfaces (the one that

faces the physical and the digital side) in this context can be abstracted as input and output interfaces. Flexibility characteristics in terms of communication protocols described in Section 3.2.1 are also requested in composed twins as they still need to connect to composition twins and expose information to other applications or twins, without communication constraints and with the best communication patterns.

Composition, as depicted so far (and depicted in Figure 3.9), is based on a *relationship* existing between different DTs (also heterogeneous) having something in common: being components of a higher-level entity. This is the case of a set of machines composing a department. Therefore, in setting up a composition, the relation existing between sub-components and the composed DT has to be taken into account when setting the state of all DTs. Recalling 3.2.3, each DT have to have relations like “*is-part-of*” in the components DT, while a relation like “*is-composed-by*” needs to be placed in the composed DT.

DTs compositions and hierarchical views are proposed as an additional way to obtain further *abstraction* of physical assets, moving the physical world representation towards *physical concepts* that suddenly characterise the considered domain. A very clear example is the concept of *department*: indeed, a department can not be considered only as a single physical object or a collection of physical objects. Information associated with assets composing a department can be mixed to obtain an informative level (and therefore, value) higher than the one obtainable by involved assets taken singularly. The abstraction power of DT compositions is, therefore, a valuable tool for the considered environment, as on the one hand offers the opportunity to support existing management techniques (as the division of production systems into departments correlated by existing relationships), and on the other

hand supports upper-level applications in gaining information at the needed abstraction level, abstracting the department representation in the digital domain and, therefore, tackling associated static and dynamic complexity.

3.2.5 Application Interaction

The structure reported until here can be considered as the representation in the digital domain of the structured state of affairs on the physical shop floor. Being exposed in the digital domain, at a certain point, this representation will interact with applications interested in getting information or requesting actions based on the reported state of the shop floor [21].

External applications interaction can happen with different DT levels, as depicted in Figure 3.10, on information, capabilities and responsibilities that a certain DT can have respect to others. Information retrieved by external applications can be retrieved for further use or to be presented to some stakeholders, as Figure 3.11 reports. Between possible aspects an external application can be interested in, there is monitoring the whole shop floor or a sub-part of it, having a specialised view of the shop floor state of affairs to be presented to one stakeholder rather than another or modifying the shop floor state of affairs to achieve certain goals or outcomes.

If the external application has to *present* the OEE state of a given department, for example, this application can communicate with the department composed DT and ask for its current OEE state. If instead, the interest of the application is to have the OEE of a particular machine taking part in the composition, information can be retrieved directly from the machine DT, or the department-composed DT if a different implementation has been done.

A similar pattern can be found for action requests. If an action involves one and only one machine, the external application can make the action

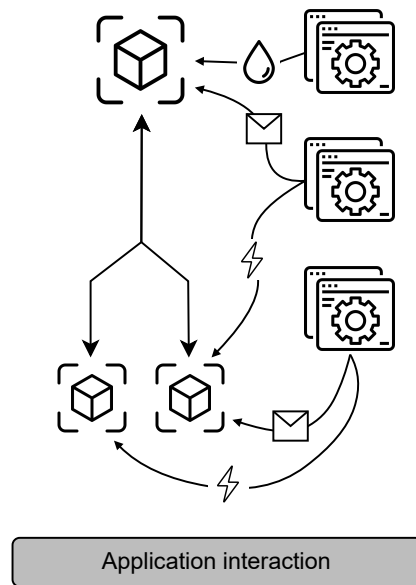


Figure 3.10: External applications can interact with the DT ecosystem at different levels, concerning convenience as well as the designed system result.

request directly to the machine DT. Then, if the machine DT analyses the request and finds it feasible, it can pass it to the underlying physical object. If an action, instead, involves a group of DTs, an external application can choose to make one request to each target DT or to make only one request to the composed DT that, in turn, analyses it and shares it with its component DT.

The depicted pattern of interaction can happen when setups are requested at a production changeover. If a setup involves the whole department or a big sub-set of it, e.g. eventuality likely to happen in a cellular manufacturing industrial architecture, is reasonable to have a composed DT for the whole department whose responsibility is also to manage setup requests for each machine. Therefore, the external application interacting to obtain the setup (for example, a scheduling application), will request the department DT. Then, the department DT forwards needed actions to each machine ac-

According to its core analysis. If instead, a setup involves a single machine, as is more likely to happen in a job-shop layout, where machines are grouped by common working capabilities but raw material usually does not traverse in sequence more than one machine in the same department, a direct setup request is more likely to be passed directly to the machine DT from the external application.

Therefore, what is argued is that different interactions can emerge when we consider the DTs ecosystem and external applications and that the most suitable concerning the actual goal is the one that should be considered during external applications implementations.

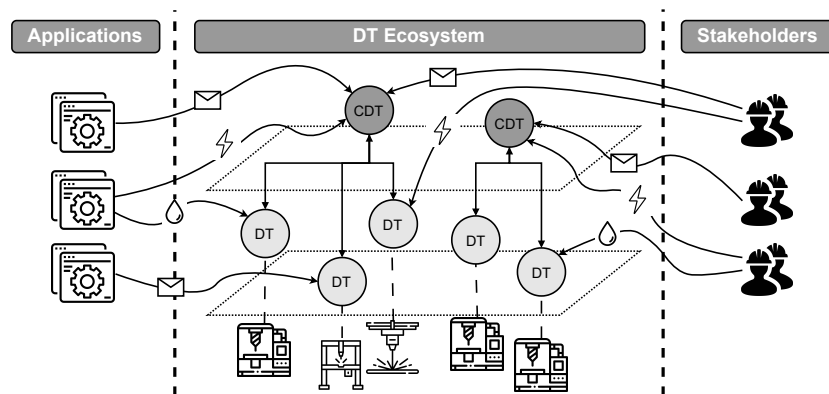


Figure 3.11: Overview of a simplified industrial environment digitised via DTs. Some possible interactions with external stakeholders as well as applications are depicted.

3.3 Organical Application of DTs

The use of DTs and associated investigated features opens up new opportunities for organically digitising and, contemporarily enabling cross-operation activities among different physical objects. Moreover, the composability ca-

pabilities of DTs give domain experts a powerful tool for grouping different DTs, specialising the associated view and obtaining digital entities representing physical notions (such as shop-floor departments) at a higher level of abstraction. It is consequently argued that this offers the opportunity to transpose not only low-level physical assets but also higher-level physical concepts. In the following, the proposed idea is detailed in-depth, and related advantages are consequently depicted.

3.3.1 A Hierarchical Architecture for Smart Industries

Following recent research trends and analysis [58] [10], the vision described in this dissertation promotes the idea that the envisioned decoupling between the physical layers, digital applications and their logical abstractions can be exploited by embracing DT. It supports the idea that the adoption of DTs at different architectural layers can enable native interoperability of systems and sub-systems in a smart factory, enhancing scalability, adaptation and coordination, promoting distributed autonomy and learning.

Specifically, are envisioned multiple digital abstraction layers responsible for effectively decoupling the complexity of the physical layers from intelligent services. Such a hierarchical architecture induces a logical abstraction able to: i) represent industrial assets and associated physical complexity without the effort of directly handling their *digital complexity* (in terms of communication protocols, data formats and interaction patterns) and enable native interoperability; ii) make the process of data collection, pre-processing, and formatting homogeneous and standardised; iii) augment raw input signals to create more variables to be used by the learning process; and iv) enable a first level of data analysis and decision making via learning algorithms, possibly to be used in a distributed learning.

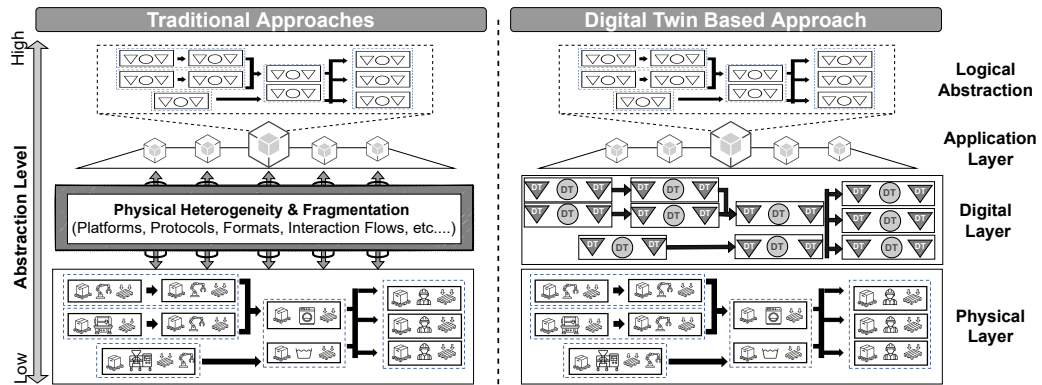


Figure 3.12: The schematic representation of the gap between a logical abstraction functional to intelligent applications and the complexity of directly managing the physical layer and its devices.

In Figure 3.12 is proposed a schematic investigation between traditional approaches, characterised by a direct link between digital services and the physical heterogeneity, and the hierarchical abstraction enabled by DTs. On the one hand (left side), the physical fragmentation is managed through vertical solutions characterised by a low level of interoperability and the consequent inability to effectively hide the underlying heterogeneity for the upper layers. On the other hand (right side), the adoption of DTs and the exploitation of their core capability to digitise and handle their physical counterparts opens the possibility of building a new digital layer in charge of managing the fragmentation characterising the PAs and exposing a uniform digital abstraction to the application layer.

3.3.2 Hierarchical Abstraction in Industrial Digitisation

This work considers that a major challenge in applying DTs to the context of smart factories is associated not only with the digitisation of a single physical asset but also with the mapping of complex and hierarchical environments

like production lines, departments and plants characterised by a plethora of devices, relationships and data. The adoption of DTs can improve application awareness by introducing a structured and uniform mapping of the physical layer.

Exploiting the *representativeness* of DTs we can model the complexity of a smart factory through the creation of a set of hierarchical, incremental and connected abstraction layers using multiple DT categories. Each of them will be responsible for retrieving data from the underlying levels, elaborating and enriching information according to the target behaviour, and finally exposing them with a specific granularity and a uniform interface to the upper layers. This approach allows to:

- decouple responsibilities because each DT handles a small group of connected components;
- augment distributed awareness since each twin focuses on a specific context and abstraction goal;
- enables interoperability through a uniform way to expose information across layers and decompose high-level action requests into low-level practical actions.

Furthermore, the *composability* can be also adopted within the proposed vision to model twins as the combination of multiple individual physical objects or other connected DTs according to the hierarchical level of interest. This possibility is not only useful from an operational point of view of connecting and aggregating multiple entities together but it can be exploited to support discoverability and awareness within the smart factory. Such a mechanism allows observers and applications to navigate the graph of DTs to find specific resources, functionalities or data with limited prior knowledge.

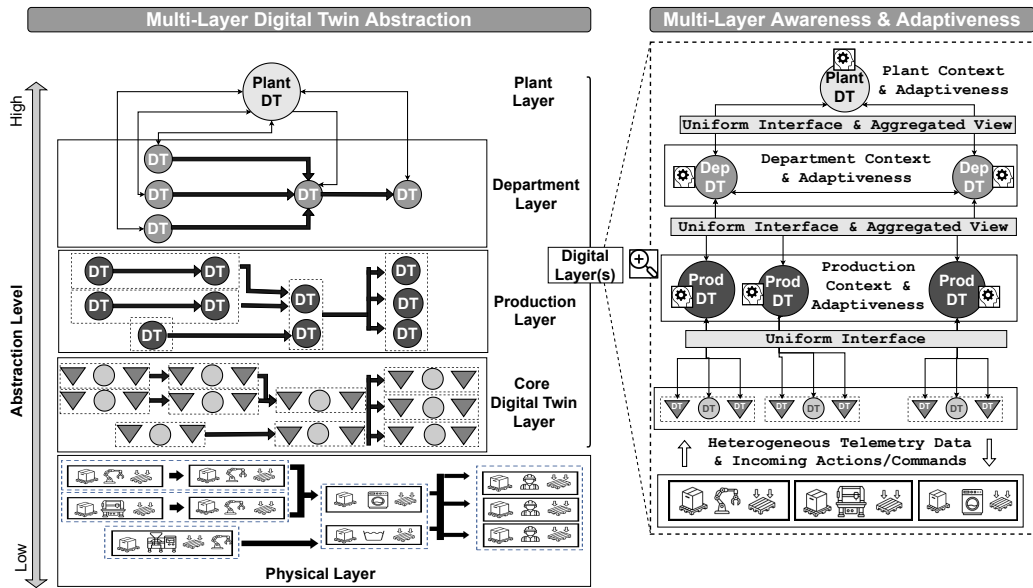


Figure 3.13: The digitisation of physical assets brings a new flexibility in the architecture and multiple hierarchical layers can be easily created to aggregate assets and propose different levels of observability to the industrial ecosystem and consequently multiple learning and decision points of view.

Figure 3.13 illustrates this multi-layer vision enabled through DTs where each abstraction level is, on one hand, able to build and operate on its local context with the data coming to the uniform interface of the previous layer (e.g., the Production Layer with the information coming from core DTs) and on the other hand responsible to expose enriched data and capabilities to observers and consumers of the higher chain of abstraction (e.g., providing data to the Department Layer and its DTs). In the lower layer (the physical layer), the physical world is “sensed” by the distributed sensors and, in turn, affected via distributed actuators. Distributed sensors and actuators are grouped in machines or general equipment. Composed machines and equipment represent *transformation nodes* and *input or output buffers* in the *Core DT Layer*. Data collected in this layer represents the state of

associated nodes, like buffer levels, the time needed for filling and emptying operations, the state of the transformation node, etc. Aggregation of related buffers and transformation nodes forms the production node in the production layer. Here KPIs and parameters associated with the production node are tracked, like the relative OEE [89], production stop times and rates and relative reasons (like machine or equipment breakdowns, node starving or sating). Going up in the abstraction, aggregation of different production nodes forms department nodes in the department layer, tracking their state and performance leveraging data coming from the underlying layer. Aggregation of different departments, warehouses and other operation functions (like maintenance or logistics) forms the plant node DT, the highest level of representation from an operations point of view.

3.4 DT Contribution Over Complexity

Industrial complexity investigated in Section 3.1 cannot therefore be ignored, as it derives directly from market pressures. It can be managed from a managerial point of view to minimise it, for example, by finding a trade-off when designing a production system (i.e. static complexity). But when this palliative also loses effectiveness, other support methodologies must be used.

The use of modern IT, IoT tools and related techniques can help in uniforming the representation of physical objects and transposing their complexity into the digital context, in which much more flexible management tools can be implemented. Technological simplification involves making the representation layer of the physical object uniform in the digital domain. In other words, is argued that the physical object or its representation should be equally accessible in the digital domain by any application (i.e. easy to

communicate), without necessarily having to specialise communication based on manufacturer limitations and, in general, offer the possibility to choose the best communication pattern and tool with respect the given use case.

Secondly, the representation of the physical object state (i.e. industrial equipment) and its consequent description should not necessarily be linked to the input data and their structure, but it should be possible to manipulate and possibly standardise this representation concerning the needs of the use case. In this way, the related part of *digital complexity* would be mitigated, as it enables the adaptation of data structures coming from heterogeneous physical entities (even if potentially of the same nature) to the use case and the needs of the stakeholders.

However, the described data manipulation is not only useful for managing *digital complexity*: the data carries information, which, following their manipulation, can lead to higher level knowledge, and therefore to possible actions/choices that can be transposed into the physical world. It is argued that the representation of the physical object in the digital therefore should not be strictly static, but should have a dynamic nature, to enable the physical world to (possibly) react to the information received in its context, while increasing the computational capabilities of the physical objects (often limited in hardware resources) and therefore attacking the dynamic complexity of the industrial world.

Finally, the relationships of each physical object should be transposed into the digital, to simultaneously report the logical (i.e. static) complexity existing between different objects and enable the navigability of the physical structure, in the digital. The relationships can be horizontal (e.g. object A precedes object B) or vertical, e.g. objects A and B compose object C, thus replicating the experience of the human world of objects in which a given

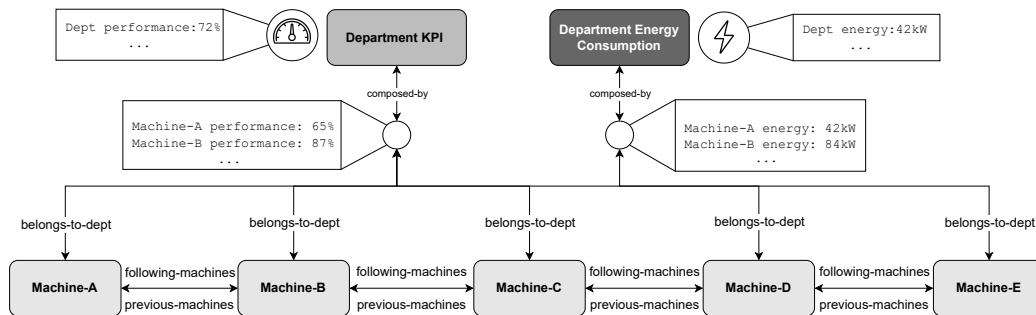


Figure 3.14: Generalised representation of possible relationships existing in the industrial scenario.

physical object can be together with several sub-components, which in turn are potentially composed of sub-assemblies (see Figure 3.11).

The vision proposed by the dissertation applies well to the industrial context, in which a heterogeneous set of equipment (e.g. a robot, a Computer Numerical Control - CNC - machine, a buffer of input and output products) composes a production node, several production nodes compose a department, multiple departments compose a business unit, the multiplicity of which finally composes a production plant. As reported before, relationships can also have a horizontal nature, in which, within a production node, for example, the product flow first visits the incoming buffer, then the CNC, and then the exit buffer. Vertical relationships in the production sector are of particular importance, as they allow increasingly larger portions of the shop floor to be kept under control and the related KPIs to be built at an increasingly high level, and are therefore often practically used in industrial contexts. An example is the calculation of weighted Overall Equipment Effectiveness (OEE), which groups the OEE of the different machines in a given department offering a higher-level summary metric. Horizontal relationships, on the other hand, can represent logical relationships existing between var-

ious physical entities, such as order relationships (e.g. the thing A comes before the thing B). These relationships represent key information as well as vertical existing relationships for the calculation of KPIs: the throughput, for example, must take into account the first of the last machine to carry out the processing, to calculate the delta processing time and divide it by the total number of pieces produced. In this case, order relations are important to identify which events of which machines to consider in the calculation.

Vertical and horizontal relationships also enable the *navigability* of the system, with therefore the possibility of exploring things representation in the digital domain (physical or logical) associated with the i -th node by examining the information linked to the relationships of the node itself.

3.5 Resulting Advantages

As introduced in previous paragraphs, is necessary for industrial manufacturing systems to handle higher complexity over time because of increasing challenges set by market requests. Complexity types can be divided into 3 categories, static, dynamic and technological which, together contribute to a Hierarchical and Ecosystem Complexity. Without DTs, applications and services are directly exposed to the heterogeneity of the physical layer, with the result of being forced to deal with the specific implementation of each physical object, encountering as a consequence design limitations, continuous maintenance on the digital side, and sensibility to technological lock-in (e.g. in scenario of service disruption, firmware updates, data format variations or machine replacements). Moreover, the absence of a DT industrial ecosystem also poses some challenges to the logical representation of a group of physical elements in the digital domain. In other words, representing vertical and

grouping relationships (i.e. physical object compositions) and horizontal relationships is hard when digital services have to interface with the physical layer. Therefore, responsibilities to model, keep synchronised existing physical relationships and build a structured logical abstraction of a complex environment as an industrial one, are entirely delegated to application layer services that, instead, should only focus on how to exploit their targets in the best way.

In this regard, is investigated how a DT abstraction layer clearly helps digital services as they have to handle, and eventually transpose the whole complexity of the physical layer, doing it better than existing digital services, and lifting them in doing it. Indeed, DTs first can talk with a plethora of physical assets, despite the asset communication protocol and pattern, as the communication itself is handled by DTs' modular communication adapters. In the same way, communication is also flexible on the digital side, because of the implementation of the same pattern.

With DTs, data and information are organised following the expected pattern of the physical domain: if information regards the physical object A, the relative data is ingested, handled and exposed by its associated DT. Since DTs, as considered and proposed in this dissertation, are also considered active entities, ingested data can be augmented through its manipulation, using modern data analytics tools or even AI systems (more on this later). Digital services can therefore take advantage of it, as data manipulated by DTs can be specialised concerning the context, the goal and other entities interacting with DTs. Because DTs are modelled as active entities, they can also pass action requests from the digital domain to the physical one, taking the responsibility to communicate with the appropriate protocol and lifting therefore up digital services in doing so. Moreover, being active, they can

also receive a high-level request, analyse it, and eventually break it down into a set of actions to perform in the physical domain. Because DTs act as representatives of physical objects in the digital domain, they can also refuse to pass an action (and therefore, let the physical counterpart perform it) if the request results are infeasible. Moreover, it also supports the idea that the act of passing actuation requests can also be launched by the DT itself, if its internal model is designed to do it in determined conditions (e.g. if the object sensor A reaches a certain threshold, active the actuator B). Therefore, the actuation request can be both received by an entity external to the DT or launched directly from the DT if its internal model meets some constraints.

Another benefit given by the adoption of DTs which has been investigated as being very useful in industrial environments, derives from their ability to be composed out of multiple DTs or physical assets. Indeed, a single DT can be the counterpart of a set of multiple sensors and actuators which compose an actual physical entity or a set of multiple physical entities that, logically, belong under only one concept. In the industrial domain, a similar example can be found thinking about *retrofitting* activities, usually involving already deployed equipment. In this scenario, an existing production machine can be enriched with multiple sensors and actuators, having, therefore, one logical object (the actual transforming machine) composed in the physical domain by multiple objects, some of which already exist and others instead added after the deployment of the machine. Another example, following the same pattern, refers to assembly lines, which, in the lean context are often built using modular structures and electronic hardware. Therefore there is an assembly line that can be equipped of multiple connected sensors and actuators, of different vendors, which is better to represent in the digital domain

as a whole for the production purposes, instead of single sensors/actuators DTs.

The composition can be also used to mirror in the digital domain existing relationships among industrial equipment already in use in manufacturing domains, such as departments or business units. The composition pattern, in this case, follows the same principle as before, with data received from a machine that can be manipulated, through augmentation, to retrieve and track department or business unit KPIs for monitoring purposes. Eventually, department DTs can be used to distribute a set of actions, such as production schedules and emergency service calls to other manufacturing pillars (e.g. safety or maintenance) and in general to act as supervisors and proxies for the whole department. A similar application has been already theorised, as in [68], and [144]: the former expects a further convergence between schedules generated by optimisation software and their effective actualisation, proposing an anomaly detection and associated dynamic scheduling tool based on DTs; the latter reports a DT-enhanced dynamic scheduling methodology, considering in the scheduling activity machines real-time metrics and events, as availability, disturbance detection and performance evaluation.

Composition abilities as well as the possibility to implement multiple specialised DTs on the same physical objects, bring also the opportunity to offer to stakeholders or external application-specific point of view of the same system concerning the final purpose the stakeholder or the application needs to fulfil. Indeed, is argued that this characteristic comes particularly in handy in exposing DTs to different industrial organisation pillars, each of which specialises in the pillar activities. The logistic pillar will therefore see its specific DTs mostly composed of production system buffer levels, Automated Guided Vehicles (AGVs) or logistic operators, warehouses and associated equipment,

forklifts and so on, from the lower to the higher composition level; safety will be therefore mainly exposed to safety barriers and equipment, operators' health and stress state, and compositions between a certain workplace and associated operator, to check if the risk assessment of a certain work area fits the assigned operator or not. Other examples can be done for maintenance, workplace organisation, production managers, schedulers, continuous improvement and other pillars.

Becomes therefore clear why the investigated representativeness and composition are two powerful DT tools to expose in the digital context industrial static and dynamic complexity, offering a real-time structure composition of the actual shop floor, comprised of its horizontal and vertical relationships, and reporting the actual state-of-affairs. The ability to specialise the view of a certain DT acts towards the simplification of the actual shop-floor management in the proposed approach: indeed, in this way, stakeholders and applications are exposed only to the information they are interested in, without any additional effort in retrieving data, manipulating and presenting it. Relationships and composition structure instead transpose the equipment organisation in the digital domain, enabling therefore the application of modern IT solutions to the real view of the physical counterpart. Among possible applications, there is for example the classification of the actual shop-floor system, with the consequent selection of the most suitable scheduling strategy. Moreover will be possible to schedule the whole production system as an entire monolithic entity, or only a subpart of it as a department, to obtain better optimisation performance in a shorter period or to respond to an unexpected event which, by the way, affects only a small part of the system. Another opportunity consequent to the proposed investigation can come in the simulation of new, better layout configurations or the fitting

of new products in the production system, to understand if the production system is technically able to produce a new product, how it will impact the portfolio complexity [87], and if the actual production capability can fulfil the new product demand or not.

The set of topics investigated so far support the idea that opportunities enabled by the considered active DT shift static and dynamic complexity from the physical to the digital, simplifying *digital complexity* and offering the opportunity to work with tools that can solve problems at a different scale concerning the one that involves information handling without digital tools.

Among the nowadays most powerful tools, there is also AI, which until now had only a minimum space in the dissertation. AI covers a strategic part in the smart digitisation of industrial systems, as brings intelligence into the mix. This intelligence can be placed in strategic points of the depicted digitised system, powering up augmentation capabilities and opening new scenarios for existing, connected equipment. In the following chapter possible relationships between AI and a DT industrial ecosystem are investigated deeply, and emerging patterns of interaction between AI and DTs are proposed.

Chapter 4

Industrial Digital Twins and AI

Managing industrial complexity and intelligently transposing physical assets and related higher-level concepts into the digital domain are only the first steps into the industry of the future journey. AI is already bringing great expectations and impacts in several sectors of human economics, and industry is expected to receive as many disruptions. Emerging relationships between the AI model and assets digitised through DTs need therefore to be explored and discussed. In the following paragraph, the concept is investigated, depicting 4 possible interaction patterns between the considered technologies and depicting how they potentially impact shop-floor environments.

4.1 The Role of AI

AI is widely recognised as a really powerful tool to extract high value from data of any kind. AI is impacting several world scenarios carrying general improvement expectations, and the manufacturing context is no exception. However the distributed nature of these systems, their heterogeneity and inherent complexity make this application critical.

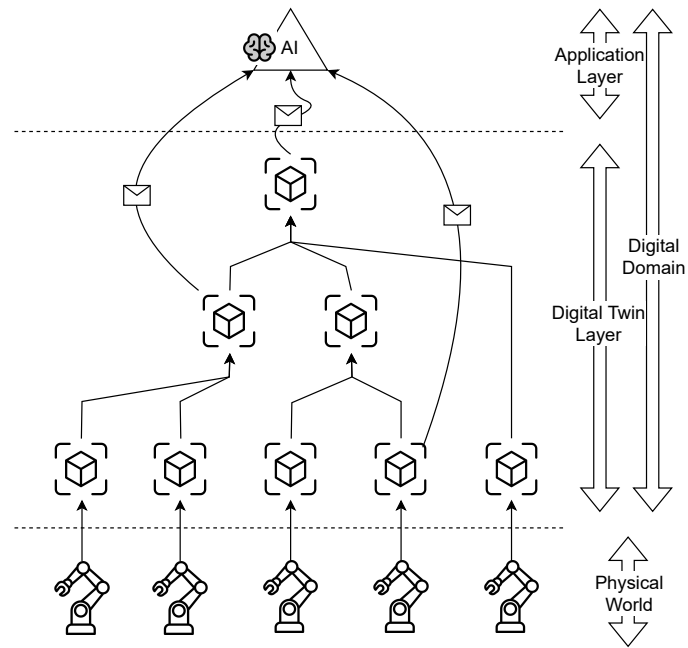


Figure 4.1: AI placed externally the DT ecosystem, analysing information provided by DTs, and exposing relative insights.

AI application in the industrial domain is not new, and its application and relative innovation are already under our eyes. Several areas are investigated, from logistics to quality applications, passing through scheduling, ergonomics and maintenance activities.

Some of them have already been highlighted, for example, [134] in the area of logistic activities improvement. The mentioned activity in particular witnesses the benefits of bringing AI into operations activities, to make them more effective. Another example can be done in the same area considering CNC machines, as reported by [116]. Several valuable outcomes of applying these tools are depicted, e.g., extending tools' life-span during machining operations, predicting surface quality of a processed part, or minimising machines' power usage, mostly obtained by analysing cutting forces and tools wear measures as well as the amount of power consumed during a determined

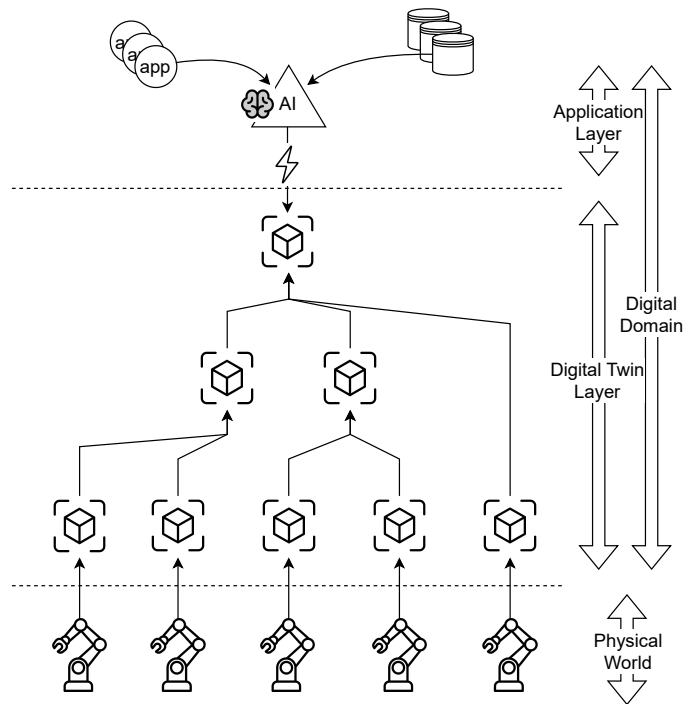


Figure 4.2: AI placed externally the DT ecosystem, receiving information from digital sources and triggering actions exposed by DTs.

process.

Optimisation problems under the maintenance pillar are improved too by the application of AI, for example in the area of Total Production Maintenance (TPM) [35]. The article describes that TPM problems can be modelled as semi-Markov decision processes (SMDPs), optimally solvable in a reasonable time for small scenarios. By the way, optimisation becomes tricky when the problem size grows, making the optimality impracticable to reach with traditional methods. Alternative approaches rely on the application of AI techniques, and the one reported relies on the application of Reinforcement Learning (RL). Therefore, an application of RL to a TPM scenario is described, resulting in optimal solutions for small-scale problems and near-to-optimal solutions for large-scale scenarios. Steps towards problem prediction

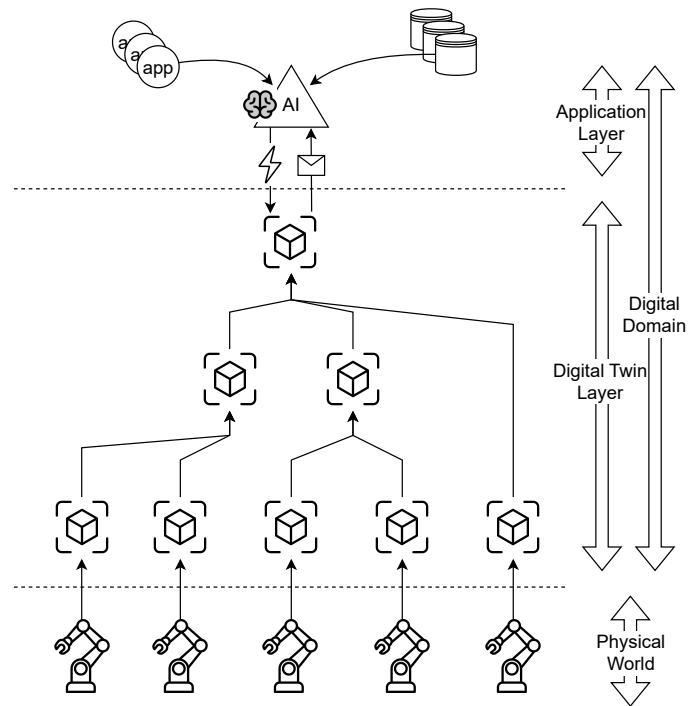


Figure 4.3: AI placed externally the DT ecosystem, triggering actions exposed by the DT ecosystem, concerning data received by DTs and external digital sources.

and solution prescription have been also done [95]. In such work, AI in conjunction with simulation and optimisation, has been applied in a CPS environment under MTO and ETO industrial architectures, implementing a decision support prototype able to link prescriptive maintenance capabilities with Production Planning and Control tools. Instead, a more generalised time-to-failure prediction system is explored in [148]. The article itself points out how existing predictive maintenance tools are mostly applied to highly specialised use cases, with a lack, and a consequent need, of a generalised approach. Therefore, a generic end-to-end method for TTF prediction is proposed, where a universal feature extraction method is applied, in conjunction with established feature transformation, selection, and other techniques. Dif-

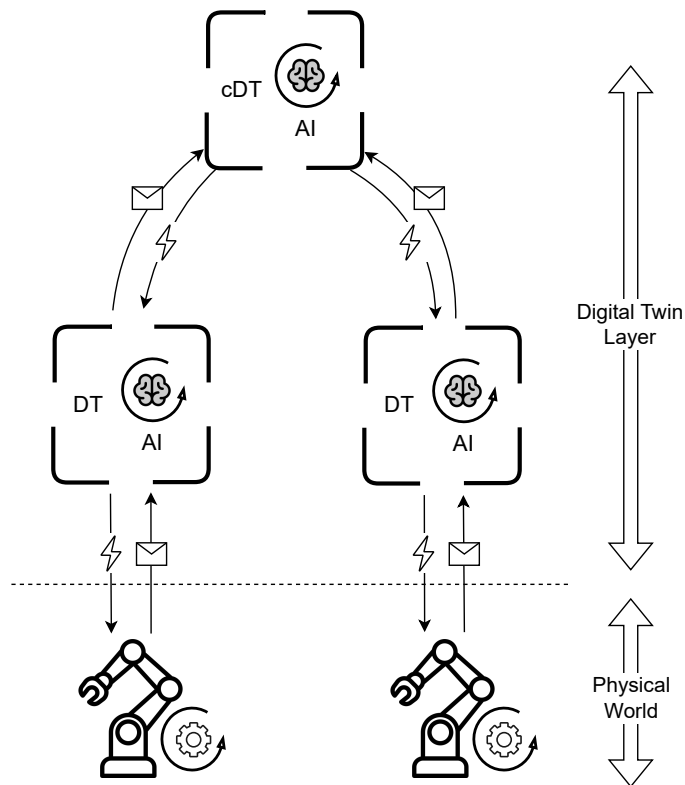


Figure 4.4: AI embedded inside a single DT, augmenting its behaviour based on internal data.

ferent state-of-the-art learning techniques have been considered, comprising the associated hyperparameters optimisation strategies.

Scheduling activities are no exception when it comes to the application of AI in manufacturing. An example is the application of RL algorithms for a scheduling problem of an AGV fleet in a shop-floor scenario [141]. Target AGVs were characterised by fixed tracks, and were equipped with a robotic manipulator. Their tasks were about transporting work in process products between different operations machines, with the final target of minimising the average job delay and total make-span. The most interesting part of that work is that each AGV makes decisions based on all machines' states

and relative jobs, which can be accessed in real-time by AGVs themselves. The task of scheduling activities across a flow-shop production system has been also studied in [53], where Q-learning techniques have been applied to reach better performance with the target of minimising the make-span. In particular, the algorithm target was to choose whether to change or not the NEH heuristic job inserting mode, in the attempt to improve the overall system efficiency.

AI applications are not limited to the well-known areas of scheduling, optimisation or maintenance. Indeed, opportunities opened by AI enable the application of IT also in industrial shop-floor areas where was hard to monitor and gather data. An example is the tracking of ergonomics and workplace safety, as demonstrated by the following work [75]. Ergonomic risk assessment has been considered in an activity-picking job scenario, with the introduction of a height-adjustable mesh truck to limit operator fatigue and associated hazards. System improvement evaluation has been done through the adoption of kinetic-based software that, with the help of modern AI and Machine Vision techniques, classifies automatically the ergonomic NIOSH index of the recorded workplace.

Despite the not negligible number of AI applications in industrial scenarios, most of them are tailor-made for specific tasks or problems, limiting projects to their silo and preventing solutions to be transferred from one application area to another [60]. In response to industrial requirements, heterogeneity and the rapid response to changes needed, the dissertation promotes the need for a level of abstraction capable of intelligently “disconnecting” physical complexity from digital complexity. A holistic and fully integrated approach in digitising an entire production system, exposing its representation to the digital domain, and finally adding intelligence in key-strategical

points is not yet done, but desired.

As pointed out in [60], DTs are widely recognised as the candidate technology to overcome difficulties in integrating AI in industrial systems and industry 4.0 without disrupting production. DTs, indeed, act as an abstraction of the underlying real-world structure, offering to the digital domain applications a real-time representation of the real-world state of affairs. Moreover, has been proposed in the previous Section that DTs also *transpose* the internal physical complexity in the digital domain, both at the static and dynamic level, making it seamless to apply analysis, optimisation and intelligent tools. In other words, DTs can have the responsibility of abstracting real-world complexity, managing the information ingested by the associated physical object, exposing relatively organised information to digital applications such as AI models or algorithms, and eventually acting as the entry point for actions to be performed by the associated physical object. Indeed, acting as an active representative of associated physical counterparts, DTs can also lower the difficulty in collecting data for AI algorithms and eventually communicate a reaction, as the information they carry is already organised as expected in the physical domain. The advantages of applying AI algorithms in conjunction with a DT industrial ecosystem are straightforward. Nevertheless, the expected impact is well defined in the expected level of maturity of DTs architectures reported in [27]:

- *Descriptive*: at this level, the DT is fed with real-time data streams from the physical counterpart, describing its status and events, with the possibility of obtaining historical data from its DT. In this case, AI does not provide any kind of added value, as there is no real “intelligence” in the definition; the physical state is, indeed, just neatly reflected in the digital domain.

- *Informative*: at this level, the DT can present *augmented* information, such as diagnostics, physical asset health conditions, fault finding and troubleshooting. In this scenario, a higher degree of data manipulation is needed to obtain higher-quality information concerning the descriptive level of maturity. In some cases, simple augmentation capabilities carried out by a simple DT internal model are enough to obtain needed information. Nevertheless, sometimes the usage of AI techniques can be necessary to extract diagnostic information or for troubleshooting activities.
- *Predictive*: in this case, some prediction starting from received data is expected to be exploited. Therefore, the system can understand the close-future trajectory, thanks to the use of AI and ML techniques. Indeed, there is no other way to obtain prediction capabilities, therefore AI utilisation is needed to reach this level of maturity.
- *Prescriptive*: as a further step, the system can recommend actions concerning available data and associated prediction to optimise future trajectories and/or avoid some future happenings. This can be considered as an advanced application of AI, therefore the ability to reach the described maturity is strictly bonded to it.
- *Autonomous*: as a last step, the system can autonomously adjust its predicted future trajectory, being in a close control loop and therefore executing corrective actions autonomously. This is the highest level of maturity for a digitised system, and the most difficult to reach. Eventually, if the system can not obtain the corrective action expected outcome, it should be able to ask for human help and activate a collaborative plan to reach expected targets. In this regard, the ability of

DTs to accept actions and pass them to their physical counterparts, enables autonomous capacity, drawing a future of collaboration between AI techniques and DTs ecosystems.

The positive spiral of interactions between DTs and AI tools opens up new application scenarios in the manufacturing domains, exploiting possible interaction patterns that can arise between AI and DTs in the industrial domain. Among them, 4 patterns are going to be presented in this dissertation, as briefly introduced in the following:

- *Observer AI*: AI external to an industrial DT ecosystem, receiving data from the DT architecture and monitoring it as a whole process or department, depicted in Figure 4.1 and described in Section 4.2;
- *Advisor AI*: AI external to an industrial DT ecosystem, collecting data from external sources and eventually requesting actions to the DT ecosystem to reach a target state of affairs, depicted in Figure 4.2 and described in Section 4.3;
- *Controller AI*: a mix between the above two cases, with an AI system external to the DT ecosystem, collecting data both from the DT ecosystem itself and from external sources and requesting actions to DTs, depicted in Figure 4.3 and described in Section 4.4;
- *Embedded AI*: AI internal to single DTs, augmenting its capabilities, depicted in Figure 4.4 and described in Section 4.5.

In the following paragraphs, these four proposed interaction patterns between AI and DTs in the industrial domain are deeply explored and described.

4.2 Observer AI

The first pattern of interaction between DTs and AI techniques investigated is also the easier to think about an AI model external to the DT ecosystem, learning from received data and exposing its insights to other applications or stakeholders, therefore defined *Observer AI*. In this scenario, the Observer AI does not know anything about DTs internal structure, models, behaviours, and ecosystem structure. The Observer AI has its internal business logic, knows the kind of data needed to work, and exploits information coming from DTs to properly absolve to its analysis tasks. DT information exploitation can happen through hardwired processes, with a configuration, or using advanced service discovery solutions [26]. There are no prior limits to the layer of the DTs ecosystem that can communicate with the Observer AI at the application level. In other words, the Observer AI application can retrieve data from a single high-level DT, from multiple high, middle or low-level DTs, or with a hybrid approach, as reported for example in Figure 4.1, in the left box. The actual implementation depends on the modelling approach followed in each application context and on the level of complexity accepted in each implementation. More details can also complicate each scenario, such as the updating frequency of the data of the target DT, access grants to different DTs, and so on. After having received the needed information, the Observer AI performs its analysis and offers gained insights into some services of the application layer, such as a scheduler, or a dashboard for potential stakeholders.

A practical use case proposed of this pattern in the industrial scenario may involve causality learning, for example. Indeed, manufacturing operations strongly depend on a variety of different factors and situations whose management and coordination typically require domain knowledge, decision-

making and problem-solving skills, and the ability to react to rare or unexpected events. To gradually acquire these adapting skills, the systems must learn their internal model of the environment, representing relations and dependencies between different components and variables. The abstraction level provided by DTs represents a key enabler for this ambition. Since many control and management tasks in manufacturing processes deal with the identification of reasons for specific behaviours and situations [82], such as in root cause analysis [130], learned models should also represent *causal* relations. Exploiting causality contributes both to enhance system explainability [23] (a crucial property having humans-in-the-loop, a component that is not going to be deleted soon from manufacturing environments) and to improve generalisation skills, by transferring the acquired knowledge to different situations and tasks, including scenarios that involve forms of reasoning [110].

Classic applications of causal models and causality learning to smart factories include fault detection and root cause analysis [131]. While the former focuses on the task of detecting, classifying, and thus properly handling faults to minimise their impact on production activities, the latter aims to identify the very first cause in the cause-effect chain and eliminate it with a corrective permanent action. Root cause analysis activities usually require a strong understanding of the production context, manual information collection across the plant, days or weeks of data analysis, corrective activities implementations, and results monitoring. Causality can also improve different forms of automation. For example, in the scenario of dynamic and adaptive scheduling [68] [17] [77], a machine stop can be classified in different ways according to the root cause of the fault. In this regard, different types of causes can lead to different rescheduling approaches, maintaining overall transparency to the management concerning the choices made by the automatic system.

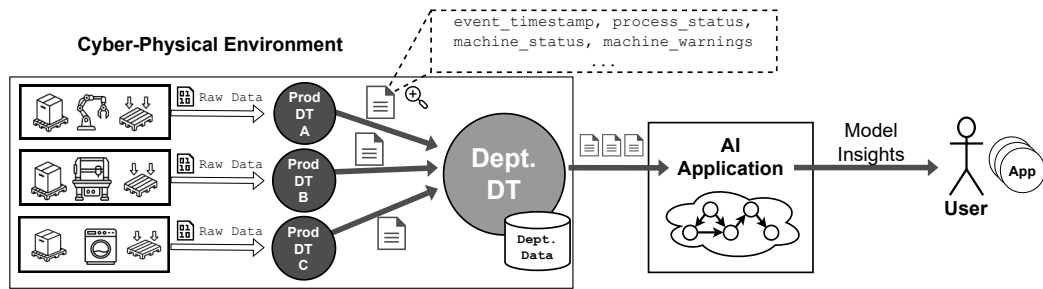


Figure 4.5: Interaction pattern between the DT ecosystem and the external learning model analysing data provided by the ecosystem itself and exposing relative learned insights.

Section 5.1 will present the experimental investigation results of a causality learning application in a digitised industrial environment through the use of DTs [72]. Application of DTs to physical environments decouples physical entities from the digital domain. Moreover, the additional composition ability of DTs moves the system towards abstraction, from a representation of low-level physical entity data towards high-level physical concept information. As a consequence, variables and links in a causal model generated from a DT architecture can be associated with high-level concepts (see Figure 3.13). Upper layers in the hierarchy have a wider view of the events happening within the considered scenario, and thus can capture dependencies among a larger number of variables: for example, the department layer can take into account the relations amongst all its production nodes.

Figure 4.5 represents an example of the proposed system, where information coming from physical assets is ingested by the production-level DTs and possibly manipulated by their internal model. Then, exposed information is passed to the department DT, which merges lower DTs information and executes other augmentation activities if needed. Then, such information feeds the AI application external to the DT architecture, creating its inter-

nal model and passing the results to other applications in the digital domain or to end users.

In the architecture investigated so far, DTs are used as a building block for ML activities within the smart factory [59]. This could be the case, for example, of object detection in images or videos captured by on-board digital cameras, activity recognition from signals emitted by sensors placed in the environment, fault detection and time-series forecasting for predictive maintenance, coordination amongst goal-oriented smart devices [78]. Some of these learning activities, especially low-level tasks (e.g., dealing with signal processing and data analysis) might take place also *within* DTs [59], as will be described in Section 4.5.

4.3 Advisor AI

As described so far, the most basic form of interaction between a DT ecosystem and AI tools is when AI acts *externally* to the ecosystem, fed by system data. Nevertheless, this scenario can quickly pose some limitations that can essentially arise from two different aspects: the first is the lack of possible control that the AI system can have over the CPS through the DT ecosystem, limiting valuable guidance based on AI over the physical system state of affairs. The second aspect comes from the source of information that nourishes the AI learning activities: indeed, CPS data could not give the type of information needed to determine learning outcomes useful for guiding the DTs ecosystem and related physical counterpart. This could be the case for a scenario where the industrial system is modulated concerning information coming externally from the CPS itself, e.g., the market demand for the product or service provided by the system. A practical example comes from the

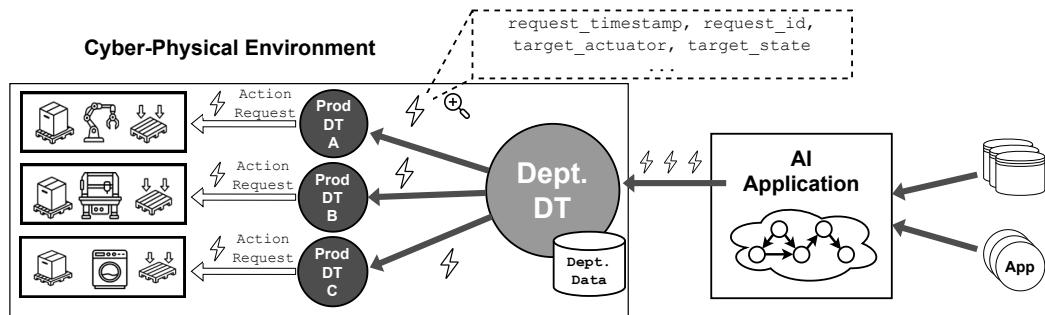


Figure 4.6: Pattern of interaction between a DT ecosystem and an external AI fed with external data as well.

energy production industry. Most of the energy production systems have to produce the exact amount of energy that the market is asking for, respecting the power frequency of the infrastructure. In certain scenarios, additional production equipment is eventually started (and then modulated as a consequence) to meet peaks in power demand. Monitoring the market energy demand and modulating the right amount of produced energy is, therefore, crucial. In this context, the AI needs to receive the requested instant power of the system, which data comes from sources external to the production CPS system, and analyse it. Then, the outcome of the AI model may bring some needs for the CPS system to execute a set of actions, to meet the forecast production demand. Therefore the AI system has to interact with the CPS, sending action requests to reach the target. Such actions will then be analysed by the DT ecosystem, possibly decomposed into a set of sub-actions, and finally passed to the physical counterparts of the system, reaching the target state of affairs required by the AI system.

Because of the role that AI solves in this case, the proposed name is *Advisor AI*: indeed, the system collects information from sources external to the DT ecosystem, changing DTs' behaviours through their exposed actions,

concerning the result of gathered information. The pattern is depicted in Figure 4.6: in this case, the Advisor AI interacts with the DT ecosystem only by providing action requests, that are the result of learning activities based on information received by external sources (databases or other applications). In the figure, action requests are passed to the Department DT, but this detail is not mandatory to respect the pattern. The Advisor AI can indeed interact with DTs posed at any level of the hierarchy, as described in Section 4.2. The approach depends on the implementation choices of each practical scenario.

Following the system classification proposed in Section 3.2, it is argued that the relationship pattern between an Advisor AI and a DT ecosystem lets the system move towards the *autonomous* class. Indeed, future states of *components outside the DT ecosystem* are considered and then the actual system trajectory is adjusted accordingly. Some degree of adaptation to real-world changes is therefore reached by this pattern. Nevertheless, in this case, the internal state of the system is not considered, lacking the *closed control loop* cited in [27], and opening the path to a mixed scenario described in the following section.

4.4 Controller AI

Is now analysed the scenario where it is necessary to mix the two patterns described so far to obtain a system that can retrieve information both from the monitored DT ecosystem and other external sources of information, combine them and eventually actively control the physical behaviour trajectory to obtain the desired outcome. The proposed name for the depicted pattern is *Controller AI*.

It fostered the idea that can be considered Controllers AI tools as AI-enabled industrial schedulers. Enriching an activity scheduler with AI to overcome optimisation limitations in computing time for large-scale problems or the quality of the solution is not a new idea. Several implementations and proposals have already been made in the industrial sector, both with the use of agents, as well as other types of learning algorithms. For example, authors in [54] propose a Q-learning algorithm for the flow-shop scheduling problem, with the make-span used as a feedback signal, and combined with the NEH heuristic. Another Q-learning application for job-shop scheduling based on a multi-agent technology application is proposed in [136], as an adaptive scheduling strategy to uncertainties in dynamic environments such as industrial ones. The contribution of [147] considers smart manufacturing environments as different from traditional ones, because of their connected and collaborative nature, proposing therefore a deep reinforcement learning method to minimise the maximum completion time of all production tasks. A different approach is the one described in [67], where an Artificial Neural Network (ANN) is generated through a specialised algorithm called NEAT (NeuroEvolution of Augmenting Topologies), in a hybrid flow-shop scheduling use case considering product families setup times. To understand whether a production rescheduling is worth the effort in front of a disruptive event, a rescheduling framework integrated with optimisation tools and ML techniques is proposed by authors of [69]. ML techniques have been used to classify rescheduling patterns, while, different rescheduling strategies have been used according to the actual classification. In [143] Priority Dispatching Rules for job-shop scheduling problems are learned by a deep reinforcement learning agent, resulting in high-quality PDRs learned by the agent and relative performance capabilities.

Having such a plethora of scheduling applications that are also moving towards the concept of smart manufacturing, considered as a production environment connected to the network, opens up the depicted use case. Firstly, smart applications (i.e., applications using AI as proposed scheduling ones) collect information from different data sources, both internal and external to the DT ecosystem. Market requests, customer orders, as well as planned stops for maintenance or other management operations activities planned by human managers, for example, represent external data sources, while the CPS state description as well as its future predicted state represent the internal source of data of the system. Secondly, after having efficiently extracted a feasible solution with the given information, the external smart application gains the opportunity to directly pass the solution to the underlying CPS. Lastly, the CPS fulfils its responsibility in actuating (or not actuating, concerning each DT internal rule) the received production schedule in its physical counterpart.

The relationship between the Controller AI, external data sources as operations manager or customer orders, internal data sources as DTs ecosystem state and existing *actionability* of the latter, are depicted in Figure 4.7. It is worth pointing out that the Controller AI not only serves as a step towards the reactive and/or proactive behaviour of a manufacturing system, but it also comprises the exposure of useful information insights to other applications or stakeholders, as described in Section 4.2.

The Controller AI can enable the system towards *predictive*, *prescriptive*, and possibly *autonomous* class, for the classification proposed in Section 4.1. A similar conclusion has been proposed in Section 4.3, as the system behaviour is adjusted by the external AI concerning the trajectory extracted by its learning activities. Nevertheless, if this ability is gained by the previous

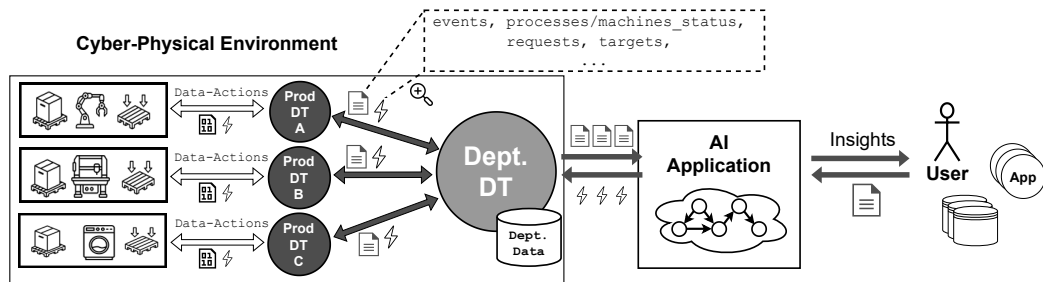


Figure 4.7: Pattern of interaction that mixes the ability for the external AI system to gain information from a multiplicity of data sources, comprise the DT ecosystem, and then eventually modify through DT actions' the physical behaviour.

pattern, the last one described in this chapter should reach a *higher* level of autonomy, because of the possibility to enrich the learning capabilities of the AI system with also the trajectory depicted by a representation of the state of the industrial system with information provided by the DTs ecosystem. It is worth highlighting that the implementation of such tools does not guarantee the ability to reach a *full* or *partial* level of autonomy of the system: the outcome of such classification depends on implementation details and the quality of the overall system, as well as by the level of reliability reached and maintained by the depicted patterns over time.

Since complexity has been pointed out as one crucial element for industrial systems, AI implementations and other not-deterministic tools have to be monitored closely in their specialisation and architecture, to avoid the scenario of having all the complexity moved from the physical shop floor towards the technical implementation of the digital architecture. In this regard however, is believed that the expected standardisation brought by the DT architecture, and possible migration of actual manufacturing management tools (already standardised) in the digital domain, should enable to building

AI system with a lower degree of isolation and a higher transfer capability from one production context to the other, lowering as a consequence the overall AI implementation, testing and maintaining costs.

4.5 Embedded AI

The last pattern of interaction between AI and DTs reported in 4.1 and evaluated in the dissertation considers an AI system internal to the DT.

In this scenario, the DT is equipped with an internal AI model, executing its tasks with the information coming from the DT Physical Interface. The model is encapsulated inside the DT, therefore any other software element in any external system does not know about its existence. The AI model is, therefore, part of the internal model of the DT: the DT can be the AI model itself, but a one-to-one correspondence is not mandatory. In reverse, is more likely for a DT to carry multiple internal models concerning its nature, each of which absolves to the specific task it is designed for.

The analysed internal AI model is fed by incoming information of the DT and can output information changing the state of the DT, or trigger some action towards its associated physical asset, following some *intelligence* carried out by internal AI models. Because of its nature, the proposed name for this interaction pattern is *Embedded AI*, and can be considered a step towards the *Cognitive DT* [149]. Referring to the framework offered by White-Label-Digital-Twins (WLDT) [100], used for experiments described in Chapter 5, a possible DT state model can be composed with 4 fields: properties, events, relationships and actions, as described in Subsection 5.2.2. Therefore, the Embedded AI model, accessing incoming DT data, has also the ability to output a certain DT property, or to trigger a determined event after having

analysed DT incoming data, to set or unset a relationship between the DT it is living in and another DT, to determine the availability of one or more actions of the DT, or, lastly trigger one or more actions towards the associated physical asset.

Actions triggered by Embedded AI can be used to autonomously maintain the state of the physical asset between a set of predefined boundaries, as happens with working temperatures of engines or mechanical elements having moving parts. Through the use of the Embedded AI, the system can therefore learn by itself what kind of actions execute to maintain critical parameters between target values. Nevertheless, reported capabilities are useful also for some of the following use cases. Having an AI model aimed at predicting failures of its physical counterparts, it can send to the physical asset a set of actions to put it in a safe mode and, after that, disable a set of probable critical actions that the DT can no more accept, to prevent further damages for the underlying physical asset. Then, watching the evolution of the received data from the physical asset, the Embedded AI can output opposite results to then bring back the DT and the relative counterpart in a fully operative mode. Contextually to the proposed use case, the Embedded AI can output an event that occurred for the model, concerning the received data. This event can then be passed to all other software entities that are listening to it, letting emerge coordination capabilities between the DT, its physical counterpart, and other DTs, physical objects or software entities.

Hence, Embedded AI boosts *augmentation* capabilities of physical assets, bringing intelligence to them even if they are not designed for it. Indeed, their intelligence lives in their own DT, which outputs intelligence results concerning the data provided by the physical asset itself. To this regard, is fostered the idea that AI models internally applied to DT brings not only a cost op-

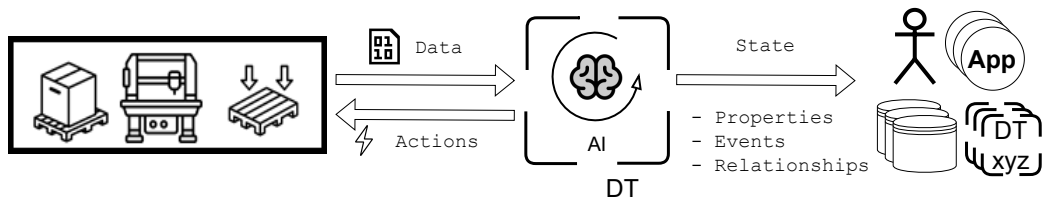


Figure 4.8: Detail about the possible interactions between a DT, its physical counterpart, and the Embedded AI.

portunity ¹, but bring also some improvements under the lens of the division of responsibility in the software context. Indeed, it can be the responsibility of DTs to provide and store physical assets' intelligence, while it is the responsibility of the physical counterparts to provide data via their sensors to be analysed and, in case, modify the real-world state with their actuators. This opens up several scenarios where non-smart hardware, equipped only with connection capabilities, becomes dramatically smart even maintaining not specialised hardware characteristics for this purpose. Moreover, AI models placed in the network can be updated and can evolve in time, a task that can be considered tougher for constrained hardware placed on the edge to collect or action something, or even impossible and not practical.

The pattern of interaction described in this paragraph is depicted in Figure 4.8. Relationships between DTs with Embedded AI and other actors interacting with are reported. The DT state is the reflection of the data coming from the represented physical assets. Received data takes then part in forming the state of the DT, or is possibly manipulated before being exposed as such. The Embedded AI component of the DT takes eventually part in the *augmentation* activity of the DT, which can expose as a consequence a

¹As it can be considered much cheaper to have an AI model developed and living in the network concerning possibly embed it directly on the hardware, maybe also implementing some specialised AI chips

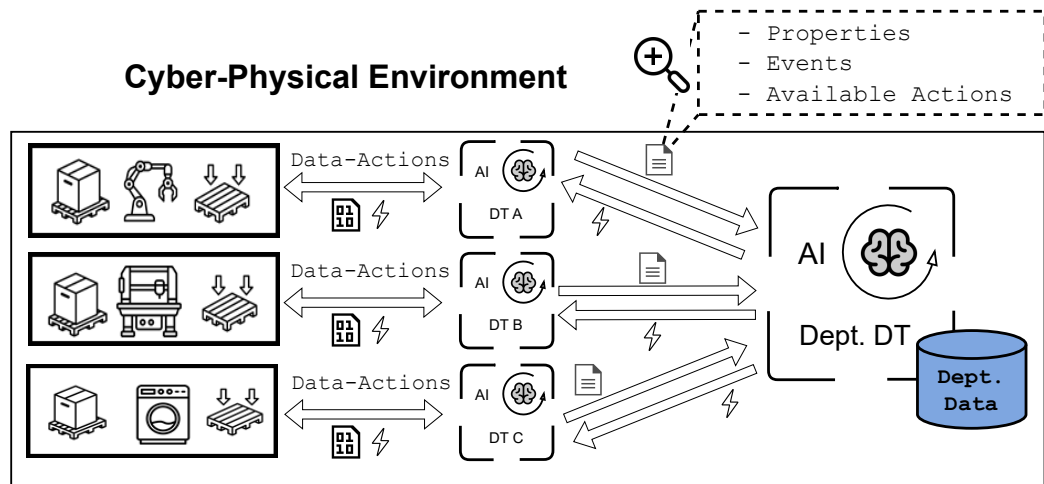


Figure 4.9: Representation of the possible distribution of intelligence among several DTs, and their composition. Each DT can modify the exposition of its internal state representation, and eventually trigger some actions towards underlying counterparts.

set of information that cannot be retrieved with simple data manipulation activities. Enriched information can relate DT *augmented* properties, triggered *augmented* events, *augmented* relationships where the DT is involved, *augmented* capability to enable or disable a set of DT exposed actions. Actions can also be triggered towards the physical assets of the system by the Embedded AI, if necessary for the DT to obtain a certain needed behaviour. This is the case for maintaining a certain state of affairs in the physical domain or obtaining a set of reactions given some physical observations collected by the sensorial part of the physical asset.

Augmented DT state descriptions by the Embedded AI are offered outside of the DT, to other software entities living in the digital domain. Among them, there can be stakeholders, other applications or databases, as for the case depicted in 4.2. Nevertheless, being the Embedded AI stored *inside* the

DT, it is also possible to have another DT, at a higher level, receiving data from one or more DTs with Embedded AI. This scenario is depicted in Figure 4.9, making it easier for the reader to understand how, having a set of different DTs relating to each other, implements an intelligent model embedded in each of them. The Embedded AI can then augment, manipulate and extract more valuable information from the received data from the physical layer, and expose the resulting DT state to composed DTs at a higher level. Then, the composed DT can do the same, receiving data, augmenting it via Embedded AI if needed, and then exposing their state result to other DTs at a higher level, thus repeating the described relation pattern. The consequence of the proposed approach is having multiple intelligent entities communicating, coordinating, and eventually cooperating in representing, exposing and controlling their physical assets counterparts. The resulting system can be considered as a distributed intelligent system of DTs.

The proposed contribution poses a common ground for the efficient management of a collection of smart devices. The need to cope with the physical heterogeneity, fragmentation and complexity associated with industrial systems is tackled by the adoption of DTs, and the transposition of depicted limitations in the digital domain. DTs provide the strategic infrastructure for an effective abstraction and coordination of novel smart devices, the collection and integration of data coming from heterogeneous sources, providing a generalised collection of tools for the uniform digitisation of the whole information process driving the factory of the future [124] [58]. Investigated complexity transposition towards the digital domain offers the possibility to manage interoperability, data collection and interaction of smart devices better than how it has been done until today in the physical industrial domain. Furthermore, DT *augmentation* offers the opportunity to empower new and

existing deployed devices, by modelling the relationships and links characterising the physical assets, enabling their composition into new digital entities that support discovery and learning [32], as well as reasoning upon dynamic strategies for optimisation purposes [1]. All the devices and components acting in such a complex system need to develop the capability to autonomously and dynamically learn how to react and to adapt in front of evolving situations, both at the individual and at the collective level. It is unfeasible to embed smart devices with handcrafted models to be used for decision-making in any critical situation. To develop adaptive skills over time, systems need to build their internal model of the environment. This model should capture the relationships and dependencies among various components and variables. The abstraction level facilitated by DTs plays a vital role in achieving this goal, providing a crucial opportunity for internal AI models as the proposed Embedded AI, to mirror the contextual world they observe through their physical counterparts, as well as to offer high-level information AI systems external to the industrial DTs architecture, as the Observer, Advisor and Controller AI.

Chapter 5

Experimental Evaluation

The proposed relationship patterns between AI and DT promise to change the shop-floor setup as we know. It indeed, the standardised approach in digitising the shop-floor environment as well as the potential of added intelligence offer practitioners a powerful tool to organically digitise the physical environment in any industrial scenario characterised by connected devices. Nevertheless, the implementation of such new synergies between DTs and AI has not only been described but also experimented in 3 out of 4 proposed patterns. The considered patterns are *Observer AI*, *Advisor AI* for the AI placed externally in the DT ecosystem (remembering that the *Controller AI* is a mix the first two), and *Embedded AI* for the AI placed inside a single DT.

The proposed experiments target the interaction between AI and DTs in CPPS, to understand how these relationships can impact the industrial production environment. These experiments explore the identified ways of integrating AI into CPPS represented by DTs to improve efficiency, automation, and resource management. The *Observer AI* experiment explores an AI training system and model *external* from the CPPS. It aims to infer causality

from the data received from the underlying DT ecosystem. Essentially, the AI observes physical events through DTs without understanding their internal structure or data manipulation processes. The gained insights are then shared with external applications or stakeholders without further interaction with the CPPS. Similarly, the *Advisor AI* experiment operates externally from the CPPS but receives information from external sources. Its objective is to interact with the CPPS to achieve specific outcomes in the physical domain, sending action requests to a target DT of the system, or a composition of target DTs. As in the previous experiment, the *Advisor AI* does not know anything about the internal structure of each DT. Is, instead, aware of the DT-ecosystem structure. The *Embedded AI*, instead, considers an AI model *internal* to a DT, aware only of the information received by the DT itself and, therefore, internal to its structure. This experiment illustrates how an internal AI system can manipulate contextual data within the DT for various purposes such as data augmentation, monitoring, or triggering actions. Indeed, it can prompt smart action requests towards underlying DTs or directly on associated physical assets. The first experiment, discussed in Section 5.1, is conducted in an industrial simulation environment, while the second and third are built on top of a real DT-ecosystem representing a physical production system. This second group of experiments have the additional target of demonstrating the DT-ecosystem’s ability to abstract industrial physical assets and provide a standardised representation of the production system. Moreover, experiments involving the *Advisor AI* and *Embedded AI* leverage the outcomes of the DT architecture, maintaining consistency in the physical structure. This demonstrates the decoupling capabilities, homogenisation of digitised assets, and standardisation in building the CPPS.

The physically tested scenarios are proposed and discussed in the follow-

ing sections, as a contribution to concretely depicting the potential digital structure of the factory of the future.

5.1 Informational AI and Causality

The first pattern proposed is the Informational AI, which considers an AI system monitoring an entire production department, as a support for practitioners, such as managers, and external applications, as task schedulers.

Nowadays one of the challenges of AI is to understand causal relations between variables of a complex system [111] to allow the interpretation of choices and decisions made by intelligent systems and improve generalisation skills of smart devices across different situations. Following this trend, a causal learning application has been implemented on top of an industrial environment.¹ Causal learning is a powerful tool in industrial environments, as it speeds up activities such as root cause analysis, lowering the complexity barrier (especially the dynamic complexity) to practitioners. Moreover, having an industrial physical system decoupled from its digital representation, decouples also the associated learning activities, having therefore a much easier-to-handle system representation and lowering, as a consequence, the static complexity representation of the system in the digital domain. In an industrial setting, typically, some of the variables within the environment can be modified by the actions of the devices, while others are clearly out of their control, and can only be observed by agents. The proposed scenario follows the ideas of Judea Pearl [97] who introduced the concept of the “ladder of causation”, defining a hierarchy of causality levels that can be developed by an intelligent agent. The first level of the ladder consists of simply detect-

¹In a sense, this can be seen as an instantiation of eXplainable AI (XAI) [6, 24].

ing relations as associations (correlations), whereas the second one assumes the possibility of intervening in the environment and observing the (causal) effects of the actions taken. Finally, the third level enables reasoning and planning based on counterfactual analysis, using do-calculus to learn causal dependencies directly from data observations.

Bayesian networks (BNs) are one of the most widely used frameworks to represent interpretable models of the world [62]. A BN is a directed graph where nodes represent variables, and edges represent dependencies amongst them. Two variables that are not connected by an edge are conditionally independent one from the other. Each node (variable) is associated with a Conditional Probability Table (CPT) which defines the conditional probability distribution of that variable, given the variables which it depends upon (i.e., its parent nodes). In the context of a smart factory, nodes can represent state variables of devices, i.e., the last signal emitted by a sensor, the fact that a bearing is currently faulty, and the level of a production buffer. Edges represent relations that naturally occur between variables whose value affects each other: a faulty bearing might be responsible for a starving machine, whose buffer will therefore be stuck at the same level for some time.

Three main problems can be characterised when dealing with BNs: inference, parameter learning and structure learning. In case the BN is available (both the structure and the CPT values are known) then it is possible to perform probabilistic inference over it, that is to compute the probability distribution of some of the variables, by giving specific values to others, i.e., observing the chain of dependencies propagating through the graph. In some cases, instead, only the structure of the BN might be known (e.g., designed by a domain expert) whereas the parameters of the CPTs are to be learned from data observations. This is typically done via maximum likelihood esti-

mation from a training set [62]. Finally, a much more challenging scenario is the one where not even the structure of the BN is available, but it should also be learned from data.

Although they do not explicitly model *causal* links, but only conditional dependencies amongst variables, BNs are still widely used as a building block of causal models. In fact, causality can be inferred with *interventions* in the environment, i.e., via do-calculus [98]. Specifically, given a direct link $X \rightarrow Y$ between two variables X and Y , one can say that X is a cause for Y if the probability of Y is affected by a direct intervention on X . By *imposing* a specific value for X , one can observe the changes in the probability distribution of Y , and deduce the presence of a causal relation accordingly. A causal relation can be assumed to hold between X and Y if $P(Y|do(X))$ is significantly different from $P(Y)$, where the *do* notation is used to indicate that is forced a specific value assumed by X . In the considered application domain, this mechanism can be exploited to infer that a fault occurring in a given component is indeed the *cause* of some unexpected behaviour downstream of the manufacturing process or that, conversely, an increment in the production is due to certain situations and conditions occurring in the workflow.

Fault detection and root cause analysis are classic applications of causal discovery in smart factories [131]. Lacking a full exploration of causal chains in an industrial environment can easily lead to sub-optimal decisions that fail to solve the issue and have consequences on the whole production [130]. This is especially true in manufacturing environments, where the output carried out by an activity in a certain workstation strictly depends on what happened in the previous steps of the supply chain. With increasingly faster production times, quickly understanding the origin of a certain problem can

indeed shorten the root cause analysis time.

If the chain of causality and its propagation paths through the shop floor are known, activities can be dynamically re-scheduled in all the machines that have been hit by the upstream breakdowns [68].

To test the proposed architecture, a manufacturing environment is simulated. The whole environment has been represented via a high-level DT embedding key characteristics of the involved machines and tracking the evolution of its happenings. The goal of the experiment is twofold: (1) to learn a causal model of the manufacturing operations directly from data observations stored in the machine logs produced by all the DTs involved in the process; (2) to evaluate the impact of the use of DTs on reducing the digital complexity.

5.1.1 Production System Model: Description & Terminology

Each production system can be considered as a system composed of different sub-systems. Essential elements are an input warehouse, a shop-floor where the input material is transformed somehow, and an output warehouse. The same pattern can be recognised in the shop floor: each production node is composed of an input buffer node, a processing node (which can be a transforming or an assembly stage), and an output buffer node. Each of them is in turn composed of different elements (which can be other buffers, many sub-processing stages and so on), breaking down the system into physical equipment. Therefore, it is possible to assume that a certain aggregation of physical equipment composes the input buffer node, the processing node and the output buffer node. An aggregation of input buffer, processing and output buffer nodes composes a production node. One or many production

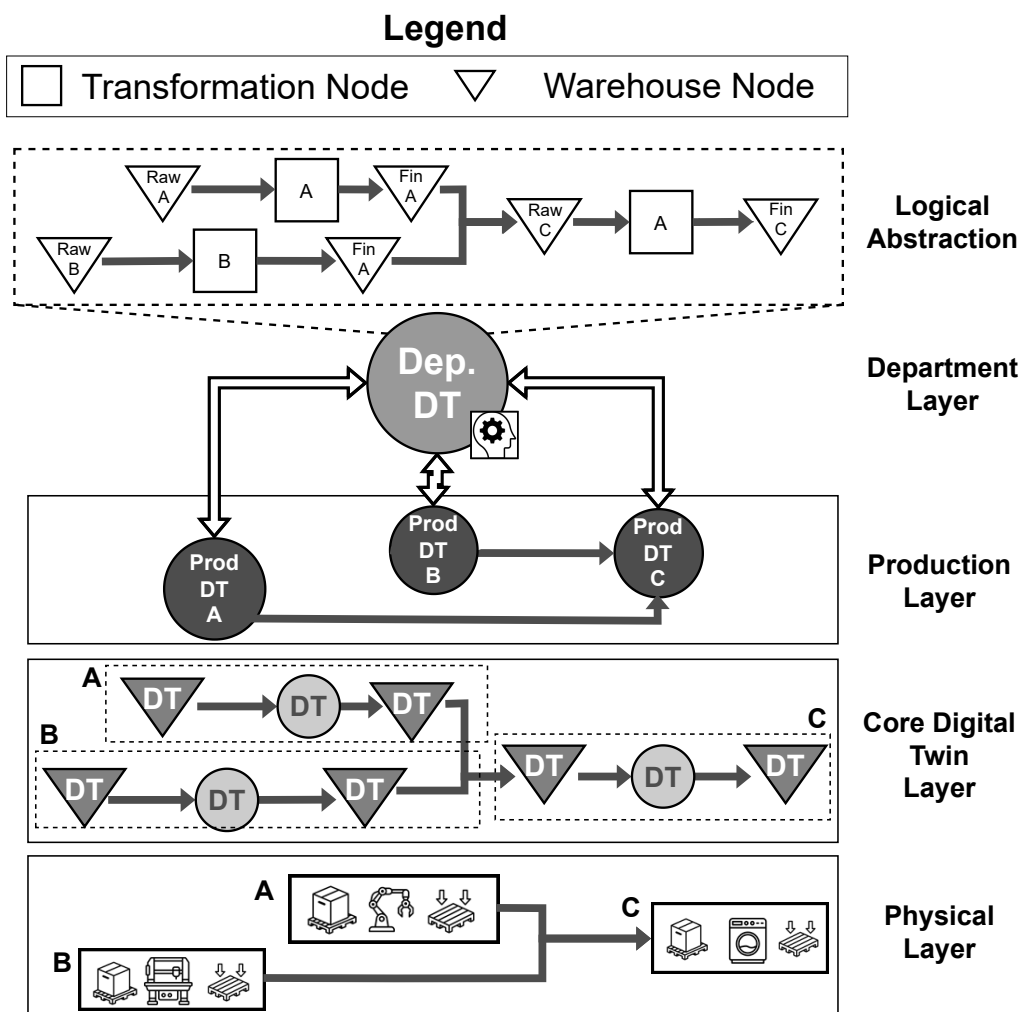


Figure 5.1: Logical representation of the simulated production system.

nodes form the shop floor. An aggregation of production nodes all aimed at reaching the same target composes a department. An aggregation of different departments composes a business unit. An aggregation of different business units composes a production plant.

The simulated scenario consists of three production nodes, named A, B and C, associated with several physical devices (see Figure 5.1, top and bottom). Following the aforementioned pattern, each production node x is

composed of three elements: (i) an input buffer node, where raw material is stored, with capacity IC_x ; (ii) a processing buffer node, where raw material is processed; (iii) an output buffer node with capacity OC_x , where finished material is stored. Each of these components is associated with a specific DT. When the level of input buffers for A and B is below a certain threshold, the buffers are refilled.² The input material picking process has been modelled taking into consideration the material handling time H_x (for machine x) which is also collected by the DT. The output material disposal process follows the same behaviour as the input process. Nodes A and B produce semi-finished parts which are inputs for node C (e.g., nodes A and B produce a gear and a shaft, respectively, using a computerised numeric control machine, whereas node C assembles the gear on the shaft). Node C can produce if and only if both input pieces are available.

During the operation phase, a production node breakdown may occur. The breakdown can happen at any moment during the operation of the node, i.e., during input material handling, during material processing, and output material handling. In that case, the whole node operations are stopped until the node is restored, that is after the needed repairing time. As mentioned before, node C can produce only if its input buffer is filled with at least one piece of material coming from both A and B. When the input buffer for C is populated, then C operations start. If no material is found in the input buffer, the associated operation stage does not start. If the output buffer is full, the operation stage stops and waits until some piece is removed.

Figure 5.2 shows our experimental workflow. The production nodes generate simulated data through the use of DTs and contribute to a data set in

²The details of this refilling process have not been considered as they are not relevant for the simulation purpose.

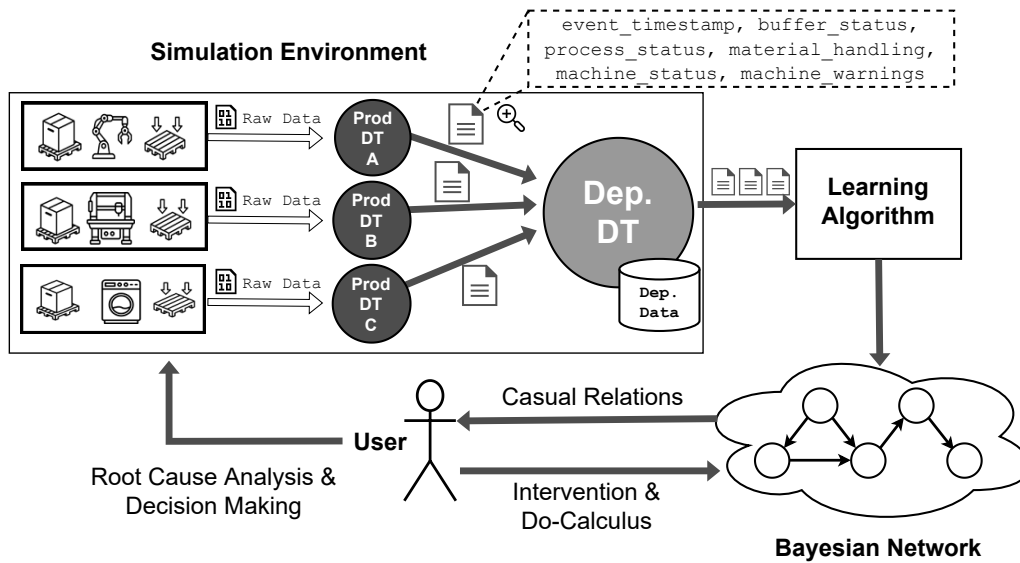


Figure 5.2: Workflow of the experimental setting. The production nodes, using DTs, generate simulated data in the form of log files. That data is used to learn a Bayesian network, upon which users can perform intervention and do-calculus to infer causal relations. This information can be exploited for root cause analysis and decision-making.

the form of plain log files. This data is used to learn a Bayesian network, which users can interact with to perform intervention and do-calculus and infer causal links to perform root cause analysis and decision-making.

5.1.2 Industrial Simulation Setting

We implemented the case study in Python, with the SimPy library.³ All the code and data used to reproduce the experiments are publicly available on a GitHub repository.⁴ As already said, we assume data to be stored in the

³<https://simpy.readthedocs.io/>

⁴GitHub repository of the experiment: <https://github.com/matteo-martinelli/Enabling-causality-learning-with-Digital-Twins>

Parameter	Description
T	Number of simulation seconds
$MTTF_x$	Mean Time to Failure for machine x
$MTTR_x$	Mean Time to Repair for machine x
H_x	Material handling time for machine x
μ_x^P, σ_x^P	Processing time (mean and std) for machine x
IC_x	Input buffer capacity for machine x
OC_x	Output buffer capacity for machine x
Δ	Tolerance threshold for considering machine C as starving

Table 5.1: Summary of parameters considered in the simulation.

form of a log file, continuously updated by the DTs during simulation. The stored information includes:

1. the logged data timestamp;
2. the levels of input raw material buffer and output processed material buffer for each machine;
3. the number of pieces totally processed until that moment for the i -th machine;
4. a Boolean flag F_x to indicate an occurring event failure for machine x ;
5. a Boolean flag Q_x indicating whether the output buffer of a machine has recently received at least one piece (i.e., to detect whether the machine is currently producing).

The last two pieces of information will be particularly relevant for the case study, to identify causal relations between the failure of a specific machine

and the missing production for that machine, as well as for others. For example, machine C can be in an unproductive state even without being in a faulty state, in case of failure for either machine A or machine B. This information is obtained with a simple control: considering machine x , if after a time equal to the mean processing time plus a tolerance Δ , a new piece is not placed in the output buffer of x , then flag Q_x is raised. The flag is reset to zero as soon as a new piece is placed in the output buffer. This control over flag Q_x is performed by each DT supervising the corresponding node in the production model.

Table 5.1 summarises the most important parameters considered within the simulation. We identified a reference setting to obtain a solid and realistic behaviour for the simulated scenario and to highlight the potential of the learning mechanism. Thus, there are no bottlenecks in the production process, all machines have similar production rates, and the frequency of breakdowns has been chosen to produce a sufficient number of examples to train the system in a limited amount of time. The processing time is modelled as a time delay with a stochastic behaviour following a normal distribution: for machines A and B has been set $\mu_x^P = 250$ and $\sigma_x^P = 15$; for C has been set $\mu_C^P = 230$ and $\sigma_C^P = 10$. For the sake of simplicity, the material handling time H_x has been considered as a deterministic parameter, modelling an automated material handling. Breakdowns are characterised by randomness and, thus are represented by random variables following an exponential distribution. For nodes A and B, the MTTF is 77.760 and 86.400 seconds, respectively, whereas the MTTR was set to 10.800 seconds and 12.000 seconds, respectively. For node C, the MTTF was set equal to 100.800 seconds and the MTTR equal to 1.920 seconds. The input and output buffer capacities IC_x and OC_x were set to 500 pieces. The tolerance threshold used to raise

flag Q_C and to indicate that machine C is starving has been set to $\Delta = 243$ seconds (the average of the production rates of the three machines).

Data is generated by simulating 36 working days with a single working shift per day, for a total of $T = 1.250.823$ seconds. After simulation, data is stored in a plain text file (comma-separated values) to be used for causality analysis.

5.1.3 Learning Causality of Failures

To learn causal models has been used `CausalNex`,⁵ a Python library that allows to perform learning and inference in Bayesian networks, and also do-calculus, to observe the effect of interventions and deduce causal links between the observed variables. `CausalNex` allows us to learn both the structure and the parameters of Bayesian networks directly from data observations. The default algorithm is *NOTEARS*, an efficient state-of-the-art method that exploits continuous optimisation to avoid combinatorial search [146].

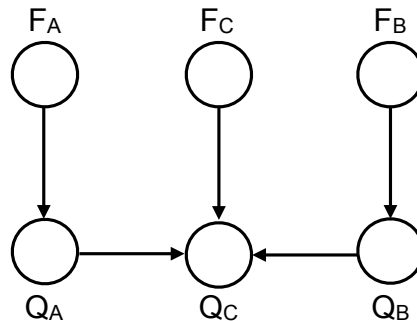


Figure 5.3: Causal network learned from simulation data. F_A , F_B and F_C are the random variables representing machine faults. Q_A , Q_B and Q_C represent the three Boolean flags indicating whether machines are currently producing.

We used a data set containing 1.250.823 examples and six variables:

⁵<https://causalnex.readthedocs.io/>

namely, flags F_x and Q_x for the three machines A, B, and C. The only hyper-parameter of the learning algorithm is a threshold value ω that is used to prune “close to zero” edges in the causal model. Following the empirical suggestions in the original paper [146] has been set $\omega = 0.3$. Since, in our simulation, has not been modelled any variable (either environmental or internal) that can be the cause of fault for either machine, was possible to set an additional constraint in the structure learning algorithm, by imposing F_x nodes to be parent nodes only. The overall data set is collected by the department-level DT (Figure 5.1) which has the vision of the whole system and activates the learning process.

To assess the causal nature of the links represented in the Bayesian Network, interventions and do-calculus have been exploited: a given node was forced to take a certain value, and then the changes were observed in the distribution of all the connected nodes. This kind of operation can be seen as a sort of causality test, by checking what is likely to happen, given the input data, in case a certain state is reached by a system node. In other words, it highlights the *cause-effect* relations between connected nodes.

The learned structure of the Bayesian network is depicted in Figure 5.3. It easily shows that the model correctly captures the links between the failures of machines A and B and the stop in production for machine C.

To validate the correctness of the learned model, a standard classification task on the nodes of the Bayesian network was performed. The data set was split into a training (90%) and a test set (10%), the model parameters were learned on the training set, and classification was performed on the test instances, using each of the variables, in turn, as the prediction target. Performance was measured using the Area Under Curve (AUC) metric [14]. For nodes F_A , F_B and F_C have been obtained 0.85, 0.86 and 0.99, respectively.

For nodes Q_A , Q_B and Q_C have been obtained 1.00, 1.00 and 0.98, respectively. These figures show that the learned model can be effectively used to perform inference.

Then, to infer causal relations from the network, two subsequent experiments have been performed. First, each machine was considered independently, and the change in the probability distribution for flag Q_x after forcing a failure of the machine was analysed, thus an intervention on F_x by setting its value to 1. Formally, $P(Q_x|do(F_x = 1))$ with $P(Q_x)$ for $x = \{A, B, C\}$ was compared. Results are reported in Table 5.2. For all the machines an evident change was observed in the distribution after intervention, which is a sign of causal dependency between the variables. For machines A and B, flag Q_x is almost always up in case of machine fault (probability around 97%, last row, which means that both machines typically break for a time longer than the mean production time (including tolerance Δ). This probability is slightly lower for machine C, indicating that machine C stops for a breakdown only in 78% of cases. In all the other cases, the breakdown is too brief for being registered by flag Q_C : in fact, the MTTR for machine C is smaller than that of A and B.

	$x = A$	$x = B$	$x = C$
$P(Q_x = 0)$	0.846	0.864	0.811
$P(Q_x = 1)$	0.154	0.136	0.189
$P(Q_x = 0 do(F_x = 1))$	0.028	0.033	0.220
$P(Q_x = 1 do(F_x = 1))$	0.972	0.967	0.780

Table 5.2: Analysis of do-calculus over each machine independently. Considering the probability of Q_x being raised in case of fault of the corresponding machine, indicated by F_x .

Then, the impact of upstream machines A and B over machine C was evaluated, by analysing $P(Q_C|do(F_x = 1))$ with $x = \{A, B\}$. Results are reported in Table 5.3. Even in this case, the change in the distribution is a strong indicator of a causal link. When either machine A or B fails, machine C is getting to starve, as indicated by the rise in flag Q_C . The propagation is slightly more frequent for machine A (69% against 64%), due to the larger MTTF and MTTR values concerning machine B.

	$x = A$	$x = B$
$P(Q_C = 0)$	0.811	0.811
$P(Q_C = 1)$	0.189	0.189
$P(Q_C = 0 do(F_x = 1))$	0.309	0.363
$P(Q_C = 1 do(F_x = 1))$	0.691	0.637

Table 5.3: Analysis of the impact of upstream machines A and B over C, via do-calculus. The probability of Q_C being raised in case of fault of either A or B was considered, indicated by F_x .

Besides these interesting results, two limitations of the experimental setting need to be underlined. First, the structure learning algorithm needs to set the ω hyper-parameter: in the original article describing the approach [146] a value $\omega = 0.3$ is suggested, but there is no real connection concerning the application domain. Secondly, another crucial point is the tolerance Δ that evaluates when a piece is considered missing in the output buffer. This parameter is very important for the system to work, and, ideally, needs to be set to model when a piece is missed by a given probability. Such threshold can be seen as a sort of trade-off between false alarms (i.e., situations where no fault is actually in progress, but the production is just slower than usual) and missing detection (i.e., when a faulty scenario is not

recognised by the system).

5.1.4 Impact of Digital Twins on Digital Complexity

The second goal of the experiment conducted in this scenario is to shed light on the impact of DTs in reducing the physical and *digital complexity* of the environment. The complexity of the scenario was estimated with and without their adoption to assess their impact in terms of digitisation and decomposition of responsibilities. The following metrics have been considered:

- *Required Protocols (p)*: the number of application layer protocols required by a digital application or service to interact with the deployed physical assets to collect data and send commands.
- *Communication Patterns (c)*: the number of communication patterns needed to interact with involved devices and platforms (e.g., Publisher/Subscriber or Request/Response);
- *Data Formats*: the number of different data formats, serialisation and information representation techniques required to read and send data from and to deployed devices;
- *Interaction Points (n)*: the number of different modules, services or platforms that an application should interact with to retrieve all the target data or consume services;
- *Aggregation Points (a)*: the number of aggregation or composition levels required to abstract the physical world into the right level of complexity concerning the observers' typologies and their application goals (e.g., merging information and telemetry data from machines in the same production line).

Each criterion has been assigned a specific Importance Factor (IF) ranging from 1 (lower) to 3 (higher), to be used as a multiplicative score for the Digital Complexity Indicator (DCI) of that criterion, according to its impact on the development, deployment and maintenance of the solution. The DCI is directly proportional to the exposed digital complexity that an external application has to manage to communicate with deployed physical assets, collect data and send commands. The $DCI = \sum_{i=1}^5 criterion_i \times IF_i$ represents the digital complexity level that the external application has to face concerning the physical use case

Typical configurations coming from the IoT and IIoT technological approaches and characteristics are considered. To describe the impact of adopting DTs, 3 example use cases are proposed:

- *Use case 1 — One Production Line with Two Heterogeneous Machines* ($p = 1, c = 1, m = 2, n = 2, a = 1$). This scenario was assumed to have 2 different machines using the same communication pattern and a common application layer protocol (e.g., Publisher/Subscriber with MQTT). The two machines generate different data formats associated with their specific characteristics, for example in terms of functionalities and generated telemetry information. From an application perspective, the interaction points are 2 while the aggregation point is 1 to digitise the overall production line by merging information of the active devices. The resulting DCI is reported in Table 5.4, showing that the adoption of DTs brings a reduction of around 50%.
- *Use case 2 — Two Production Lines with Four Heterogeneous Machines* ($p = 2, c = 2, m = 2, n = 4, a = 2$): In this use case was assumed to have two production lines, each composed of two machines.

The two machines inside a single line are different, but the two composed lines are identical, running in parallel. The machines adopt 2 different protocols and 2 different communication patterns (e.g., Publisher/Subscriber with Message Query Telemetry Transport - MQTT - and Request/Response with ModBus) and expose a different and customised data format. Each machine represents a single interaction point, whereas the two lines are considered independent aggregation points for observing and interacting applications. The DCI is reported in Table 5.4: using DTs it decreases from 26 to 8, with a reduction of around 70%.

- *Use case 3 — One Department, Three Production Lines and Five Machines ($p = 2, c = 2, m = 5, n = 5, a = 4$):*. This scenario is inspired by the case study proposed so far. Five different typologies of physical assets deployed through three production lines and one department were considered. A conservative hypothesis is made with the adoption of only two application-layer protocols associated with two different categories of communication patterns such as Publisher/Subscriber and Request/Response (e.g., MQTT and ModBus). In this context, even if we can have common shared protocols, the adopted data formats and the interaction points can be reasonably different and customised for each type of deployed machine through the use of specific configurations and modules. Furthermore, concerning the required aggregation points, is necessary to take into account the responsibility to merge information of each production line (A, B and C) and to build a unified view of the entire department. This is a key contribution to our hierarchical architecture. The DCI is reported in Table 5.4 as well as depicted in Figure 5.4, showing that in this scenario the complexity

Criterion	IF	Use Case 1		Use Case 2		Use Case 3	
		w/o DT	DT	w/o DT	DT	w/o DT	DT
Required Protocols (p)	2	1	1	2	1	2	1
Communication Patterns (c)	1	1	1	2	1	2	1
Data Formats (m)	3	2	1	2	1	5	1
Interaction Points (n)	2	2	1	4	1	5	1
Aggregation Points (a)	3	1	0	2	0	4	0
DCI	-	16	8	26	8	43	8

Table 5.4: Estimation of the Digital Complexity Index for the 3 proposed use cases. Without DTs, DCI grows with the number of involved devices and protocols. Instead, DTs decouple the digital complexity from their digital counterpart, simplifying the interaction with external applications.

reduction is around 81%.

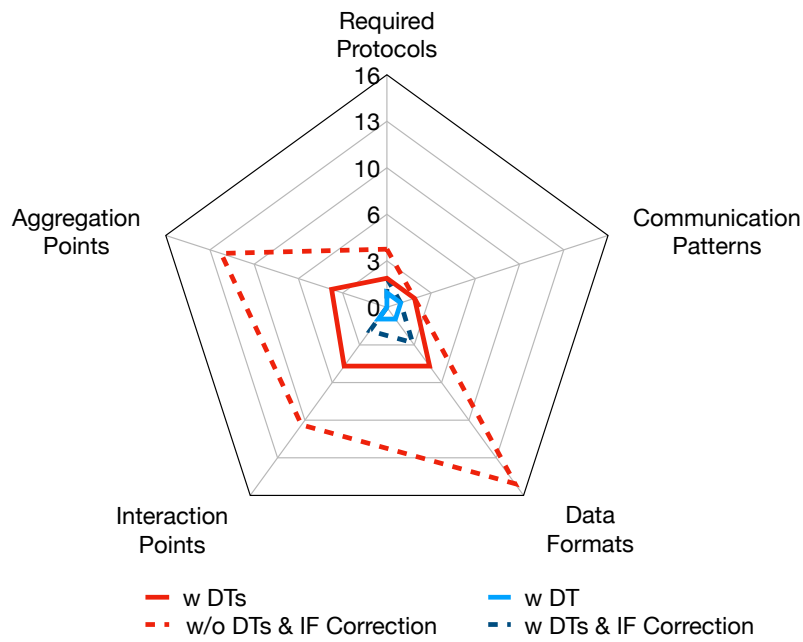


Figure 5.4: Graphical representation of the Digital Complexity Index for our scenario, taking into account results with and without the correction of the Importance Factor (IF).

The proposed analysis shows how the depicted approach and the introduction of multiple hierarchical DTs allow the building of an effective uniform digital abstraction reducing the perceived heterogeneity in terms of protocols, communication patterns and data format thanks to the adoption of a shared communication layer and a unique representation of the information. It is important to highlight how, thanks to the introduction of DT layers, the cyber-physical complexity perceived by an application willing to interact with the machines, production lines or the department is always the same ($DCI = 8$) since the increased physical and digital heterogeneity and the associated management cost is hidden behind DTs. This management is not only associated with low-level technological aspects (such as communication patterns and protocols) but it massively relies also on the possibility of effectively mapping and modelling existing physical relationships (e.g., two machines interacting on the same production line), asset composition (e.g., multiple production areas with their machines belonging to the same department) and to properly structure the knowledge in a uniform and exploitable way, reducing interaction points and simplifying aggregation.

5.2 Experimental Environment & DT Implementation

Industrial production environments are complex systems characterised by a multitude of stakeholders. Several aspects are often considered in such environments, *technical* and *business-wise*. Among those aspects, are safety, production performance and cost tracking, professional maintenance, quality control, logistics, equipment management, environment, product design, management, life-cycle, and so on. Oftentimes, one or more managers are

in charge of monitoring them, improving their performance, facing unexpected events, and being responsible for reaching business targets. What is in common between all stakeholders is that information they are interested in comes from the same source, i.e. the shop floor. Nevertheless, it is expected that, even for the same manager given a niche of interest, collecting all the needed data and organically representing them would be impossible due to the high heterogeneity expected in processes, documents, hardware, and software to satisfy needed tasks. Moreover, usually, shop floors are not static environments, experiencing little but continuous change due to continuous process optimisations or updates in product demand requests.

As explored in this dissertation, in such a dynamic and complex environment, having a layer of abstraction between the real world and the digital domain, and between different views of the same considered context, organically representing the world state of affairs is desirable, and becomes necessary the more the system grows in complexity. DTs are a candidate technology to suit as a physical abstraction layer in a production environment [61], as reported also in paragraphs 3.2 and 4.1. The required set of features for industrial DTs should make them suitable for representing the industrial domain structure in real-time, and expose this structure in the digital domain to applications and stakeholders organically. Required features have been described in paragraph 3.4 and are briefly recalled in the following. Usually, industrial architectures are divided into sub-components, i.e. an entire production plant is divided into departments, and departments are divided again into working areas or production nodes. Production nodes are then constituted by a set of heterogeneous hardware equipment, that could be both automatised in the case of a highly automated node or manually operated [50] [93].

Between different production nodes at the department level usually exists relationships that need to be exploited, e.g. the assembly nodes come most of the time at the end of a production system. Moreover, relations are also present at the single production node level, e.g. a robotised production node is usually equipped with an input buffer, containing one or two material pallets, a machine operating the needed processes on the material, an output buffer, and a robotic arm handling the material. As a final point, components and relationships can also change over time, due to standard operation activities or shop-floor upgrades. It is therefore clear that the digitised layer of abstraction needs to support composability and relationship capabilities. DTs are a viable solution to build and maintain this abstraction layer, supporting an architecture structure that can represent the shop-floor state of affairs organically in the digital domain, augmenting shop-floor capabilities, offering a homogeneous level of access to data to external applications, and supporting composability and relationships by-design. Augmentation capabilities instead, are basically required to enable data manipulation and information extraction from data received by physical counterparts. Finally, actionability enables the opportunity to modify the physical counterpart context through physical actuators by passing action requests to their associated DTs. DTs, then have the responsibility to analyse, confirm or reject the request, eventually break down it into sub-actions, and finally pass the received request to targeted underlying DTs or physical components, acting therefore as an interface between the physical context, the digital application domain and system stakeholders.

The described characteristics have been organised as 5 capabilities, as described in Section 3.4. Moreover, they have been implemented and tested in a practical use case involving a hierarchical architecture of composed DTs.

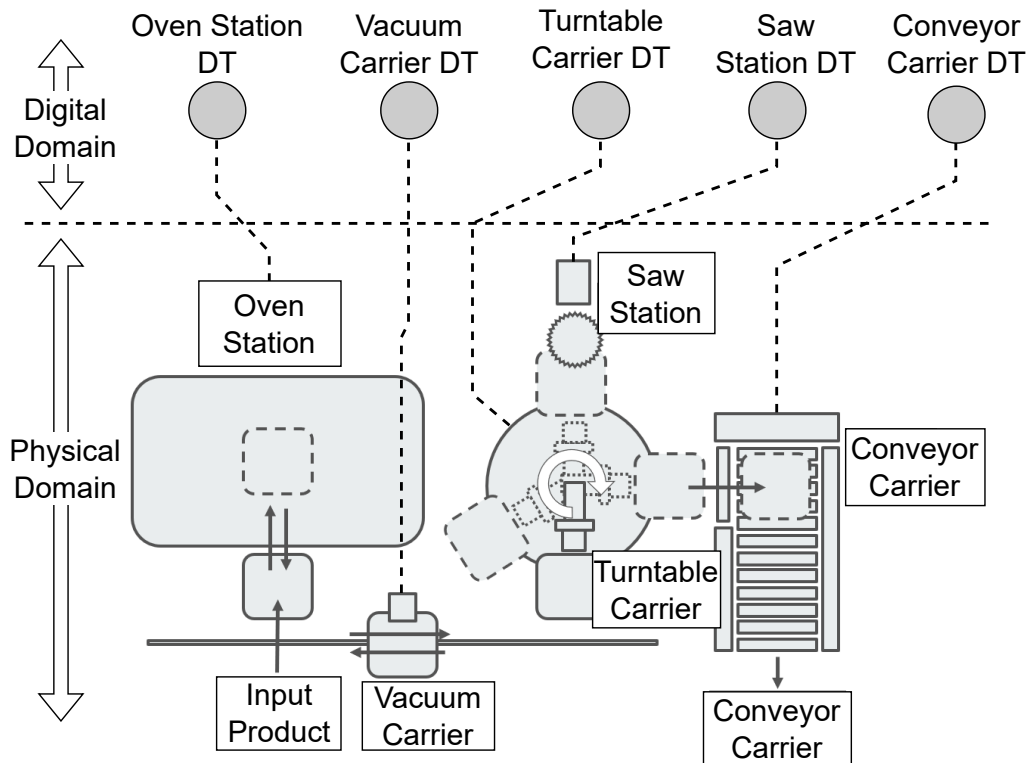


Figure 5.5: Schematic of the *Fischertechnik Multiprocess Station with Oven* industrial system and of the associated first level of DTs.

5.2.1 Experimental Setup

A practical implementation of a twinned industrial scenario implementing composed DTs has been performed, to show a functioning application of principles analysed until now in the dissertation. In the implementation the manufacturing system was represented by the *Multiprocess Station with Oven* module of the *Fischertechnik Training Factory Industry 4.0* research equipment, schematised in Figure 5.5 and photographed in Figure 5.6. The module is characterised by 5 machines disposed of in series imitating a flow shop layout, with 3 material handling machines (a vacuum gripper carrier, a turntable and a conveyor) and 2 transformation stations (an oven, and a

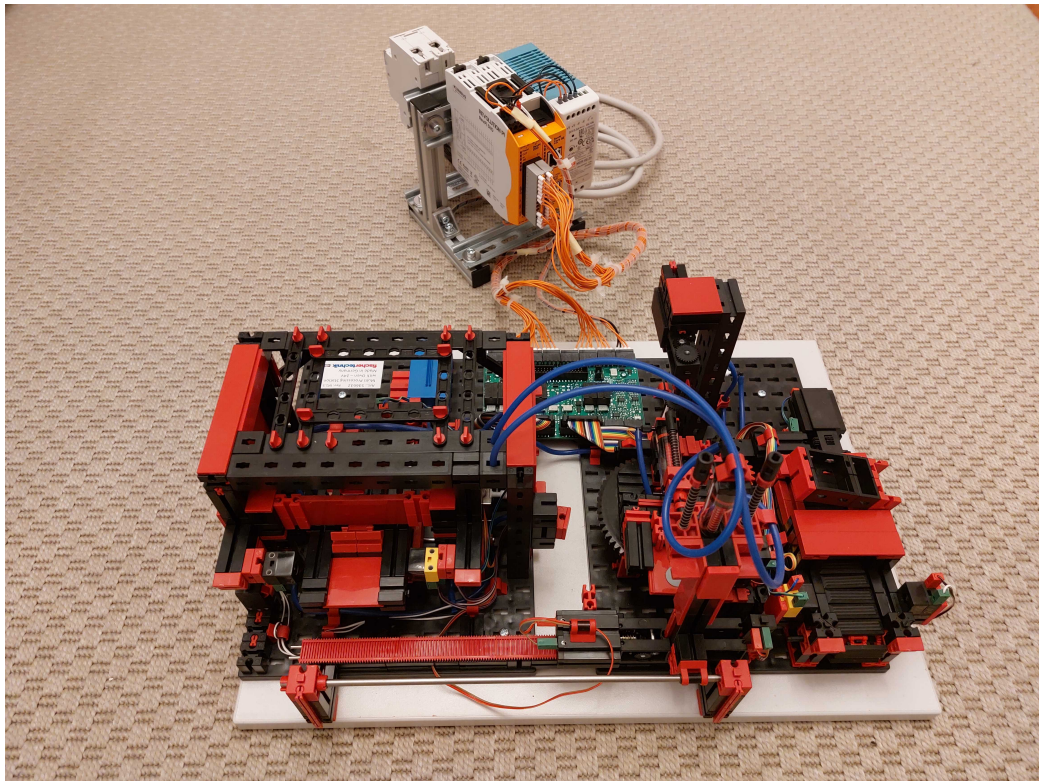


Figure 5.6: An actual picture of the *Fischertechnik Multiprocess Station with Oven* industrial system. It is directly connected and guided by a Revolution Pi, via a 24V industry-grade digital I/O interface.

saw station).

Machines are composed of a set of sensors and actuators. In particular, in the considered module there are 2 types of sensors (light barriers and limit switches) and 2 types of actuators (actuators with 2 operative states, i.e. on and off; actuators with 3 operative states, i.e. off, move towards thing A, move towards B). Sensors and actuators are controlled by a 24V industry-grade digital board.

On the controlling hardware side, 2 computers have been used: a Raspberry Pi-based soft-Programmable Logic Controller (soft-PLC) directly ma-

noeuving the Fischertechnik factory replica equipped with a 24V Digital Input-Output expansion board and a laptop.

The software is instead organised with 3 components controlling, representing and exposing the manufacturing system to the digital domain. Implemented components are the manufacturing process control component, an MQTT broker, and the DT component. The DT component has been implemented using functionalities offered by the WLDT framework, a library for digitising physical IoT assets into the digital domain [100]. The Raspberry Pi soft-PLC hosted the software responsible for controlling the Fischertechnik, while the laptop hosted the DT software. The MQTT broker has been placed on the Raspberry Pi soft-PLC: nevertheless, the choice to place it on the same hardware controlling the Fischertechnik manufacturing model is not mandatory and has been made only for convenience in this experiment. Generally speaking, the only piece of software that really needs to stay in a specific place is the manufacturing process control component. Indeed, this piece of software directly interacts with the underlying hardware via the 24V communication interface. The other two pieces of software could have been on the Raspberry Pi itself, or on the laptop, disregarding the amount of resources available.

The Raspberry Pi software controls the production system, i.e. coordinates machines through sensors and actuators to achieve the production of a single piece as expected. It is worth pointing out that the production system needs to be able to produce despite other systems' state, as the DTs architecture is placed on top of it. In this context, DTs work side by side with the production system and can improve capabilities of the production system. Real-time data about the process state is published by the process Raspberry Pi software on the MQTT broker. Then, the MQTT broker takes care

of transmitting the data to its subscribers which, in this case, are the DTs associated with each machine of the department. Finally, the production control checks at each production loop the *configuration* for each machine and for the whole department, which is also published on the MQTT broker it is connected to.

5.2.2 Machine-Layer Digitisation: Adapters Structure

For each machine in the production system, a DT has been implemented, categorising the layer as “Machine Layer DT”. While it is not mandatory to have a single DT per machine, in this case, a one-to-one mapping has been adopted for simplicity. Each Machine Layer DT is equipped with an MQTT Physical Adapter on the Physical Interface side, a Core component including one or more Shadowing Functions, and HTTP and MQTT Digital Adapters on the Digital Interface side. Mqtt Physical Adapter extends the generic Physical Adapter and focuses on MQTT communication between the physical domain and DT instances. The responsibility of the MQTT Physical Adapter is to monitor topics of interest published on the MQTT broker by target devices, and manage the data retrieval and passing to the DT Core.

In the depicted use case, topics on the MQTT broker are grouped by machines, and twins Physical Adapters subscribe to the topic related to the machine of interest. Information tracked by machine twins is related to each machine sensor and actuator state. In WLDT, the DT state is defined by the *Physical Asset Description* or PAD. The PAD is needed because the DT, in its instantiation phase, does not know anything about the underlying physical asset that is going to be represented. Therefore, a physical asset needs to be described somehow, to let the DT know what are the incoming data about and what kind of actions the physical asset can accept. The PAD

describes the physical asset information following 4 keys, that are then used to generate the DT state. Considered keys are properties, events, relationships and actions. Properties tracked by machines' Physical Adapters are related to sensors and actuator states. Events tracked are instead referred to 2 fields published under each machine topic: *product-on-carrier* and *process-completed* fields. The *product-on-carrier* field might or might not be directly connected to a change in the state of a sensor, e.g. the oven has a light barrier that if crossed highlights the presence of the product, the saw product presence is instead inferred by the overall state of the department. The *process-completed* field is instead directly written by the Raspberry Pi process controller when the workings on the i-th machine for the j-th piece are completed. Relationships set at the DT machine level refer to the *previous-machines* and *following-machines* in the department, and the department where the machine *belongs-to*. To enable navigability through different DTs, relationship fields hold the HTTP or MQTT address referred to as the Digital Adapter of the considered machine.

Information handled by the physical adapter of the DT is then passed to the Core of the DT, which then executes its own Shadowing Function, possibly augmenting the state of the DT from received data, and executing other needed operations concerning the target model scenario. Core details of the DT are described in Subsections 5.2.3, 5.2.5, and 5.2.4.

Digital Adapters instead expose state, events, relationships and actions of the DT to external applications interested in interacting with the DT. Each machine DT implements 2 types of Digital Adapters: an MQTT Digital Adapter and an HTTP Digital Adapter. The MQTT Digital Adapter extends the general Digital Adapter, specialising its capabilities in publishing the DT information on an MQTT broker. The same broker mentioned before has

been used to publish machine-lived DTs' data, which are then grouped under a different topic concerning the data published by the underlying process control software.

HTTP Digital Adapter, instead, extends to the generic Digital Adapter, exposing endpoints corresponding to the DT nature: as the DT is a representation of the physical object in the digital domain, its state cannot be modified by an external entity, but can only be read. Therefore, the DT state can be retrieved using the “/stat” endpoint with the verb GET. Nevertheless, the DT can also accept *actions requests*, exposed by HTTP call to, at the “/action” endpoint, using the verb POST.

The reason for implementing 2 types of adapters is to obtain 2 different results: the first, is to guarantee *composition capabilities*: indeed, MQTT Digital Adapters are used to publish DTs' data and then replicate the same data ingestion pattern depicted describing the MQTT Physical Adapter. Therefore, to achieve composition, another DT will ingest data from multiple MQTT topics generated by machine-level DTs and respond to composition interests, fusing them and obtaining a *grouped* overview of the underlying DTs. The second result is instead guaranteed by the HTTP Adapter. It indeed follows the request/response communication pattern, giving the DT the possibility to be decoupled from external applications and associated requests. In this way, if an application (or even another DT) is interested in analysing the DT state or wants to inject an exposed action, it can be done through the DT HTTP Digital Adapters.

5.2.3 Digital Twins Cores: Augmentation Functions

In WLDT, general augmentation capabilities are reached by the execution of each DT shadowing Function. The Shadowing function contains indeed the

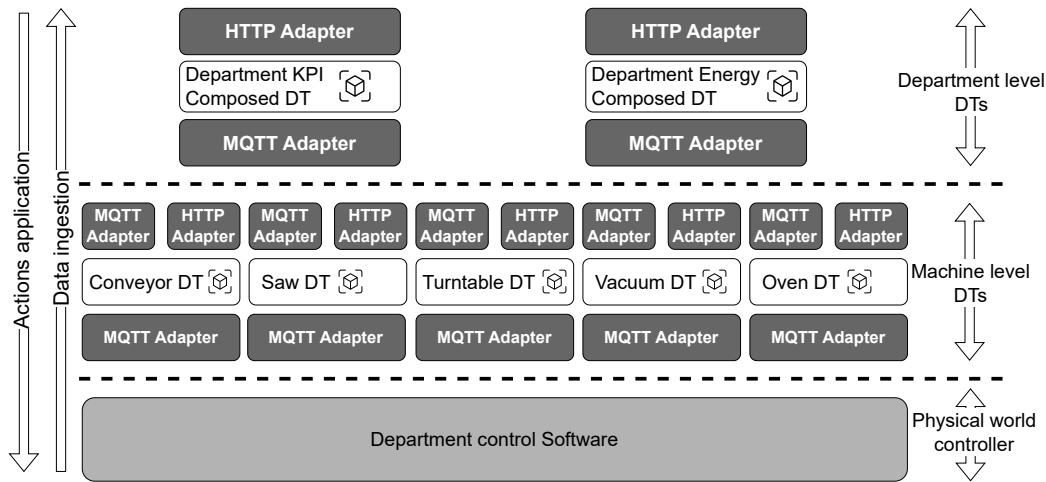


Figure 5.7: The proposed Figure depicts the implemented DT structure. For the sake of simplicity, the station processes one piece at a time.

$model(s)$ characterising each DT behaviour. Shadowing functions are therefore specialised across different machines, concerning the target behaviour that is necessary to achieve. All machines' DTs infer the machine production state concerning the events received for the 2 aforementioned variables, i.e. *product-on-carrier* and *process-completed*. The state-transition model slightly varies from machine to machine, for the logic the machine operations follow. Nevertheless, the general state description respects the following combinations:

- if *product-on-carrier* is *false* and *process-completed* and is *false*, the inferred state of the machine is *idling*, as is not holding any piece and idles waiting for a piece that is worked in previous stations to be delivered to itself;
- if *product-on-carrier* is *true* and *process-completed* and is *false*, the inferred state of the machine is *working*, as the machine is holding the piece to reach target operations activities on it;

- if *product-on-carrier* is *true* and *process-completed* and is *true*, the inferred state of the machine is *completed*, as the production process of the considered machine reached its operation target;
- finally, if *product-on-carrier* is *false* and *process-completed* and is *true*, the inferred state of the machine is *waiting*, as the machine waits for the piece in the process to be completed by the other stations.

At the end of the last operation made by the last machine in the flow (or, in other words, when all machines reach the *completed* state), all machines' states are reset to *idling* state, as another piece is expected to be delivered to the department soon. This state-transition model is then used by the augmentation capability to infer the time the machine works, and then the associated OEE level, as described in 3.2.1. Indeed, not considering breakdowns or quality issues, OEE calculation depends only on the machine working time with respect to the planned available working time. Therefore, the DT associated with each machine saves the timestamp of the state transition from *idling* to *working* and from *working* to *completed*, computes the time delta, and then accumulates the final results to extract the machine OEE.

Among other augmentation functions, for the oven DT real-time power usage data is received, allowing computation of its energy consumption in the associated Shadowing Function. Other machines also track energy consumption, but based on hypothetical data-sheet values and events representing the machine *working* state, also mapped in the DTs' Shadowing Function.

5.2.4 Digital Twins Enabled Actionability

As introduced in Subsection 5.2.2, implemented DTs can accept action requests. An action can be injected in multiple ways and, generally speaking, its life cycle is logically opposed to the data acquisition phase. Indeed, to request an action externally is necessary to trigger the Digital Adapter of the DT. Then, the adapter passes the request to the Core of the DT which eventually, through augmentation, outputs the associated request to the Physical Adapter, which in turn triggers the action directly to the physical asset.

Of course, no action can be requested from the DT. The set of available actions is described in the PAD (see the PAD introduction in Paragraph 5.2.2), and can comprise all the possible actions that the physical asset can execute, a subset of them, or a set of actions which Core augmentation function then maps on the set of possible actions that the physical asset can accept. In other words, the set of actions that a DT can expose does not necessarily need to coincide one-to-one with the set of actions that the physical asset can accept. Through augmentation, the DT can offer a set of possible actions at a higher level of abstraction which requests can, in turn, trigger the associated low-level physical asset action concerning the actual state of the physical asset at the request moment. For example, having a physical entity capable of moving between point A, on the right, point B, on the left, and point M, in the middle, can accept requests as “*move to the right*” or “*move to the left*”. By the way, a higher level action request can involve something like “*move towards A*”, “*move towards B*” or “*move towards C*”. The low-level associated action must take into consideration the actual physical entity position, stored in the DT state, to understand if to move right or left. Therefore, the set of actions possibly exposed by the DT through its Digital Adapter can be “*move towards A/B/M*”, while the set of

actions passed to the physical asset must be “move left” or “*move right*”. The entity that logically translates the high-level action request into the low-level one is the Core of the DT, in its Shadowing Function through augmentation.

In the implemented scenario, 3 up to 5 DTs of the “Machine Layer DT” expose actions. In particular, the actions exposed involve the possibility to speed up and down the machine production rate. Machines involved are the vacuum-carrier, the turntable-carrier and the final conveyor. Designed control is achieved in the following: each machine exposes an HTTP Digital Adapter, accepting action requests at the endpoint */action* used with the verb POST. Then, the DT Core passes the request to the MQTT Physical Adapter of the DT, which writes the action request in the MQTT of the machine reporting the speed of the machine itself. The possibility of accepting speed variations action requests is defined in the PAD of the DT of each machine, as described in Subsection 5.2.2.

Following the same pattern, action requests can be also done by composed DT (which details are described in the following Subsection), which, in turn, can expose also their actions. Composed DTs’ actions can then in turn leverage lower-level DT actions’, or implement other logic described in the composed DT Core. In the experimental scenario, the KPI composed DT introduced later implements a control for the lower-level machines, and the setting of several product pieces to produce in the whole department.

5.2.5 Digital Twins Hierarchy Navigability

In an industrial environment, different actors belong to different organisation pillars, which are most of the time focused on a subset of information of the same production system. As a consequence, involved actors want to receive and see only a subset of information, generated from the same shop-floor data

and, eventually, manipulated in different ways to respond to their specific needs. The same pattern repeats at each level of the organisation chart, with information that gradually moves from a low to a high level. Is therefore straightforward that KPIs of interest to a production manager are different from those of an environmental manager, for example.

Two composed DTs were created for an actual demonstration of considered principles: a department-performance composition view and a department-environmental composition view. Composed DTs have an MQTT Physical Adapter subscribed to Machine Layer DT topics, and therefore receiving data from Machine Layer DTs Digital Adapters'. Physical Adapters of a composed DT carry information about the structure of the received data from each underlying DT in its PAD, as machine-level DTs described above in 5.2.2. In this scenario, for example, the MQTT PAD contains the topic of interest where machine layer DTs publish their information for each composed DT, and associate to each of them a key. Then, at the start composed DTs subscribe to topics present in the PAD and start to listen to information published on the considered topic. In this way, data can be received from multiple sources, i.e. multiple DTs of a lower level, and then listened information is passed to the composed DTs Shadowing Function.

The received data is processed through each composed DT Shadowing Function, where each DT can eventually exploit its augmentation capabilities to then form its state representation, and expose it externally through one or more Digital Adapters forming the Digital Interface.

In particular, the two compositions have implemented a Digital Interface composed of one HTTP Adapter for each DT. Each HTTP Adapter follows the characteristics depicted in 5.2.2 referring to its connected DT.

Shadowing Functions are specialised concerning the composed DT nature

and scope. In the department-performance composed DT Shadowing Function, composition and augmentation are used to compute the department-weighted OEE, while in the department-environmental composition instead, the focus is the energy consumption of the entire department. In this case, differently from the single machines DTs', property updates trigger these calculations, allowing different perspectives of the same production system. Composed OEE is calculated in very different ways [44]. Nevertheless, considering a group of machines going from $i = 1, \dots, n$, in the proposed experiments Weighted OEE is calculated as:

$$\sum_{i=1}^n OEE_i \times \frac{\text{net available time}_i}{\text{total net available time}} \quad (5.1)$$

For energy, instead, the total energy of the department is calculated as the sum of the energy consumption of each machine, updated at each machine energy update.

The composed KPI DT includes also throughput computation, a metric representing the production rate capability of the system. The computation of this metric considers timestamps of moments when a single piece starts and completes the production crossing. Therefore, its implementation involved the analysis of each received property by the performance-composed DT, to understand if the information involves the first or the last machine in the department, needing, therefore, the analysis of relationships existing by each machine DT. In this case, the throughput calculation algorithm is triggered or stopped, returning the actual throughput level at the i -th piece crossing the system. To perform correctly the actual analysis, at each property update, if the received property ID meets the needed criteria, the DT sending the actual update is queried through an HTTP call recalling the address from the composed DT relationships field. Then, the received composing DT state

is analysed and, if its relationships highlight that it is the first or the last machine in the department, then the received property is used to update the throughput state. In the considered use case, navigability stands in the fact that the KPI DT through its relationship with composition components, *navigates* the structure depicted in Figure 5.7, retrieving the needed information from components themselves. Then, relationships of retrieved components are analysed too, to get if the received update comes from the first or the last machine in the department.

The resulting composition structure, (see Figure 5.7), ensures an organic perspective for stakeholders interested in the performance of the production system and overall energy consumption.

5.2.6 Architecture Performance & Complexity

The DT architecture detailed here provides distinct advantages in reducing the physical and *digital complexity* of the industrial use case scenario. To estimate the complexity levels with and without their adoption, and therefore propose a new point of view when researching the application of DTs in the industrial domain, the indicator introduced in [72], described in Subsection 5.1.4, and called Digital Complexity Indicator (DCI) has been used. This metric measures the perceived complexity associated with any application requiring the interaction with the physical layer taking into account the following criteria: i) *Required Protocols (p)*: the number of application layer protocols required by a digital application or service to interact with the deployed physical assets to collect data and to send commands; ii) *Communication Patterns (c)*: the number of communication patterns needed to interact with involved devices and platforms (e.g., Publisher/Subscriber or Request/Response); iii) *Data Formats*: the number of different data for-

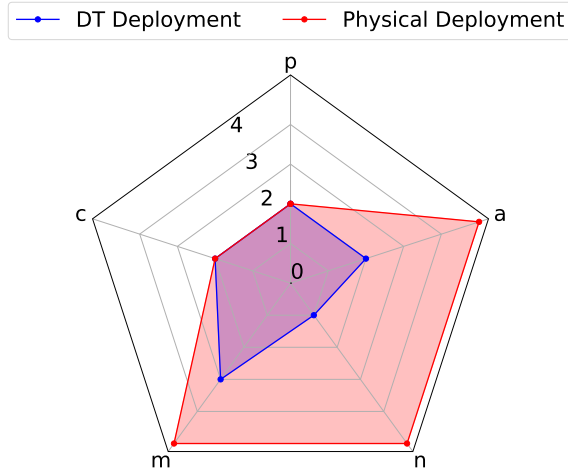


Figure 5.8: DCI metric comparing system complexity with and without a DT deployment.

mats, serialisation, and information representation techniques required to read and send data from and to deployed devices; iv) *Interaction Points (n)*: the number of different modules, services or platforms that an application should interact with to retrieve all the target data or consume services; and v) *Aggregation Points (a)*: the number of aggregation or composition levels required to abstract the physical world into the right level of complexity with respect to the observers' typologies and their application goals (e.g., merging information and telemetry data from machines in the same production line).

Moreover, each criterion is designated a specific Importance Factor (IF) on a scale from 1 (lower) to 3 (higher). This factor serves as a multiplicative score for the DCI (Digital Complexity Index) of the criterion, reflecting its influence on the development, deployment, and maintenance of the solution. The DCI is directly linked to the exposed digital complexity that an external application must manage to communicate with deployed physical assets, gather data, and issue commands. The formula $DCI = \sum_{i=1}^5 criterion_i \times IF_i$

Criterion	Imp. Factor	Use Case	
		without DT	with DT
Required Protocols (p)	2	2	2
Communication Patterns (c)	1	2	2
Data Formats (m)	3	5	3
Interaction Points (n)	2	5	1
Aggregation Points (a)	3	5	2
DCI	–	46	23

Table 5.5: Estimation of the Digital Complexity Index for proposed use cases

signifies the level of digital complexity that the external application encounters concerning the physical use case.

The result is reported in Figure 5.8 and in Table 5.5 the indicator result highlights how using DTs standardises the physical-worlds data formats, exposing to the digital domain a single interaction and aggregation point. Without DTs, DCI grows with the number of involved devices and protocols while on the other hand, DTs decouple the digital complexity from their digital counterpart, simplifying the interaction with external applications.

Nevertheless, the reduced complexity comes with a cost in terms of machine computation, because of the introduced layer of abstraction. To understand the impact of the added solution, CPU, memory and Shadowing Function average execution times have been computed in an experiment producing 20 pieces for a total run time of 17 minutes. Results reported respectively in Figure 5.11a, 5.11b, and Figure 5.10: execution times never exceeds 5 milliseconds on average, with a CPU usage of 2% for the maximum peak. Memory usage stays always at under 200MB, with the garbage collector activity evident in optimising it. Is worth pointing out that Shadowing

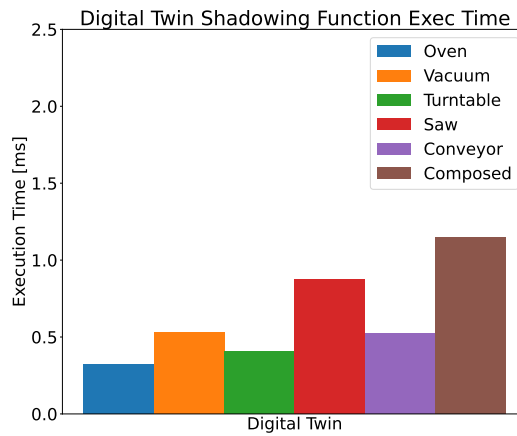


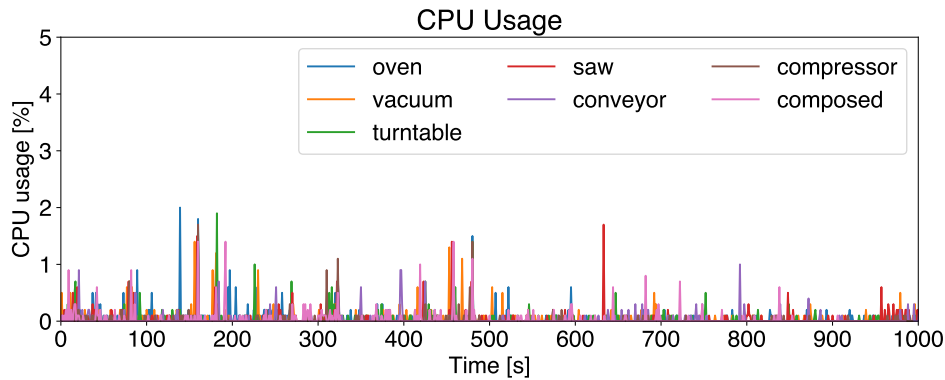
Figure 5.9

Figure 5.10: Shadowing functions execution times.

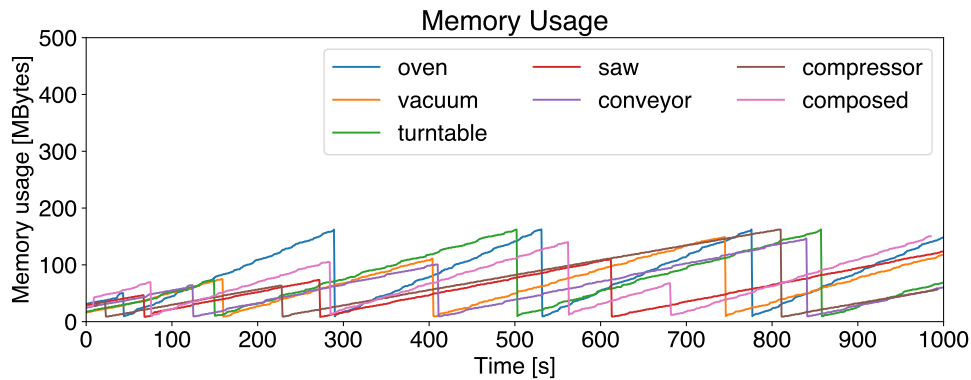
Functions have been implemented without taking into account specific memory optimisation techniques. Average Shadowing Function execution rates range from 0.120681 to 1.348063 events/second.

5.3 External AI & Industrial Behaviour

In a real-world industrial scenario, it is reasonable to think that the DT architecture of the shop-floor will interact or be used by external applications as “handlebars” to control the underlying shop-floor Physical Assets. The shop-floor DT architecture offers a point of view of the real-world state of affairs, eventually specialised on the external application needs, updated in a close-to-real-time fashion, and homogeneous access to physical assets for action requests coming from outer entities. This is the case of an external app controlling the production of the system. In the prototype, has been implemented software that monitors the external markets requests or customer orders, and dynamically adjusts the production rate accordingly. The



(a)



(b)

Figure 5.11: Resource consumption for each involved DT in terms of CPU percentage and Memory.

external software involved in the scenario can be considered as an AI model, mapping the actual market needs to the production capacity of the system. The output of the external software is then passed, through an action request, to the underlying CPS, consequently modulating the system production rate. The external system does not take any additional information from the CPS for the model to work, falling then in the described pattern of interaction called Advisor AI, where an AI external to the DT architecture takes advantage of the *actionability* offered by the DT architecture to control it with

respect to information received from an external context.

The production rate adjustment has been obtained by passing a configuration to the DT representing the target machine whose production speed has to be updated. In particular, the production speed update request is made through the means of the machine DT HTTP Digital Adapter, using a POST request. The request is then passed to the machine DT Shadowing function, which checks for the integrity of the request, i.e., if the command is valid and suitable for the machine the DT represents in the digital domain. If the control is positive, the request is passed to the machine DT Physical Adapter, which publishes the configuration in the associated MQTT topic. When a new product is processed, the department production control checks for configuration published on the MQTT broker, in the machine configuration topic, controls its feasibility, and, if the configuration is suitable, produces the piece respecting the passed configuration. Three are the machines enabled to vary their speed: the vacuum gripper carrier, the turntable carrier and the conveyor carrier. Additionally, the external AI-based software outputs also the number of pieces to be produced by the whole department, passing the request to the department-composed DT through its HTTP Digital Adapter. Consequently, the department DT monitors the overall amount of pieces produced by the system and, when the production amount is met, stops the production activity of the whole department.

To verify that the application built respects the expected behaviour of its requirements, a test has been made with the following characteristics: each machine has 3 possible speed configurations, i.e. “low”, “medium” and “high”; at the beginning of the test all speeds are set to “low”; the first and the second pieces are produced maintaining the speed configuration for all machines to “low”, to pass the oven heat up transitory phase; then, from the third onwards

the external AI application outputs a higher needed production speed to meet the growing market requests, translating than the output to a target machine. Since the production requests grow constantly, the production speed grows constantly too, having each machine piece after piece sped up by one step, i.e. from “low” to “medium” or from “medium” to “high”. Contextually to the market requests production speed, the amount of pieces to produce are monitored too for the single experiment, and the corresponding information is passed to the department Composed DT through an action to stop the whole production when the needed amount of products are completed.

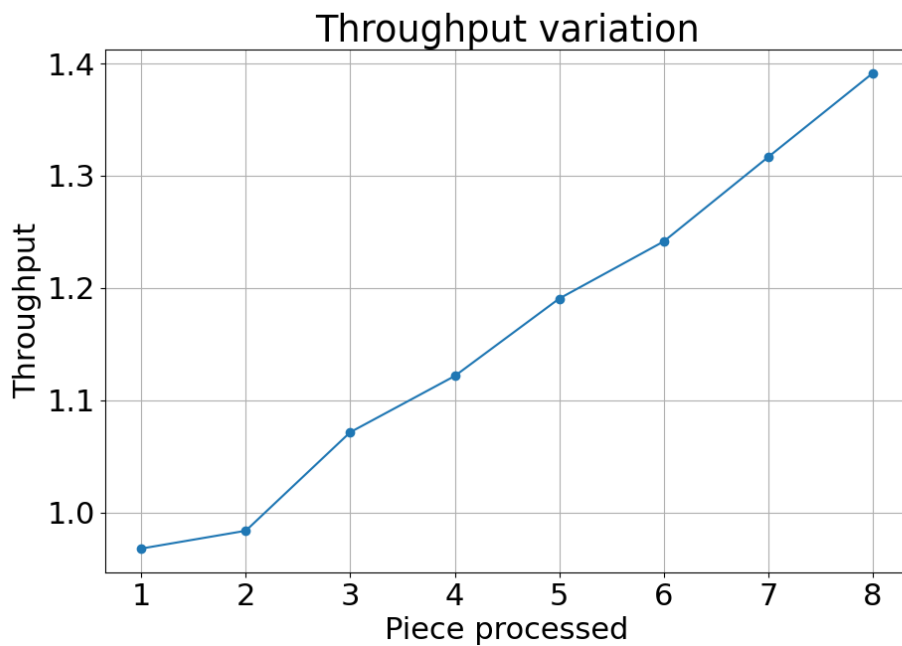


Figure 5.12: Throughput variation in speed adjustment experiment.

The test result has been monitored using the DTs ability to monitor the real-world state of affairs in a close to real-time fashion. Indeed, the throughput monitoring feature implemented in the composed Department DT has been used to track the whole department throughput which was expected

to change during the eight runs because of the speed updates during the production test. The throughput changes are reported in Figure 5.12. The result highlights how the production system sped up during the experiment, with the CPS being controlled by an Advisor AI application, external from the DT architecture, and fed with data coming from external sources. The difference in speed between the production of the first and second pieces is justified by the overcoming of the oven heating transition phase. From the third phase, instead, the production sped up because of the different machine speed configurations set by the external application through machine DTs. The system throughput passed from a little less than $1\text{ pc}/\text{min}$ to $1.4\text{ pc}/\text{min}$ in production capacity.

5.4 Embedded DT Intelligent Augmentation

As described so far, DTs are software entities aimed at representing a physical counterpart in the digital domain as a close to real-time high-fidelity replica. They map physical capabilities, ingest associated data forming DTs' internal state, eventually expose action capabilities and propagate events to software components linked to the DT (i.e. other software or other DTs). DT augmentation capabilities can bring to higher information level extracted from data ingested by the DT and exposure to external entities as its actual state. This is the case for the third experiment proposed in this dissertation.

The experiment setting involves the *Fischertechnik Multiprocess Station with Oven* hardware described in Subsection 5.2.1. A simplified version of the industrial digitised architecture has been implemented: the structure is reported in Figure 5.13. Most of the architecture is similar to what is reported in Figure 5.7, with 2 main differences: first, the Conveyor DT now has an

additional adapter, aimed at accepting data from an additional source and propagating them towards the Conveyor DT core; second the Department Energy Composed DT is not used, as the following activities are focused only on machine-layer DTs and general composition of the associated department.

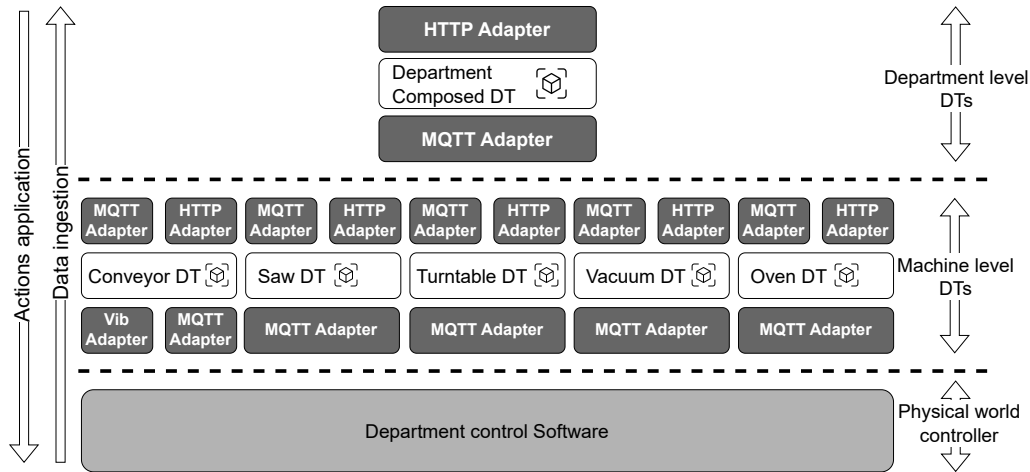


Figure 5.13: Implemented architecture for the Embedded DT experiment. In this scenario, the Composed Energy Department DT and Composed KPI Department DT have been replaced by a general Department Composed DT.

To demonstrate potential augmentation capabilities brought by the Embedded AI proposed pattern, a system classifying input vibration data to obtain system health state has been implemented. In particular, the Fischertechnik does not include specific sensors for measuring its elements' vibrations. Moreover, it is not present as a physical device that, if actuated, enables a physical breakdown of the Modelfabrik. Therefore, to obtain such data as “coming from the physical entity”, Physical Adapters capabilities of the DT have been used to our advantage.

A specific Physical Adapter has been implemented for the depicted use case. Data is taken from an existing dataset and then passed to the relative

DT-core once a second. In this way, the Physical Adapter simulates a reading coming from the sensor each second, passing the received information to the DT-core according to the data-reception rate.

The selected dataset is the *NASA Bearing Dataset*.⁶ It is the result of an experimental setup involving 4 bearings put under stress and monitored by one accelerometer applied to each bearing. The dataset is therefore composed of 4 columns plus the timestamp column. Each reading has been made 10 minutes apart. The data collection took about 6 days for a total of 982 data points collected.

The associated AI model used for the experiment specialises in the anomaly detection of the selected dataset.⁷ The AI system is based on an auto-encoder neural network model created using Long Short-Term Memory (LSTM). In particular, the neural network is trained on the portion of data not affected by any kind of breakdown. Indeed, is very easy to spot, more or less, the breakdown point of each bearing by looking at the vibration graph along the whole experiment (see Figure 5.14).

The training part involves the prediction of the next vibration points for each bearing. Then, the trained model is tested against the training data extracting the loss value - i.e., the absolute difference between the predicted and the actual value - and using the resulting graph to select a suitable threshold for the loss value where to trigger a broken classification of the bearing 5.15.

The selected loss value is 0.275, then tested on the whole experiment timeline. The resulting graph is reported in Figure 5.16, showing that the

⁶Data set available at: <https://www.kaggle.com/datasets/vinayak123tyagi/bearing-dataset>

⁷Model training available at <https://www.kaggle.com/code/rkuo2000/sensor-anomaly-detection>

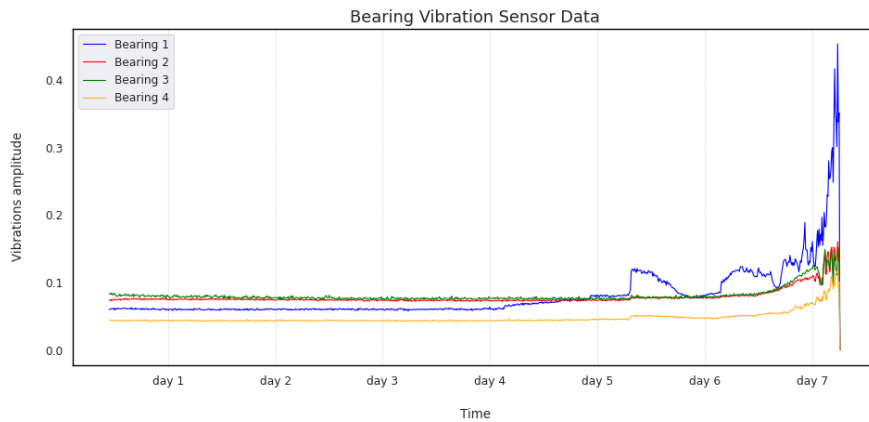


Figure 5.14: Graph of the bearings vibration data collection of the NASA-bearing dataset. It is easy to see that after day 4 a drifting in bearings vibrations emerges.

selected value is suitable for breakdown classification purposes.

The model defined so far has then been encapsulated into a Python file with all the associated procedures aimed to perform a correct classification from the vibrations passed data.

The vibrations physical adapter is then used to feed the conveyor DT. To reflect the actual information received by the conveyor DT, the physical conveyor has then been set to be always running, even if its operation is not explicitly required for moving a product towards the conveyor exit. The conveyor DT shadowing function then uses the received information in two different ways: first, to describe the DT state and expose it through conveyor DT physical adapters; second, by passing vibration data to a thread started internally the DT Shadowing Function and running parallel to the DT and calling the Python function aimed in performing the health classification from bearing data.

The mentioned thread is started if and only if a buffer of enough data

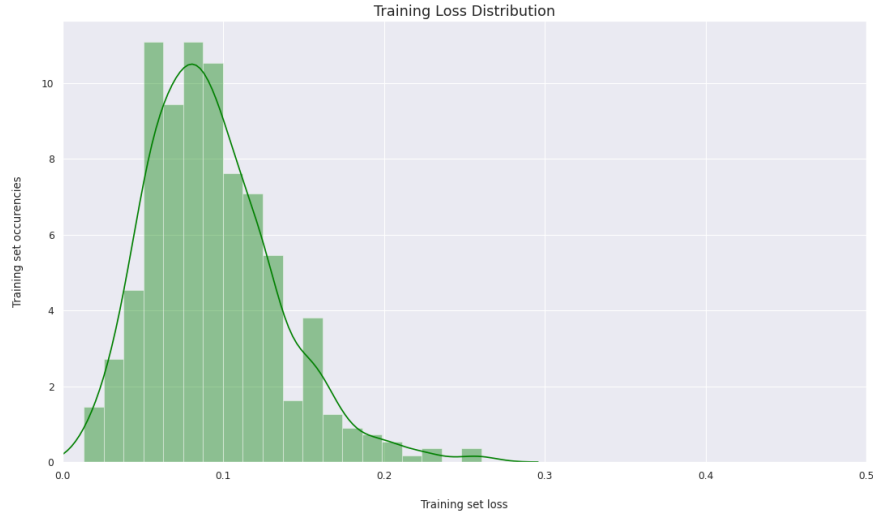


Figure 5.15: Loss distribution of the Nasa-bearings dataset. Is easy to spot as a good cut point for the classification a loss value of 0.275.

is obtained (the model needs 4 vibration data points to perform the classification) and if another thread is not already running. The received classification is then added to the DT state, contributing to augmenting its state description. It is worth pointing out that the whole AI-based classification mechanism is not performed on the hardware physically controlling the Fischertechnik activity. Indeed, the conveyor DT does not reside on the RevPi. The only purpose of the RevPi is to control the Fischertechnik operations, to send telemetry and events data to associated DT and eventually accept DT actions in the form of system configurations. The classification is performed by the conveyor DT, which resides on an external device equipped with the DT architecture, exchanging through an MQTT broker data with the software controlling the Fischertechnik residing on the RevPi.

A “safe shutdown procedure” based on the conveyor DT with Embedded AI, has been also developed. Its behaviour through time is depicted in

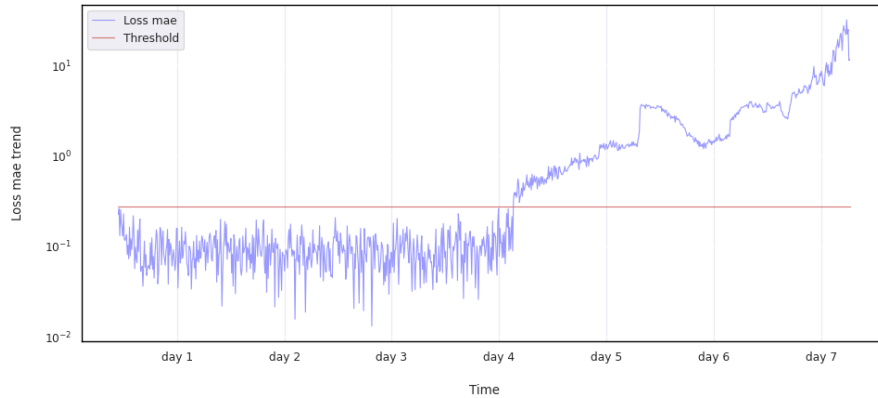


Figure 5.16: Loss distribution across the whole of the Nasa-bearings dataset. The selected threshold works well in classifying when the bearings are healthy or not.

Figure 5.17. During normal conveyor operations, its health status check is continuously performed leveraging the associated DT. After some operations, bearings vibrations drift, crossing the classification threshold and letting the associated conveyor health state change from true (i.e., healthy) to false (i.e., unhealthy - point 1 in the Figure). The DT health information propagates to the composed DT, which in turn, activates the “safe shutdown procedure”. The procedure, as designed for the experiment, consists of a slowdown request made to the conveyor through a DT action by the composed department DT, and then, when the piece in the system reaches the conveyor exit light barrier, a whole department shutdown action request is propagated to all DTs and then to the associated physical assets by the composed DT. Following this procedure, the composed DT sends a new speed level through an action to the conveyor DT (point number 2 in Figure 5.17). As a consequence, the conveyor DT receives the new speed (point 3 in the Figure), checks the action feasibility and propagates it to the underlying DT (point 4 in the Figure).

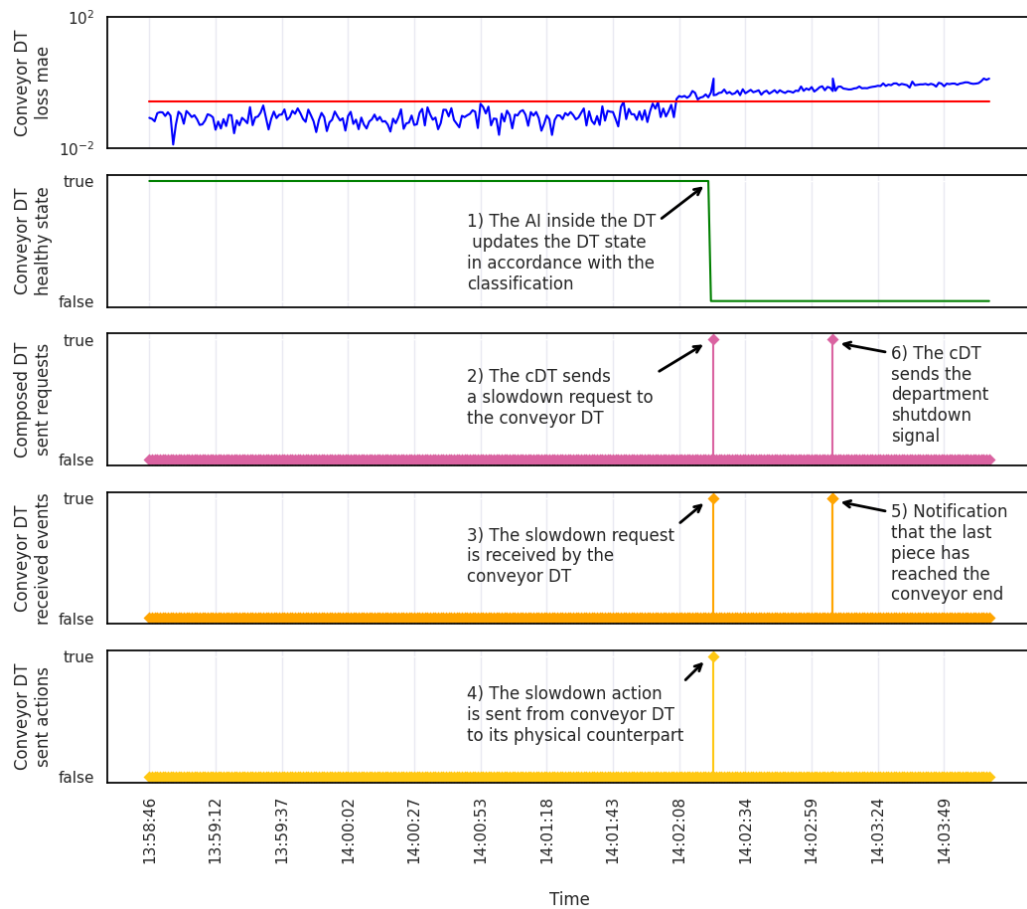


Figure 5.17: Graph depicting changed states and relative events occurred captured by DTs in the Embedded DT experiment.

After that, the system waits for the last piece in the system to reach the exit of the conveyor, crossing its light barrier. When the last product reaches the barrier, an associated event reaches the conveyor DT (point 5 in the Figure), which is then equally propagated to the composed DT. When the event is received by the composed DT, it triggers the whole system shutdown (point 6 in Figure).

The described experiment highlights how Embedded AI can enable advanced opportunities in bringing AI to the edge also for not native AI hard-

ware. Moreover, their combination brings intelligence to various aspects of DTs internal behaviour, making it available to various Industrial IoT scenarios.

Chapter 6

Conclusions & Future Research

Modern industrial shop floors are facing a new wave of challenges, pushing them to face problems opposite to the massive standardisation approach that had characterised such systems. Growing complexity is demanding new approaches to managing industrial information and operational states. A high level of expectations is posed in AI technologies: indeed, researchers as well as practitioners already started using the technology to solve different sets of problems. Nevertheless, most of the time, problems have been faced considering only its niches, with promising results due to the abundance of data that production systems generate day by day, but without considering generalised approaches.

Therefore, a way for transposing shop-floor structures and events in the digital domain with a standardised approach, following the main evolution principle of industry, without losing a customised physical representation possibility, was necessary. As many researchers pointed out, DTs are the candidate technology for bringing to life the depicted expectations.

The current dissertation considered DTs as a way to digitise physical assets with a standardised approach. After having reported different appli-

cations of both AI and DTs in the industry, DTs characteristics have been reported and described. DTs are characterised by Physical Adapters (i.e. input adapters) and Digital Adapters (i.e. output adapters), aimed at managing any kind of communication protocol. Moreover, in this dissertation DTs are considered as *active* entities, with the ability of using one or more internal *models*. DT internal models characterise actual DT behaviour and are designed concerning the context where the DT operates and its actual purpose. The model can manipulate and eventually augment DT representation, as its state, events, and relationships. Moreover, models can affect also action requests, if and how they are communicated to physical counterparts to affect the physical world state of affairs.

After having characterised DTs characteristics, shop-floor modelling has been described and useful features for an effective representation are reported, such as data ingestion and augmentation capabilities, physical world actionability, physical relationships representation, composition and customisable hierarchical views, and interaction with other applications.

The dissertation then moved towards the investigation of possible interaction patterns between DTs and AI techniques. 4 total patterns have been identified:

- *Observer AI*: AI external to the DTs ecosystem, gaining DTs data and offering elaborated insights to external applications;
- *Advisor AI*: AI external to the DTs ecosystem, gaining data from external applications, extracting valuable information, and then making action requests in accordance to gain insights into the DTs architecture for reaching a desired state-of-affairs at the physical level;
- *Controller AI*: a mix of the first two, i.e. an AI system external to

the DT architecture accepting data both from the system as well as from external sources, and then affecting the real-world state-of-affairs through actions requests made to associated DTs or an eventual target composition following insights gained from the actual AI model;

- *Embedded AI*: AI internal to a single DT, being part of the model of the DT, and augmenting capabilities of the DT.

Actual implementations of the proposed patterns have then been investigated, in a simulated and realistic physical environment. Together with the concrete implementation exploration, the impact of implemented DTs architectures over the complexity of considered systems has been computed and proposed, suggesting complexity management as an aspect to take into account when applying DTs in a real-world production environment. Indeed, it fostered the idea that DTs should not only represent the physical state of affairs in the digital domain but have to do it by tackling physical complexity from its foundations. It is argued that DTs have great potential in lower from the very initial application industrial *digital complexity*, and support the management of also *static* and *dynamic industrial complexity*, supporting therefore the more general concept of *Cyber-Physical Complexity*.

Application opportunities of DT technologies mixed with mainstream IT tools as AI is only at its beginning, both at the research and application level. Most applications principles need to be explored in every sector, and the industrial one is no exception. DTs, as an enabling technology, open up a series of new opportunities in different industrial areas, and they need to be studied. In particular, DT usage brings into the digital domain a trusty representation of the physical structure that can be used for different purposes. The most straightforward yet critical regards scheduling activities.

Despite being activity scheduling very deeply studied, its application in real-world scenarios still needs improvements, due to the lack of bond between the scheduling result and its actual implementation monitoring. Indeed, actual technologies already provide enough information to set an effective scheduling of production, but monitoring the evolution of its implementation is still a problem. The “circle-of-information” from the digital to the physical (the actual scheduling setup), and backwards from the physical to the digital (the scheduling physical implementation monitoring) have always been “open”, with a lack in the monitoring of the set scheduling. DTs can now bring the foundation for closing the circle, opening up new opportunities in the scheduling activities, making them *dynamic* concerning the real happenings of the physical domain. Moreover, new scheduling strategies and heuristics can be set, concerning the actual industrial structure transposed into the digital domain.

A second interesting area of research considers exactly industrial architectures. Indeed, as considered until now, they are “design” principles that gave the ability to domain experts to set up the main backbone of the production system, concerning market requests and production competitive priorities. Nevertheless, the actual monitoring of the industrial shop-floor evolution over time is neglected, and gathering key information about architecture indicators at the production runtime has always been a hard task with approximate results. Having the transposed representation of the shop-floor structure and existing relationships can instead break data collection barriers, bringing new tools into the industrial research area. Architecture indicators evolution over time can be available to practitioners, opening up a new set of opportunities when a shop-floor update is necessary to face new business opportunities. Moreover, the same indicators (as well as potentially new ones) can allow

researchers to study industrial architectures under a new lens.

Among other future research, there are also effective application principles of DTs in the industrial domain. Indeed, the representation of the physical shop floor in the digital domain should be effective and useful for all the different needs of each industrial pillar. For example: is it better to have only one big DT collecting all the relevant information for each physical asset, or break down its representation into multiple DTs, specialised on one or more pillars needs? Moreover: how the DT structure can face physical updates over time? How do updates impact the chosen DT architecture? Or, for example: given a defined digitisation strategy, should associated DTs live on the edge, in the cloud, in between or whatever? This and more questions about how to effectively build a DT architecture for a given industrial structure need to be studied.

The last (but not least) research path is the possible DT specialisations that arise with respect to the considered application domain. The considered research path is somehow correlated to the previous, but not limited to the implementation strategy concerning the involved DT target. Indeed, DTs can digitise all the physical assets of a shop floor but not only them. For example: thinking about industrial quality measuring tools, is possible to consider based on them both “machine DTs” as well as “product DTs”. Machine sensors can indeed represent and collect at the same time information about their internal state and operation state, as well as information about the crafted product that can, as a consequence, feed associated DTs. Considering the same scenario is also worth asking ourselves if DTs are the right tool to represent not only physical assets but also what is going on at the physical level under the lens of *processes*, moving towards the digitisation of *physical concepts*. In other words, representing a production process or a sub-step of

it as a DT can be worthwhile for multiple reasons, such as monitoring the actual completion state of a production request, as well as acting as a bridge between the information set during the design phase of a product - as product design, cycles, Bill Of Material and so on - and the actual implementation of the product designed characteristics.

These are only some of the possible future research activities involving DTs in the industrial domain. Other application potentials can be found by the research community as well as by practitioners. Despite DTs as a concept that has been around for a while, a fully integrated application of them in the domain is still at its beginning, with lots of opportunities that are still to be considered.

Bibliography

- [1] Sailesh Abburu, Arne J. Berre, Michael Jacoby, Dumitru Roman, Ljiljana Stojanovic, and Nenad Stojanovic. Cognitwin – hybrid and cognitive digital twins for the process industry. In *IEEE Int. Conf. on Engineering, Technology and Innovation (ICE/ITMC)*, pages 1–8, 2020.
- [2] Adnan Aktepe, Emre Yanık, and Süleyman Ersöz. Demand forecasting application with regression and artificial intelligence methods in a construction machinery company. *Journal of Intelligent Manufacturing* 2021 32:6, 32:1587–1604, 2 2021.
- [3] Abdullah Alkhoraif, Hamad Rashid, and Patrick McLaughlin. Lean implementation in small and medium enterprises: Literature review. *Operations Research Perspectives*, 6:100089, 1 2019.
- [4] Nima Amani and Ehsan Kiaee. Developing a two-criteria framework to rank thermal insulation materials in nearly zero energy buildings using multi-objective optimization approach. *Journal of Cleaner Production*, 276:122592, 2020.
- [5] I Antonioli, P Guariente, T Pereira, L. Pinto Ferreira, and F.J.G. Silva. Standardization and optimization of an automotive components production line. *Procedia Manufacturing*, 13:1120–1127, 2017. Manu-

facturing Engineering Society International Conference 2017, MESIC 2017, 28-30 June 2017, Vigo (Pontevedra), Spain.

- [6] Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador García, Sergio Gil-López, Daniel Molina, Richard Benjamins, et al. Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. *Information fusion*, 58:82–115, 2020.
- [7] Juuso Autiosalo, Joshua Siegel, and Kari Tammi. Twinbase: Open-source server software for the digital twin web. *IEEE Access*, 9:140779–140798, 2021.
- [8] Ali Vatankhah Barenji, Xinlai Liu, Hanyang Guo, and Zhi Li. A digital twin-driven approach towards smart manufacturing: reduced energy consumption for a robotic cell. *International Journal of Computer Integrated Manufacturing*, 34:844–859, 2021.
- [9] Paolo Bellavista, Nicola Bicocchi, Mattia Fogli, Carlo Giannelli, Marco Mamei, and Marco Picone. Requirements and design patterns for adaptive, autonomous, and context-aware digital twins in industry 4.0 digital factories. *Computers in Industry*, 149:103918, 8 2023.
- [10] Paolo Bellavista, Carlo Giannelli, Marco Mamei, Matteo Mendula, and Marco Picone. Application-driven network-aware digital twin management in industrial edge environments. *IEEE Transactions on Industrial Informatics*, 17(11):7791–7801, 2021.
- [11] Hind Benbya, Ning Nan, Hüseyin Tanriverdi, and Youngjin Yoo. Complexity and information systems research in the emerging digital world. *Mis Quarterly*, 44(1):1–17, 2020.

- [12] Tim Bolender, Gereon Burvenich, Manuela Dalibor, Bernhard Rumpe, and Andreas Wortmann. Self-adaptive manufacturing with digital twins. *Proceedings - 2021 International Symposium on Software Engineering for Adaptive and Self-Managing Systems, SEAMS 2021*, pages 156–166, 5 2021.
- [13] Darya Botkina, Mikael Hedlind, Bengt Olsson, Jannik Henser, and Thomas Lundholm. Digital Twin of a Cutting Tool. *Procedia CIRP*, 72:215–218, 2018.
- [14] Andrew P Bradley. The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern recognition*, 30(7):1145–1159, 1997.
- [15] Sara Braganca and Eric Costa. An application of the lean production tool standard work. *Jurnal Teknologi (Sciences and Engineering)*, 76(1):47–53, 2015.
- [16] Thyago P. Carvalho, Fabrizzio A.A.M.N. Soares, Roberto Vita, Roberto da P. Francisco, João P. Basto, and Symone G.S. Alcalá. A systematic literature review of machine learning methods applied to predictive maintenance. *Computers & Industrial Engineering*, 137:106024, 11 2019.
- [17] Chao Chen, Zhicheng Ji, and Yan Wang. Nsga-ii applied to dynamic flexible job shop scheduling problems with machine breakdown. *Modern Physics Letters B*, 32, 12 2018.
- [18] Jian Chen, Tong Ning, Gangyan Xu, and Yang Liu. A memetic algorithm for energy-efficient scheduling of integrated production and

- shipping. *International Journal of Computer Integrated Manufacturing*, 2022.
- [19] Xiao Chen, Martin A. Eder, Asm Shihavuddin, and Dan Zheng. A human-cyber-physical system toward intelligent wind turbine operation and maintenance. *Sustainability*, 13(2), 2021.
- [20] Ximing Chen, Eunsuk Kang, Shinichi Shiraishi, Victor M. Preciado, and Zhihao Jiang. Digital behavioral twins for safe connected cars. In *Proceedings of the 21th ACM/IEEE Int. Conf. on Model Driven Engineering Languages and Systems*, page 144–153. Association for Computing Machinery, 2018.
- [21] Jiangfeng Cheng, He Zhang, Fei Tao, and Chia-Feng Juang. Dt-ii:digital twin enhanced industrial internet reference framework towards smart manufacturing. *Robotics and Computer-Integrated Manufacturing*, 62:101881, 2020.
- [22] Jiangfeng Cheng, He Zhang, Fei Tao, and Chia-Feng Juang. Dt-ii:digital twin enhanced industrial internet reference framework towards smart manufacturing. *Robotics and Computer-Integrated Manufacturing*, 62:101881, 2020.
- [23] Yu-Liang Chou, Catarina Moreira, Peter Bruza, Chun Ouyang, and Joaquim Jorge. Counterfactuals and causability in explainable artificial intelligence: Theory, algorithms, and applications. *Information Fusion*, 81:59–83, 2022.
- [24] Yu-Liang Chou, Catarina Moreira, Peter Bruza, Chun Ouyang, and Joaquim Jorge. Counterfactuals and causability in explainable artificial

intelligence: Theory, algorithms, and applications. *Information Fusion*, 81:59–83, 2022.

- [25] Chiara Cimino, Elisa Negri, and Luca Fumagalli. Review of digital twin applications in manufacturing. *Computers in Industry*, 113:103130, 2019.
- [26] Simone Cirani, Luca Davoli, Gianluigi Ferrari, Rémy Léone, Paolo Medagliani, Marco Picone, and Luca Veltri. A scalable and self-configuring architecture for service discovery in the internet of things. *IEEE Internet of Things Journal*, 1(5):508–521, 2014.
- [27] N. Crespi, A.T. Drobot, and R. Minerva. *The Digital Twin*. Springer International Publishing, 2023.
- [28] Manuela Dalibor, Nico Jansen, Bernhard Rumpe, David Schmalzing, Louis Wachtmeister, Manuel Wimmer, and Andreas Wortmann. A cross-domain systematic mapping study on software engineering for digital twins. *Journal of Systems and Software*, 193:111361, 2022.
- [29] Jovani Dalzochio, Rafael Kunst, Edison Pignaton, Alecio Binotto, Sri-jnan Sanyal, Jose Favilla, and Jorge Barbosa. Machine learning and reasoning for predictive maintenance in industry 4.0: Current status and challenges. *Computers in Industry*, 123:103298, 12 2020.
- [30] Hossein Darvishi, Domenico Ciuonzo, Eivind Rosón Eide, and Pier-luigi Salvo Rossi. Sensor-fault detection, isolation and accommodation for digital twins via modular data-driven architecture. *IEEE Sensors Journal*, 21(4):4827–4838, Feb 2021.

- [31] Rebecca Duray. Mass customization origins: Mass or custom manufacturing? *International Journal of Operations & Production Management*, 22(3):314–328, Mar 2002.
- [32] Pavlos Eirinakis, Kostas Kalaboukas, Stavros Lounis, Ioannis Mourtos, Jože M. Rožanec, Nenad Stojanovic, and Georgios Zois. Enhancing cognition for digital twins. In *2020 IEEE International Conference on Engineering, Technology and Innovation (ICE/ITMC)*, pages 1–7, 2020.
- [33] Waguih Elmaraghy, Hoda Elmaraghy, Tetsuo Tomiyama, and Laszlo Monostori. Complexity in engineering design and manufacturing. *CIRP Annals*, 61:793–814, 1 2012.
- [34] Angelo Encapera, Abhijit Gosavi, and Susan L. Murray. Total productive maintenance of make-to-stock production-inventory systems via artificial-intelligence-based ismart. *International Journal of Systems Science: Operations and Logistics*, 8:154–166, 2021.
- [35] Angelo Encapera, Abhijit Gosavi, and Susan L. Murray. Total productive maintenance of make-to-stock production-inventory systems via artificial-intelligence-based ismart. *International Journal of Systems Science: Operations and Logistics*, 8:154–166, 2021.
- [36] Romina Eramo, Francis Bordeleau, Benoit Combemale, Mark van den Brand, Manuel Wimmer, and Andreas Wortmann. Conceptualizing digital twins. *IEEE Software*, 39:39–46, 2022.
- [37] Weiguang Fang, Yu Guo, Wenhe Liao, Karthik Ramani, and Shaohua Huang. Big data driven jobs remaining time prediction in discrete

- manufacturing system: a deep learning-based approach. *International Journal of Production Research*, 58:2751–2766, 5 2020.
- [38] Ford corporate.
- [39] Jonas Friederich, Deena P. Francis, Sanja Lazarova-Molnar, and Nader Mohamed. A framework for data-driven digital twins for smart manufacturing. *Computers in Industry*, 136, 4 2022.
- [40] David J Friedman and George Foo. Variability and capability: The foundation of competitive operations performance. *AT&T Technical Journal*, 71:2–9, 1992.
- [41] Yaping Fu, Hongfeng Wang, Guangdong Tian, Zhiwu Li, and Hesuan Hu. Two-agent stochastic flow shop deteriorating scheduling via a hybrid multi-objective evolutionary algorithm. *Journal of Intelligent Manufacturing*, 30:2257–2272, 6 2019.
- [42] Anthony Gabriel. *The effect of internal static manufacturing complexity on manufacturing performance*. Phd thesis, Clemson University, May 2007.
- [43] L. Gaio, F. Gino, and E. Zaninotto. *I sistemi di produzione : manuale per la gestione operativa dell'impresa*. Università / [Carocci]. Carocci, 2002.
- [44] Rita Gamberini, Luca Galloni, Francesco Lolli, and Bianca Rimini. On the analysis of effectiveness in a manufacturing cell: A critical implementation of existing approaches. *Procedia Manufacturing*, 11:1882–1891, 1 2017.

- [45] David Gelernter. *Mirror Worlds or the Day Software Puts the Universe in a Shoebox: How Will It Happen and What It Will Mean*. Oxford University Press, Inc., New York, NY, USA, 1991.
- [46] Michael Grieves. Origins of the digital twin concept, 08 2016.
- [47] Xi Gu, Xiaoning Jin, and Jun Ni. Prediction of passive maintenance opportunity windows on bottleneck machines in complex manufacturing systems. *Journal of Manufacturing Science and Engineering, Transactions of the ASME*, 137, 6 2015.
- [48] Fan Guoliang, Li Aiping, Giovanni Moroni, Xu Liyun, and Liu Xuemei. Operation-based configuration complexity measurement for manufacturing system. *Procedia CIRP*, 63:645–650, 1 2017.
- [49] Sebastian Haag and Reiner Anderl. Digital twin – proof of concept. *Manufacturing Letters*, 15:64–66, 2018. Industry 4.0 and Smart Manufacturing.
- [50] Mattias Hallgren and Jan Olhager. Differentiating manufacturing focus. *International Journal of Production Research*, 44:3863–3878, 2006.
- [51] Abhishek Hazra, Mainak Adhikari, Tarachand Amgoth, and Satish Narayana Srirama. A comprehensive survey on interoperability for iiot: Taxonomy, standards, and future directions. *ACM Comput. Surv.*, 55(1), nov 2021.
- [52] Zimiao He, Kunlan Wang, Hanxiao Li, Hong Song, Zhongjie Lin, Kaizhou Gao, and Ali Sadollah. Improved q-learning algorithm for solving permutation flow shop scheduling problems. *IET Collaborative Intelligent Manufacturing*, 4:35–44, 3 2022.

- [53] Zimiao He, Kunlan Wang, Hanxiao Li, Hong Song, Zhongjie Lin, Kaizhou Gao, and Ali Sadollah. Improved q-learning algorithm for solving permutation flow shop scheduling problems. *IET Collaborative Intelligent Manufacturing*, 4:35–44, 3 2022.
- [54] Zimiao He, Kunlan Wang, Hanxiao Li, Hong Song, Zhongjie Lin, Kaizhou Gao, and Ali Sadollah. Improved q-learning algorithm for solving permutation flow shop scheduling problems. *IET Collaborative Intelligent Manufacturing*, 4:35–44, 3 2022.
- [55] Germán Herrera Vidal and Jairo Rafael Coronado Hernández. Complexity in manufacturing systems: a literature review. *Production Engineering*, 15(3–4):321–333, Jan 2021.
- [56] P T Ho, J A Albajez, J A Yagüe, and J Santolaria. Preliminary study of augmented reality based manufacturing for further integration of quality control 4.0 supported by metrology. *IOP Conference Series: Materials Science and Engineering*, 1193, 2021.
- [57] Erik Hofmann and Marco Rüsç. Industry 4.0 and the current status as well as future prospects on logistics. *Computers in Industry*, 89:23–34, 8 2017.
- [58] Karl Hribernik, Giacomo Cabri, Federica Mandreoli, and Gregoris Mentzas. Autonomous, context-aware, adaptive digital twins—state of the art and roadmap. *Computers in Industry*, 133:103508, 10 2021.
- [59] Florian Jaensch, Akos Csiszar, Christian Scheifele, and Alexander Verl. Digital twins of manufacturing systems as a base for machine learning. In *2018 25th International Conference on Mechatronics and Machine Vision in Practice (M2VIP)*, pages 1–6. IEEE, 2018.

- [60] Zohaib Jan, Farhad Ahamed, Wolfgang Mayer, Niki Patel, Georg Grossmann, Markus Stumptner, and Ana Kuusk. Artificial intelligence for industry 4.0: Systematic review of applications, challenges, and opportunities. *Expert Systems with Applications*, 216:119456, 4 2023.
- [61] Mohd Javaid, Abid Haleem, and Rajiv Suman. Digital twin applications toward industry 4.0: A review. *Cognitive Robotics*, 3:71–92, 2023.
- [62] Finn V Jensen and Thomas Dyhre Nielsen. *Bayesian networks and decision graphs*, volume 2. Springer, 2007.
- [63] Matthew Joordens and Mo Jamshidi. On the development of robot fish swarms in virtual reality with digital twins. In *2018 13th Annual Conference on System of Systems Engineering (SoSE)*, pages 411–416, 2018.
- [64] Dimitrios Kaklis, Iraklis Varlamis, George Giannakopoulos, Takis J. Varelas, and Constantine D. Spyropoulos. Enabling digital twins in the maritime sector through the lens of ai and industry 4.0. *International Journal of Information Management Data Insights*, 3(2):100178, 2023.
- [65] Cristina Orsolin Klingenberg, Marco Antônio Viana Borges, and José Antônio Valle Antunes. Industry 4.0 as a data-driven paradigm: a systematic literature review on technologies. *Journal of Manufacturing Technology Management*, 32:570–592, 3 2021.
- [66] Peter Koudal. *Mastering Complexity in Global Manufacturing: Powering Profits and Growth through Value Chain Synchronization*. Deloitte, 01 2003.

- [67] Sebastian Lang, Tobias Reggelin, Johann Schmidt, Marcel Müller, and Abdulrahman Nahhas. Neuroevolution of augmenting topologies for solving a two-stage hybrid flow shop scheduling problem: A comparison of different solution strategies. *Expert Systems with Applications*, 172, 6 2021.
- [68] Yibing Li, Zhiyu Tao, Lei Wang, Baigang Du, Jun Guo, and Shibao Pang. Digital twin-based job shop anomaly detection and dynamic scheduling. *Robotics and Computer-Integrated Manufacturing*, 79:102443, 2 2023.
- [69] Yuanyuan Li, Stefano Carabelli, Edoardo Fadda, Daniele Manerba, Roberto Tadei, and Olivier Terzo. Machine learning and optimization for production rescheduling in industry 4.0. *International Journal of Advanced Manufacturing Technology*, 110:2445–2463, 10 2020.
- [70] Chun Cheng Lin, Der Jiunn Deng, Yen Ling Chih, and Hsin Ting Chiu. Smart manufacturing scheduling with edge computing using multiclass deep q network. *IEEE Transactions on Industrial Informatics*, 15:4276–4284, 7 2019.
- [71] Peng Lin, Leidi Shen, Zhiheng Zhao, and George Q. Huang. Graduation manufacturing system: synchronization with iot-enabled smart tickets. *Journal of Intelligent Manufacturing*, 30:2885–2900, 12 2019.
- [72] Marco Lippi, Matteo Martinelli, Marco Picone, and Franco Zambonelli. Enabling causality learning in smart factories with hierarchical digital twins. *Computers in Industry*, 148:103892, 6 2023.
- [73] Qiang Liu, Chao Chen, and Shanben Chen. Key technology of intelligentized welding manufacturing and systems based on the internet

- of things and multi-agent. *Journal of Manufacturing and Materials Processing*, 6, 12 2022.
- [74] Francesco Lolli, Elia Balugani, Maria Angela Butturi, Antonio Maria Coruzzolo, Alessio Ishizaka, Simona Marinelli, and Vincenzo Romano. A decision support system for the selection of insulating material in energy retrofit of industrial buildings: A new robust ordinal regression approach. *IEEE Transactions on Engineering Management*, 2022.
- [75] Francesco Lolli, Antonio Maria Coruzzolo, Chiara Forgone, and Platao Gonçalves Terra Neto. Ergonomic risk reduction: A height-adjustable mesh truck for picking activities evaluated with a depth camera. *Ergonomics In Design*, 77, 2023.
- [76] Francesco Lolli, Francesco Lodi, Claudio Giberti, Antonio Maria Coruzzolo, and Samuele Marinello. Order picking systems: A queue model for dimensioning the storage capacity, the crew of pickers, and the agv fleet. *Mathamtical problems in engineering*, 2022.
- [77] Ping Lou, Quan Liu, Zude Zhou, Huaiqing Wang, and Sherry Xiaoyun Sun. Multi-agent-based proactive-reactive scheduling for a job shop. *International Journal of Advanced Manufacturing Technology*, 59:311–324, 3 2012.
- [78] Alexander Maier, Sebastian Schriegel, and Oliver Niggemann. Big data and machine learning for the smart factory—solutions for condition monitoring, diagnosis and optimization. In *Industrial Internet of Things*, pages 473–485. Springer, 2017.
- [79] Stefano Mariani, Marco Picone, and Alessandro Ricci. Agents and digital twins for the engineering of cyber-physical systems: opportuni-

ties, and challenges. *Annals of Mathematics and Artificial Intelligence*, 2023.

- [80] Members of the Digital Framework Task Group. White paper: The gemini principles. Technical report, Centre of Digital Built Britain, January 2018. Available at <https://www.cdbb.cam.ac.uk/DFTG/GeminiPrinciples>. Last access: 20210401.
- [81] Judith Michael, Jerome Pfeiffer, Bernhard Rumpe, and Andreas Wortmann. Integration challenges for digital twin systems-of-systems. *Proceedings - 10th IEEE/ACM International Workshop on Software Engineering for Systems-of-Systems and Software Ecosystems, SESoS 2022*, pages 9–12, 2022.
- [82] Vera L Miguéis, José L Borges, et al. Automatic root cause analysis in manufacturing: an overview & conceptualization. *Journal of Intelligent Manufacturing*, pages 1–18, 2022.
- [83] Roberto Minerva and Noël Crespi. Digital twins: Properties, software frameworks, and application scenarios. *IT Professional*, 23(1):51–55, 2021.
- [84] Roberto Minerva, Gyu Myoung Lee, and Noel Crespi. Digital twin in the iot context: A survey on technical features, scenarios, and architectural models. *Proceedings of the IEEE*, 108:1785–1824, 10 2020.
- [85] Vladimir Modrak and Zuzana Soltysova. Exploring the complexity levels of discrete manufacturing processes. *IFAC-PapersOnLine*, 52:1444–1449, 1 2019.

- [86] Rakshit D. Muddu, D.M. Gowda, Anthony James Robinson, and Aimee Byrne. Optimisation of retrofit wall insulation: An Irish case study. *Energy and Buildings*, 235:110720, 2021.
- [87] Anna Myrodiya, Lars Hvam, Enrico Sandrin, Cipriano Forza, and Anders Haug. Identifying variety-induced complexity cost factors in manufacturing companies and their impact on product profitability. *Journal of Manufacturing Systems*, 60:373–391, 7 2021.
- [88] Elisa Yumi Nakagawa, Pablo Oliveira Antonino, Frank Schnicke, Thomas Kuhn, and Peter Liggesmeyer. Continuous systems and software engineering for industry 4.0: A disruptive view. *Information and Software Technology*, 135:106562, 2021.
- [89] Seiichi Nakajima. *Introduction to TPM: total productive maintenance*. Productivity Press, 1995.
- [90] Léony Luis Lopes Negrão, Moacir Godinho Filho, and Giuliano Marodin. Lean practices and their effect on performance: a literature review. *Production Planning and Control*, 28:33–56, 1 2017.
- [91] Taiichi Ohno. *Toyota Production System: Beyond Large-Scale Production*. Productivity Press, 1988.
- [92] Jan Olhager. Strategic positioning of the order penetration point. *International Journal of Production Economics*, 85:319–329, 9 2003.
- [93] Jan Olhager. Strategic positioning of the order penetration point. *International Journal of Production Economics*, 85:319–329, 9 2003.
- [94] Antonio Padovano, Francesco Longo, Letizia Nicoletti, Lucia Gazzaneo, Alessandro Chiurco, and Simone Talarico. A prescriptive mainte-

- nance system for intelligent production planning and control in a smart cyber-physical production line. *Procedia CIRP*, 104:1819–1824, 2021.
- [95] Antonio Padovano, Francesco Longo, Letizia Nicoletti, Lucia Gazzaneo, Alessandro Chiurco, and Simone Talarico. A prescriptive maintenance system for intelligent production planning and control in a smart cyber-physical production line. *Procedia CIRP*, 104:1819–1824, 2021.
- [96] Kijung Park and Gül E. Okudan Kremer. Assessment of static complexity in design and manufacturing of a product family and its impact on manufacturing performance. *International Journal of Production Economics*, 169:215–232, 11 2015.
- [97] Judea Pearl. The seven tools of causal inference, with reflections on machine learning. *Commun. ACM*, 62(3):54–60, February 2019.
- [98] Judea Pearl and Dana Mackenzie. *The book of why: the new science of cause and effect*. Basic Books, 2018.
- [99] Bob Piascik, John Vickers, Dave Lowry, Steve Scotti, Jeff Stewart, and Anthony Calomino. Materials, structures, mechanical systems, and manufacturing roadmap. *NASA TA*, pages 12–2, 2012.
- [100] Marco Picone, Marco Mamei, and Franco Zambonelli. Wldt: A general purpose library to build iot digital twins. *SoftwareX*, 13:100661, 1 2021.
- [101] Marco Picone, Stefano Mariani, Marco Mamei, Franco Zambonelli, and Mirko Berlier. Wip: Preliminary evaluation of digital twins on mec software architecture. In *2021 IEEE 22nd International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, pages 256–259, 2021.

- [102] Lorenzo Ragazzini, Elisa Negri, and Marco Macchi. A digital twin-based predictive strategy for workload control. *IFAC-PapersOnLine*, 54:743–748, 2021.
- [103] Alessandro Ricci, Angelo Croatti, Stefano Mariani, Sara Montagna, and Marco Picone. Web of digital twins. *ACM Trans. Internet Technol.*, dec 2021. Just Accepted.
- [104] Dominik Riemer. Feeding the digital twin: Basics, models and lessons learned from building an iot analytics toolbox (invited talk). In *2018 IEEE International Conference on Big Data (Big Data)*, pages 4212–4212. IEEE, 2018.
- [105] Enrico Rossini, Marcello Pietri, Roberto Cavicchioli, Marco Picone, Marco Mamei, Roberto Querio, Laura Colazzo, and Roberto Procopio. 5g mec architecture for vulnerable road users management through smart city data fusion. In *Proceedings of the 29th Annual International Conference on Mobile Computing and Networking*, New York, NY, USA, 2023. Association for Computing Machinery.
- [106] Joze M. Rozanec, Lu Jinzhi, Aljaz Kosmerlj, Klemen Kenda, Kiritsis Dimitris, Viktor Jovanoski, Jan Rupnik, Mario Karlovcec, and Blaz Fortuna. Towards actionable cognitive digital twins for manufacturing. In *Proceedings of the International Workshop on Semantic Digital Twins co-located with the 17th Extended Semantic Web Conference, SeDiT@ESWC 2020, Heraklion, Greece, June 3, 2020*, volume 2615 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2020.

- [107] S N Samy, T Algeddawy, and H Elmaraghy. A granularity model for balancing the structural complexity of manufacturing systems equipment and layout. *Journal of Manufacturing Systems*, 36:7–19, 2015.
- [108] Roberto Saracco. Digital twins: Bridging physical space and cyberspace. *Computer*, 52(12):58–64, 2019.
- [109] Roberto Saracco. Digital twins: Bridging physical space and cyberspace. *Computer*, 52(12):58–64, 2019.
- [110] Bernhard Schölkopf, Francesco Locatello, Stefan Bauer, Nan Rosemary Ke, Nal Kalchbrenner, Anirudh Goyal, and Yoshua Bengio. Toward causal representation learning. *Proceedings of the IEEE*, 2021.
- [111] Bernhard Schölkopf, Francesco Locatello, Stefan Bauer, Nan Rosemary Ke, Nal Kalchbrenner, Anirudh Goyal, and Yoshua Bengio. Toward causal representation learning. *Proceedings of the IEEE*, 2021.
- [112] Lei Shi, Gang Guo, and Xiaohui Song. Multi-agent based dynamic scheduling optimisation of the sustainable hybrid flow shop in a ubiquitous environment. *International Journal of Production Research*, 59:576–597, 2021.
- [113] Shigeo Shingo. *A study of the toyota production system: From an industrial engineering viewpoint*. Productivity Press, 2019.
- [114] Tomé Silva and Américo Azevedo. Production flow control through the use of reinforcement learning. *Procedia Manufacturing*, 38:194–202, 2019.
- [115] Jack Sleuters, Yonghui Li, Jacques Verriet, Marina Velikova, and Richard Doornbos. A digital twin method for automated behavior

- analysis of large-scale distributed iot systems. In *2019 14th Annual Conference System of Systems Engineering (SoSE)*, pages 7–12. IEEE, 2019.
- [116] Mohsen Soori, Behrooz Arezoo, and Roza Dastres. Machine learning and artificial intelligence in cnc machine tools, a review. *Sustainable Manufacturing and Service Economics*, 2:100009, 4 2023.
- [117] Charles Steinmetz, Achim Rettberg, Fabíola Gonçalves C Ribeiro, Greyce Schroeder, and Carlos E. Pereira. Internet of things ontology for digital twin in cyber physical systems. In *2018 VIII Brazilian Symposium on Computing Systems Engineering (SBESC)*, pages 154–159. IEEE, 2018.
- [118] Fei Tao, Jiangfeng Cheng, Qinglin Qi, Meng Zhang, He Zhang, and Fangyuan Sui. Digital twin-driven product design, manufacturing and service with big data. *International Journal of Advanced Manufacturing Technology*, 94:3563–3576, 2 2018.
- [119] Fei Tao, Jiangfeng Cheng, Qinglin Qi, Meng Zhang, He Zhang, and Fangyuan Sui. Digital twin-driven product design, manufacturing and service with big data. *International Journal of Advanced Manufacturing Technology*, 94:3563–3576, 2 2018.
- [120] Fei Tao and Qinglin Qi. Make more digital twins. *Nature*, 573(7775):490–491, 2019.
- [121] Fei Tao, He Zhang, Ang Liu, and A. Y. C. Nee. Digital twin in industry: State-of-the-art. *IEEE Trans. on Industrial Informatics*, 15(4):2405–2415, 2019.

- [122] Fei Tao and Meng Zhang. Digital twin shop-floor: A new shop-floor paradigm towards smart manufacturing. *IEEE Access*, 5:20418–20427, 9 2017.
- [123] Fei Tao, Meng Zhang, and A.Y.C. Nee. Chapter 1 - background and concept of digital twin. In Fei Tao, Meng Zhang, and A.Y.C. Nee, editors, *Digital Twin Driven Smart Manufacturing*, pages 3 – 28. Academic Press, 2019.
- [124] Fei Tao, Meng Zhang, and A.Y.C. Nee. Chapter 1 - background and concept of digital twin. In Fei Tao, Meng Zhang, and A.Y.C. Nee, editors, *Digital Twin Driven Smart Manufacturing*, pages 3 – 28. Academic Press, 2019.
- [125] Khalil Tliba, Thierno M.L. Diallo, Olivia Penas, Romdhane Ben Khalifa, Noureddine Ben Yahia, and Jean Yves Choley. Digital twin-driven dynamic scheduling of a hybrid flow shop. *Journal of Intelligent Manufacturing*, 2022.
- [126] Eric J Tuegel, Anthony R Ingraffea, Thomas G Eason, S Michael Spottswood, et al. Reengineering aircraft structural life prediction using a digital twin. *International Journal of Aerospace Engineering*, 2011, 2011.
- [127] Thomas H.J. Uhlemann, Christian Lehmann, and Rolf Steinhilper. The digital twin: Realizing the cyber-physical production system for industry 4.0. *Procedia CIRP*, 61:335–340, 2017.
- [128] Jumyung Um, Jens Popper, and Martin Ruskowski. Modular augmented reality platform for smart operator in production environment.

2018 *IEEE Industrial Cyber-Physical Systems (ICPS)*, pages 720–725, 2018.

- [129] Silvestro Vespoli, Andrea Grassi, Guido Guizzi, and Liberatina Carmela Santillo. Evaluating the advantages of a novel decentralised scheduling approach in the industry 4.0 and cloud manufacturing era. *IFAC-PapersOnLine*, 52:2170–2176, 9 2019.
- [130] Brian Vo, Elif Kongar, and Manuel F Suárez-Barraza. Root-cause problem solving in an industry 4.0 context. *IEEE Engineering Management Review*, 48(1):48–56, 2020.
- [131] Matej Vuković and Stefan Thalmann. Causal discovery in manufacturing: A structured literature review. *Journal of Manufacturing and Materials Processing*, 6(1):10, 2022.
- [132] Jiafu Wan, Xiaomin Li, Hong-Ning Dai, Andrew Kusiak, Miguel Martínez-García, and Di Li. Artificial-intelligence-driven customized manufacturing factory: Key technologies, applications, and challenges. *Proceedings of the IEEE*, 109(4):377–398, April 2021.
- [133] Jin Wang, Yingfeng Zhang, Yang Liu, and Naiqi Wu. Multiagent and bargaining-game-based real-time scheduling for internet of things-enabled flexible job shop. *IEEE Internet of Things Journal*, 6:2518–2531, 4 2019.
- [134] Wenbo Wang, Yingfeng Zhang, and Ray Y. Zhong. A proactive material handling method for cps enabled shop-floor. *Robotics and Computer-Integrated Manufacturing*, 61:101849, 2 2020.

- [135] Yu Fang Wang. Adaptive job shop scheduling strategy based on weighted q-learning algorithm. *Journal of Intelligent Manufacturing*, 31:417–432, 2 2020.
- [136] Yu Fang Wang. Adaptive job shop scheduling strategy based on weighted q-learning algorithm. *Journal of Intelligent Manufacturing*, 31:417–432, 2 2020.
- [137] Peter T. Ward, John K. McCreery, Larry P. Ritzman, and Deven Sharma. Competitive priorities in operations management. *Decision Sciences*, 29:1035 – 1046, 1 1998.
- [138] William W. Weiss, Robert S. Balch, and Bruce A. Stubbs. How artificial intelligence methods can forecast oil production. *Proceedings - SPE Symposium on Improved Oil Recovery*, pages 212–227, 4 2002.
- [139] Jiahua Weng, Shota Mizoguchi, Shingo Akasaka, and Hisashi Onari. Smart manufacturing operating systems considering parts utilization for engineer-to-order production with make-to-stock parts. *International Journal of Production Economics*, 220, 2 2020.
- [140] Chin Chia Wu, Jia Yang Chen, Win Chin Lin, Kunjung Lai, Danyu Bai, and Sz Yun Lai. A two-stage three-machine assembly scheduling flowshop problem with both two-agent and learning phenomenon. *Computers and Industrial Engineering*, 130:485–499, 4 2019.
- [141] Tianfang Xue, Peng Zeng, and Haibin Yu. A reinforcement learning method for multi-agv scheduling in manufacturing. *Proceedings of the IEEE International Conference on Industrial Technology*, 2018-February:1557–1561, 4 2018.

- [142] Chaoyang Zhang and Weixi Ji. Digital twin-driven carbon emission prediction and low-carbon control of intelligent manufacturing job-shop. *Procedia CIRP*, 83:624–629, 2019.
- [143] Cong Zhang, Wen Song, Zhiguang Cao, Jie Zhang, Puay Siew Tan, and Chi Xu. Learning to dispatch for job shop scheduling via deep reinforcement learning. *Advances in Neural Information Processing Systems*, 2020-December, 2020.
- [144] Meng Zhang, Fei Tao, and A. Y.C. Nee. Digital twin enhanced dynamic job-shop scheduling. *Journal of Manufacturing Systems*, 58:146–156, 1 2021.
- [145] Ting Zheng, Marco Ardolino, Andrea Bacchetti, and Marco Perona. The applications of industry 4.0 technologies in manufacturing context: a systematic literature review. <https://doi.org/10.1080/00207543.2020.1824085>, 59:1922–1954, 2020.
- [146] Xun Zheng, Bryon Aragam, Pradeep K Ravikumar, and Eric P Xing. Dags with no tears: Continuous optimization for structure learning. *Advances in Neural Information Processing Systems*, 31, 2018.
- [147] Longfei Zhou, Lin Zhang, and Berthold K.P. Horn. Deep reinforcement learning-based dynamic scheduling in smart manufacturing. *Procedia CIRP*, 93:383–388, 2020.
- [148] Marwin Züfle, Joachim Agne, Johannes Grohmann, Ibrahim Dörtoluk, and Samuel Kounev. A predictive maintenance methodology: Predicting the time-to-failure of machines in industry 4.0. *IEEE International Conference on Industrial Informatics (INDIN)*, 2021-July, 2021.

[149] Dr. Perin Ünal. Cognitive digital twins: Digital twins that learn by themselves, foresee the future, and act accordingly, 2022.