applied sciences

Review

Digital Twins in Software Engineering—A Systematic Literature Review and Vision

Miguel A. Guinea-Cabrera and Juan A. Holgado-Terriza *

Software Engineering Department, Research Centre for Information and Communication Technologies (CITIC-UGR), University of Granada, 18071 Granada, Spain; maguinea@correo.ugr.es
* Correspondence: jholgado@ugr.es

Featured Application: Secure Integration of the IoT and Digital Twins.

Abstract: Digital twins are a powerful consequence of digital transformation. In fact, they have been applied to many industries to enhance operations, predict needs, improve decision making, or optimize performance, even though the definition of digital twins is still evolving. However, their impact on the software industry is still limited. Thus, this work aims to analyze the current adoption of digital twins in the software industry as a potential path to integrate them into application lifecycle management. To achieve this objective, first, the significant characteristics of current digital twins are analyzed in their application to manufacturing to understand how the knowledge and the lessons learned can be transferred to the software industry. Second, a systematic literature review was conducted on Scopus, the Web of Science, and the ScienceDirect database. The literature review revealed 93 documents after data screening and cleaning 251 initial documents. Our main findings are that digital twins are already influencing and will significantly affect the software industry, revolutionizing various aspects of the software development lifecycle. This study tackles what identifies a digital twin in the software industry, the specific domains and areas where they can be applied in the software lifecycle, and the proposed approaches explored to build digital twins for developing, deploying, and maintaining software systems. Finally, this study proposes some guidelines for building digital twins in the context of application lifecycle management. Determining an appropriate roadmap shortly is essential to achieve a widespread applicability to building suitable digital twins and preparing organizations for the software industry.

Keywords: digital twin; digital transformation; software industry; software development; software engineering

check for updates

Citation: Guinea-Cabrera, M.A.; Holgado-Terriza, J.A. Digital Twins in Software Engineering—A Systematic Literature Review and Vision. *Appl. Sci.* **2024**, *14*, 977. https://doi.org/10.3390/app14030977

Academic Editor: Paolino Di Felice

Received: 11 December 2023
Revised: 15 January 2024
Accepted: 17 January 2024
Published: 23 January 2024

1. Introduction

On 10 October 2016, the Fourth Industrial Revolution (4IR) was popularized by the World Economic Forum [1] from the "Industry 4.0" concept that originated in 2011 at the Hannover Fair. The 4IR introduces how a highly optimized production system capable of delivering extensive customization focused on customer satisfaction can be developed, while evaluating its potential impact on sustainability. A key characteristic of this revolution is the intensive use of technology in symbiosis with humans to complete the digital transformation toward smart factories.

Complying with the principles and fundamentals defined by the Fourth Industrial Revolution involves the application and integration of multiple technologies such as the Internet of Things (IoT), big data, mixed reality, blockchain, 5G, machine learning, cloud computing, digital twins (DTs), or cyber–physical systems (CPSs) [2,3]. From these technologies, the DT emerged as a critical technology of Industry 4.0.

Initially, a DT is defined as a virtual representation of the physical counterpart (device, machine, system), a digital model that mirrors the real-world counterpart in terms of its characteristics, behavior, and conditions [4]. For this reason, they have been applied to many industries to enhance operations, predict needs, improve decision making, or optimize performance.

The digital twin definition is still evolving in order to include a virtual representation, not only of the physical elements of a system but also its processes, models, or components to be present in the development, deployment, maintenance, and management in general of any product, as it is required in product lifecycle management (PLM). In the same way, the software industry must manage processes, models, components, and tools to make possible the development, deployment, maintenance, and management of software systems in the context of application lifecycle management (ALM).

Therefore, there are similarities between PLM (focused on hardware) and application lifecycle management (focused on software). Deuter has already published articles analyzing the convergence of both concepts [5,6]. However, the growing complexity of software engineering activities and their particularity requires setting a stable ground from the definition and identifying the relevant points to consider, setting the foundations for applying software development lifecycle management (SDLM, as a part of ALM) and ALM. This can be the reason why the impact of DTs in the software industry is still limited.

Historically, the software industry has adopted many concepts from traditional industries to improve efficiency and productivity. ALM covers the entire process from inception to the end of an application's life. It comprises several disciplines: project management, requirements management, architecture and design, software development, testing and quality assurance, deployment, maintenance, and decommissioning. In this context, DTs are particularly useful for simulating and modeling the behavior of complex systems, enhancing software applications' understanding, development, and management through their lifecycle. DTs may be used to create detailed models of software systems and the dynamics around them over the changing lifecycle flow. DTs can predict performance, improve maintainability, and evaluate changes before updating and deploying software.

For this reason, this paper focuses on exploring the status of adopting the current digital twin paradigm level within industry and how DTs have evolved to unleash the advanced capabilities of the software industry. This is achieved by a deep analysis of the literature with the application of a systematic literature review conducted on Scopus, the Web of Science, and the ScienceDirect database. This work tackles what identifies a digital twin in the software industry, the specific domains and areas where they can be applied in the software lifecycle, and the proposed approaches explored to build powerful DTs to cover ALM for developing, deploying, and maintaining software systems.

Unlike other systematic literature reviews, which focus more on the definition of DTs and their impact on manufacturing and industrial systems, such as that by Dalibor et al. [7], the review presented in this work delves into how DTs can be applied to the software industry. This work tackles what identifies a digital twin in the software industry, the specific domains and areas to which they can be applied in the software lifecycle, and the proposed approaches explored to build powerful DTs to cover ALM for developing, deploying, and maintaining software systems.

This research is structured as follows: Section 2 presents the background of this research, analyzing the origin of the DT concept, and its evolution concerning usages, issues, benefits, and drawbacks in the manufacturing sector to understand how it can be applied to the changing demands of the software industry. Afterwards, Section 3 exposes the method applied to perform the systematic literature review, including the number of documents detected in each stage of the process, and the list of the final documents to be analyzed in the research questions. Then, Section 4 shows the results of our deep literature analysis, answering the four research questions individually. Later, Section 5 includes a discussion of the main findings discovered through our research. Section 6 provides some guidelines for DT development to be considered when applied to the software industry, especially for the main issues in ALM. Then, Section 7 summarizes the main conclusions of this research. Finally, Section 8 contains some considerations on future steps that can define a roadmap for IT organizations to implement DTs as a key part of their ALM.

2. Background

Digital twins are a relatively recent technology that has evolved as the Fourth Industrial Revolution has been embraced for the ongoing transformation of industries by integrating digital technologies. This section sets the ground on the state of the art for DTs and the origin of this concept. An analysis of its adoption, especially in manufacturing, is tackled to understand its usage, benefits, drawbacks, risks, and, in general, the most significant insights. Subsequently, a brief description of the current requirements that the software industry demands is presented in contrast to manufacturing for understanding how the application of DTs can benefit the development, deployment, and maintenance of software as the main focus of this work.

2.1. DT Concept and Its Adoption

The term digital twin (DT) was coined by Michael Grieves and presented in the first executive PLM (product lifecycle management) courses at the University of Michigan in early 2002. The initial definition by Grieves [4] is based on the idea that a digital informational construct about a physical system could be created independently. Then, this digital information construct would be a twin of the information embedded within the physical system itself and would be linked with the physical system throughout the entire lifecycle of the system. From this definition, three components that conform to a DT can be distinguished: the physical system, the digital informational construct, and the connectivity between both.

According to Kritzinger, three levels of DT integration can be defined based on the type of connectivity, as in Kritzinger et al. [8]: a digital model, where the DT does not exchange data within the digital model and the physical system; a digital shadow, where changes in the physical system are communicated one way to the DS; and the digital twin, in which there is complete communication in both directions and a change in one affects the other.

Conversely, Korenhof et al. [9] describe a DT as a type of emerging technology able to discern dependencies between product, process, and operations, characteristics that remained hidden before. Moreover, a DT can make issues visible before they become critical, being able to predict trends and behaviors to optimize them. Then, in this vision, a DT can capture a digital representation of a system's operation, determine the critical elements influencing its performance, and consequently find different optimized approaches.

Complementary to DTs, several technologies (the IoT, big data, mixed reality, blockchain, 5G, machine learning, cloud computing, or cyber–physical systems) appeared simultaneously in the context of Industry 4.0 that support the development of DTs. Specifically, the cyber–physical system (CPS) concept was coined in 2006 by Helen Gill at the National Science Foundation (NSF) as a system that integrates computing and physical processes [10]. In these systems, embedded computers with sensors, actuators, controllers, and software in robots, humans, and network connectivity are combined to monitor and control physical processes, usually with real-time feedback loops of sensing, decision making, and evaluations of network compatibility, as commented by Dafflon et al. [11]. Currently, the CPS designs are supported by many technological platforms such as the IoT, Fog, Edge Cloud, or 5G [12]. Furthermore, human–robot cooperation (HRC), for instance, studies the interactions between humans and robots in critical scenarios in a factory where humans and robots must work together, applying dynamic planning and safe routes for autonomous robots, as in the study by Maruyama et al. [13].

DTs and CPSs are interrelated concepts. The critical difference between DTs and CPSs relies on a DT as a virtual model, i.e., an informational construct of a physical system. In contrast, a CPS is an enhanced physical system integrating digital intelligence and connectivity, as stated by Tao et al. [2]. CPS support DTs through their capacity to integrate and synchronize the physical world with the digital world. This is achieved by collecting, transmitting, and processing the real-time data generated by sensors in the physical world. Once these data are processed, they might be used to create and update a DT, covering a virtual simulation of the physical world and all its relevant aspects. The relationship between a CPS and a DT is bidirectional, since the insights obtained by a DT can be used to change the behavior of the physical world through actuators—for instance. Even though the DT concept was coined in early 2002, its maturity was gained in the 2010s when it started to gain traction, as CPSs offered a bridge between the physical and digital worlds. Somers et al. [14] proposed using the emerging concept of DTs to help test and enhance the CPS development phases.

The definition of a DT continues to evolve as its usage is widespread in different areas. Semeraro et al. [15] analyzed different DT definitions and the specific issues that characterize the term. They reviewed 30 DT definitions, grouping them into five clusters to extract the primary features, and summarized DTs as follows: "A set of adaptive models that emulate the behavior of a physical system in a virtual system getting real-time data to update itself along its life cycle. The DT replicates the physical system to predict failures and opportunities for change and prescribe real-time actions for optimizing or mitigating unexpected events by observing and evaluating the operating system profile".

The adoption of the DT from its origin was oriented mainly toward manufacturing, where it is applied extensively. For this reason, our first step in this research was to analyze DTs' impact on manufacturing, specifically in smart manufacturing. Then, a literature search was performed in research databases such as Scopus and the Web of Science, using the terms "DT" and "manufacturing". In this case, only open-access research publications from 2020 were extracted for the analysis.

The first study analyzed DT usage in manufacturing, considering the most relevant issues and the primary purposes. Accordingly, a taxonomy (Table 1) was elaborated, collecting a list of these issues, which are summarized below:

- Monitoring and control: DTs mainly focus on monitoring assets to gain knowledge about decisive factors that can impact them. This asset understanding can be applied for different usages, such as anomaly detection, as for Calvo-Bascones et al. or Latsou et al. [16,17], or evaluating the status, history, or need for maintenance during the industrial process, especially in the supply chain, as with Dietz et al. [18].
- Quality: Research related to quality in its distinct aspects, such as inspections, verification, or defect classification, are often areas where DT applications can be involved. Sommers et al. [14] propose using DTs for CPS testing. Zheng et al. [19] define an approach to building a quality-oriented DT for manufacturing processes by combining them with multiple agents.
- The intelligent design of products and manufacturing processes: A significant body of work is focused on DTs' applicability in collaborative design, modeling, prototyping, and simulation at different stages, as well as team-based scrutiny of manufacturing processes. They also include frameworks or methods that combine or integrate the use of DTs in the design steps of manufacturing processes. For example, Nielsen et al. [20] research optimizing product design in product families to fit MMSs (matrix-structured manufacturing systems). In contrast, Cimino et al. [21] focus on the practical design of production lines.
- Intelligent planning, process, and production control: In these works, the building of the outcome of the value stream starting from the initial plan, the scheduling of the process chain at different steps, and its adaptation to produce variations and control over the process were the relevant issues for manufacturing. Chiurco et al. [22] used rover data modeling and machine learning (ML) to enable DTs in adaptive planning and control, as they are a good fit for dynamic production scheduling, dynamic performance optimization, process automation, and control. Likewise, Negri et al. [23] focus on production scheduling.

- Intelligent maintenance: Maintenance is a complementary issue linked to the design and building of manufacturing processes. Then, assuring and improving the maintenance of assets during the building of the products and in the post-building phase was recommended. Every unplanned stop in the product manufacturing process could mean a significant amount of time and cost increments. Neto et al. [24] is an excellent example of running simulations for opportunistic preventive maintenance scheduling.
- Decision making/support: DTs can help to assist in the decision or support of manufacturing products actively or passively. For instance, Villalonga [25] describes a dynamic scheduling decision-making framework based on DTs.
- Extension of product as service: Some works, as Laukotka [26] suggests, use DTs to enable product service strategy (PSS) in organizations to have more stages or steps in their product lifecycle. They also provide variations in the final product and empower digital versions to extract customer data, as with Wilking et al. [27].
- Value and supply chain: Many DTs are focused on resource procurement and supply management. For instance, Rasor et al. [28] use a systematic framework to address the collaborative development of DTs in manufacturing value chains. On the other hand, Moder et al. [29] analyze the relevant usage of semantic web technologies on DTs for the digitalization of supply chain processes. DTs can help select alternatives to increase resilience to be sure there is no stop in the manufacturing process when the simulation predicts potential issues.
- Resilience, cybersecurity: The improvements in product security and resiliency concerning the availability of assets and processes are usually recurrent concerns in designing and building manufacturing processes that DTs can validate before the actual deployment and start-up of the system. Papacharalampopoulos et al. [30] specify a roadmap for designing and implementing DTs to add agility and resilience to manufacturing. In particular, Empl et al. [31] developed a cybersecurity framework based on DTs to analyze the vulnerabilities of IoT systems applying the SOAR (security orchestration, automation, and response) paradigm.
- Continuous improvement and optimization: DTs are specialized in continuous improvement methods such as kaizen and optimization. Umeda [32] introduces the extension of DTs as digital triplets to add kaizen activities for continuous improvement between engineering cycles with educational purposes. On the other hand, Ferriol-Galmés et al. [33] cover building a DT for network optimization using neural networks so the DT can accurately estimate relevant SLA metrics for network optimization, as well as performance and optimization, like for Petri et al. [34], which use DTs better to understand the complex interplay between environmental variables and performance so the infrastructure gains resilience.
- General purpose and design of DTs: Some works are focused on the techniques and architectures required for DT generation. Efforts in this regard, like by Duan et al. [35], try to propose developing a standardized DT model. On the other hand, Göllner [36] presents guidelines for modeling DTs and their content to be interoperable and collaborative as a production plant can be seen as a system of systems (SoS) that works together towards a purpose. From another perspective, Kugler [37] provides a method for visualizing and defining use cases for DTs.
- Project management, cost reduction, and ROI: Some approaches propose using DTs related to project management for cost estimation (such as Farsi et al. [38]) and reduction, return on investment (ROI), and evolution measurement. Hickey et al. [39] discuss, on the other hand, the support that DTs can offer to project managers with more visual and effective communication methods. They also remark on the potential of DTs in risk and resource management.
- Sustainability: A claim on the importance of efficiency, which can be gained in the manufacturing process with energy consumption, recycling, or reusing, is appreciated in different articles. For example, Mouthaan et al. [40] discuss how twin transition and digitalization can contribute to sustainability and progress. Decarbonization and

dematerialization are increasingly applied. Chen et al. [41] propose a framework to support environmental sustainability through lean principals.

- Training/knowledge transfer: DTs can help teach engineering and transfer knowledge at different steps of the process and across departments. Maschler et al. [42] cover a positive feedback contribution to learning process acceleration through DTs.
- Emotion-aware processes: In environments where robots and humans interact, an awareness of fatigue levels and emotions are essential to avoid accidents, defects, and to protect people and assets, contributing to employee satisfaction by following human-centered processes. Florea et al. [43] describe many use cases: improved information delivery, ergonomics, professional development at enterprise scale.

Table 1. Main usages of DTs in smart manufacturing.

Area	Articles	References
Monitoring and Control	11	[16–18,41,44–50]
Quality	6	[14,19,44,51–53]
Intelligent Design	12	[20,21,44,52,54–61]
Intelligent Planning, Process and Production Control	23	[22–25,44,52,59,62–77]
Intelligent Maintenance	12	[24,44,54,55,78–85]
Decision Making/Support	7	[24,25,64,86–89]
Extension of Product as a service	5	[26,27,90–92]
Value and Supply chain with suppliers and third parties	8	[28,29,52,93–97]
Resilience, cybersecurity	4	[30,31,52,73]
Continuous Improvement, Optimization	9	[27,32–34,62,90,98–100]
General Purpose, Design of DT	3	[35–37]
Project Management, Cost Reduction, ROI	3	[38,39,101]
Sustainability	5	[34,40,41,102,103]
Training, Knowledge transfer	3	[42,104,105]
Emotion-aware processes	2	[43,106]
Total analyzed articles	**94**	

This list gives us an overview of how DTs have been used in manufacturing and what aspects may be covered in the software industry, particularly in ALM. However, the application of DTs in manufacturing may pose some drawbacks that limit their usage. The main drawbacks are described in Table 2.

- Heterogeneous data, harmonization, integrations, and interoperability: Despite the efforts of the industry to set standards, there are issues connected to the diversity of data when integration from manufacturing process sources needs to be consolidated for a high-fidelity representation. These data from different value-chain layers become more relevant when harmonized at different scales and semantically structured to simplify the conversion into valuable information. Talkhestani et al. [63] mention the heterogeneity between models and their relationship in the DT as one of the top challenges observed in the field.
- Preparation and redefinition of the human role, interaction, and cultural acceptance: The human factor in any part of the manufacturing process and the active role of this issue can reduce the ability to create human-centered processes with less friction on

cyber–physical systems. Ahmadi et al. [55] discuss the role and evolution of humans as they interact with recent technologies and how future skills might fit with existing roles differently.

Table 2. DT's main drawbacks in smart manufacturing.

Area	Articles	References
Heterogeneous data, harmonization, integrations, interoperability	16	[17,18,24,28,32,36,46,53,62,63,77,81,85–87,105]
Preparation and redefinition of human interaction, culture acceptance	5	[26,44,45,49,67]
Data quality: Incomplete documentation, binary data, real historical data on assets/Cold start, dark data, siloed info	16	[17,20,23,29,31,37,42,51,63,71,80,82,84,89,90,107]
Privacy/Cybersecurity/Ethics	7	[27,31,45,55,72,90,106]
Lack of General Framework, DT definition, and benefits	14	[16,34,35,39,41,45,54,61,68,72,79,92,98,101]
The complexity of systems/products, continuous change	9	[32,41,50,52,59,64,81,91,93]
Lack of real-world applications implemented	8	[22,41,45,58,74,75,83,95]
Data infrastructure, talent in data science knowledge, high fidelity in mirrored information:	12	[25,34,45,47,65,69,70,72,73,78,81,97]
Lack of more research results	12	[19,21,57,66,68,88,94,96,100,102–104]
The complexity of the DT model design and interpretation	8	[30,38,40,48,56,76,81,99]
Bias, Coding bias	2	[14,60]
Total analyzed articles	**94**	

- Data quality: Incomplete documentation, binary data, accurate historical data on assets, cold starts, dark data, and siloed information cannot be collected appropriately. Apart from identifying the sources, data preparation could be a complex issue with a lack of documentation. There is no previous knowledge about the availability of certain types of data and no previous experience integrating the data from various parts of the organization. Ehrhardt et al. [107] share the difficulties with data quality since data are recorded manually from the production systems. The accuracy of these data for optimization purposes may lead to wrong actions and decisions.
- Privacy/cybersecurity/ethics: Personally identifiable information (PII) data and information deducted from DTs or other sources raise significant concerns among the articles reviewed. For example, Neguina et al. [106] comments that developing these systems involving personal data is subject to cybercrime and non-ethical usage opportunities.
- Lack of a general framework, DT definition, and benefits: Kuehner [45] reports gaps in the DT definition and the importance of establishing a standard framework for DTs' definition. For instance, Calvo-Bascones [16] introduces variations in the definition of DTs and provides different methods to detect anomalies with DTs, but none are accepted as a general approach.
- The complexity of systems/products, continuous change: The evolution of customer demands and the need for efficiency result in a changing manufacturing process ecosystem with increasingly complex and fully automated behaviors. As a reference, Van Dinter [81] mentions that the complexity of models is one of the key issues to cover, together with the computational workload due to the variety of data, assets, and components. As Ruzsa [52] considers, DTs can help to tackle this continuous

change, but to build them, the article recognized a considerable effort in organization architecture, big data solutions, and digital transformation.

- Lack of real-world applications: Many works indicate a lack of tested initiatives for long-term real-world scenarios, and existing scenarios have many constraints to verify their efficacy. Chen et al. [41] describe the lack of practice-based frameworks and operational and implementation guidelines in the existing scenarios as a top issue.
- Data infrastructure, talent in data science knowledge, and high fidelity in mirrored information: The explosion of big data can pose problems in capturing a sufficient variety of data to mirror physical systems into a DT. Data science can minimize its impact. Kumbhar [69] believes that data science knowledge is a critical capability for industries to implement DTs-related technologies and is a potential constraint. The main reasons are that the infrastructure costs grow remarkably, and the available talent to apply data science remains limited.
- Lack of research results: There are insufficient research results in specific areas to compare and build better proposals for setting the basis for DTs. Ragazzini et al. [66] summarize a lack of concerns in specific applicability areas. Meanwhile, Langlotz [103] highlights the lack of research for DTs operating in physics and data-driven models required for industrial cases.
- The complexity of DT design and interpretation: The interpretation and design of DT dynamics are rather complex issues when used in automatic decisions. Farsi et al. [38] show complex scenarios for DTs due to a lack of data or uncertainty. This makes the design of the techniques and their interpretation more complex.
- Bias and coding bias: Simulations and results from DTs may include undesirable constraints or limitations based on training data and the process selected to reflect reality and generate simulation-based services from the virtual models. Creating tech debt in DTs can be easy from the first iteration by introducing bias towards specific options. This can be risky for the success of their implementations, as stated by Ng et al. [60].

2.2. DTs in the Software Industry

Regarding the software industry, the first company founded to provide software products and services was the Computer Usage Company in 1955, as stated by Kubie [108]. The global software product market in the software industry amounted to USD 968.25 billion (about USD 3000 per person in the US) in 2021. The market is expected to reach USD 1493.07 billion (about USD 4600 per person in the US) in 2025, according to the Software Products Global Market Report 2023 Edition. Digital transformation impacts most companies' transition to the cloud, data governance, and regulation adaptation. However, there are other issues to learn from smart manufacturing to reach Software Industry 4.0 as the industry is already thinking about Industry 5.0.

Historically, the software industry has adopted many concepts from traditional industries to improve efficiency and productivity. Nevertheless, both sides have crucial differences because the extreme adaptability and changing market of the software industry make achieving stable requirements nearly impossible. Then, it may be impossible to reach an objective conclusion about whether a software system meets its specifications, as stated by Sommerville [109].

Some of the areas where the methods and techniques of traditional industries are successfully adopted in the software industry are as follows:

1. Project management: A key concept adapted with different approaches and frameworks based on lean principles from the Japanese industry. An example could be Kanban from Toyota production lines.
2. Quality assurance and quality control: Although software quality is not directly comparable with quality in manufacturing, disciplined approaches such as Six Sigma—one of the most prevalent manufacturing philosophies—are applied in the software industry.

3. Software engineering: This field has been applied to the software industry, bringing principles and methods from traditional engineering. Lean manufacturing principles have been translated into software engineering [110].
4. Continuous improvement: Inspired by the Deming cycle. This is the spirit of many software industry processes, techniques, and standards, such as the security information management system (ISO27001 [111]).
5. Operations: Advanced manufacturing has impacted the software industry in process automation and delivery, automatized testing, reliability, and supply chain management, among others. Integration into the software development process opened the recent DevOps paradigm.
6. Security: Reaching high-level IT security is mandatory for current software products from their inception to avoid possible cyber-attacks or information theft. Security directly impacts a software product through the inclusion of development practices to strengthen security and compliance and the application of tools to improve products through a continuous static and dynamic analysis of the potential vulnerabilities at any stage of the software development pipeline.

Even though there are distinct categories in software development, such as programming services, system services, open-source tools, or SaaS (Software as a Service), all share common practices to adapt software products and value streams to the exigency level of customers. In this context, ALM is the PLM of computer programs, whereas PLM focuses more on hardware. This is precisely the origin of DTs, where Grieves [4] created the concept of the DT. Consequently, we hypothesize that all work coming from smart manufacturing related to PLM could be an excellent input for ALM.

Traditionally, the products in the industry, including hardware and software, were managed by PLM. However, the software industry shift starts with managing software products with an ALM paradigm. Research such as that by Deuter et al. [6] pointed out that the integration mechanisms between ALM and PLM can be achieved through an apparent convergence with DTs.

ALM covers the entire process from inception to the end of an application's life. It comprises several disciplines: project management, requirements management, architecture and design, software development, testing and quality assurance, deployment, maintenance, and decommissioning. SDLM is a subset of ALM covering only the phases of software development. While ALM helps to make better and brighter decisions about efficiently managing software, the software development life cycle (SDLC) helps to create robust software. ALM continues after development until the application is no longer used and may span many SDLCs.

Chapell [112] identifies mainly three areas in ALM: governance, which includes all of the decision making and project management for the application; development, the process of creating the application, which can reappear several times and which is linked directly with the SDLC; and operations, that is, all the work required to run and manage the application. From a standards perspective, ISO/IEC 12207 [113] can be taken as a reference for ALM. It includes not only main processes such as acquisition, procurement, development, operation, or maintenance, but also support processes such as documentation, configuration management, quality assurance, V&V (verification and validation), joint revision, auditing, and problem resolution, as well as organizational processes such as management, infrastructure, improvement, and human resources. Some people also refer to ADLM (application development lifecycle management) to include DevOps as a valuable piece of collaborative culture, principles, and practices towards products. Furthermore, including good practices for guaranteeing security in software design and continuous vulnerability analysis promotes the DevSecOps paradigm.

Although DTs are particularly useful for simulating and modeling the behavior of complex systems, including software and hardware, they can provide potential advantages in the context of ALM or the SDLC. DTs can enhance software applications' understanding, development, and management through their lifecycle. DTs may be used to create detailed models of software systems and the dynamics around them over their changing lifecycle flow. DTs can predict performance, improve maintainability, and evaluate changes before updating and deploying software.

Antonino et al. [114] and Nakagawa et al. [115] offer excellent examples of how Industry 4.0 requires continuous engineering monitoring practices for quality properties over a software or system architecture, and the applicability of DTs to simulate the evolving architecture and its evaluation.Likewise, Jones et al. [116] apply version control and DTs from conceptual design phases to physical prototypes using DTs to maintain synchronization. These works show how traditional and software industries converge within PLM and ALM, and how DTs can help to extend the capabilities of the value stream in both PLM and ALM. This evidence drives our ambition to explore the possibilities of DT usage in ALM.

3. Systematic Literature Review

This systematic literature review aims to focus on DTs and their applicability to the software industry around SDLM and ALM. Our approach is based on the procedure defined by Kofod-Petersen [117] and aligned with the PRISMA 2020 declaration [118]. This review was performed on 1 December 2023.

The sources investigated for this review are listed below:

- PIS_1: Scopus of the Elsevier database, available electronically at https://www.scopus.com;
- PIS_2: the Web of Science, available electronically at https://www.webofscience.com/wos/woscc/basic-search;
- PIS_3: Science Direct, available electronically at http://www.sciencedirect.com.

The method of performing the systematic literature review is schematized in Figure 1. From the selected sources of information, a search criterion was applied to extract documents potentially interesting for this review. In our case, the selected criteria to perform the search on the different PISs were applied to the title, abstract, and keywords:

(*"digital twin" AND "software development"*)

OR

(*"digital twin" AND "software engineering"*).

As a result, a pool of documents, specifically 251 articles, were extracted from 210 documents from Scopus, 29 from the Web of Science, and 12 from ScienceDirect. Subsequently, two simple operations were performed on the list of documents to remove duplicated articles and, secondly, the documents related to congress books and not to articles, obtaining 193 documents. In order to analyze the most representative works related to this review, a snowballing procedure was also applied, incorporating 31 additional articles. Then, after merging, the result was a list of 224 documents.

Later, a selection study was conducted by assigning a score based on the general understanding of the article's purpose and whether this was related to any aspect of this review. The scoring relevance was based on our interest in the SDLC or ALM for the software industry. At the same time, an exclusion criterion was applied to this selection. Expressly, an article was excluded when it is focused on the following topics as the primary goal:

- The IoT, sensors, actuators, smart cities, bridges, construction, and robots;
- Augmented reality, 3D, virtual reality, and artificial vision;
- Healthcare, construction, manufacturing, and vehicles.

The exclusions focus on articles explicitly discussing software engineering as the primary goal. After applying exclusions, a manual analysis was covered to subtract articles where DTs are not the central area; maybe the term is mentioned in the text, but it is not the subject of the text. Finally, a list of 93 documents was obtained, as shown in Table 3.

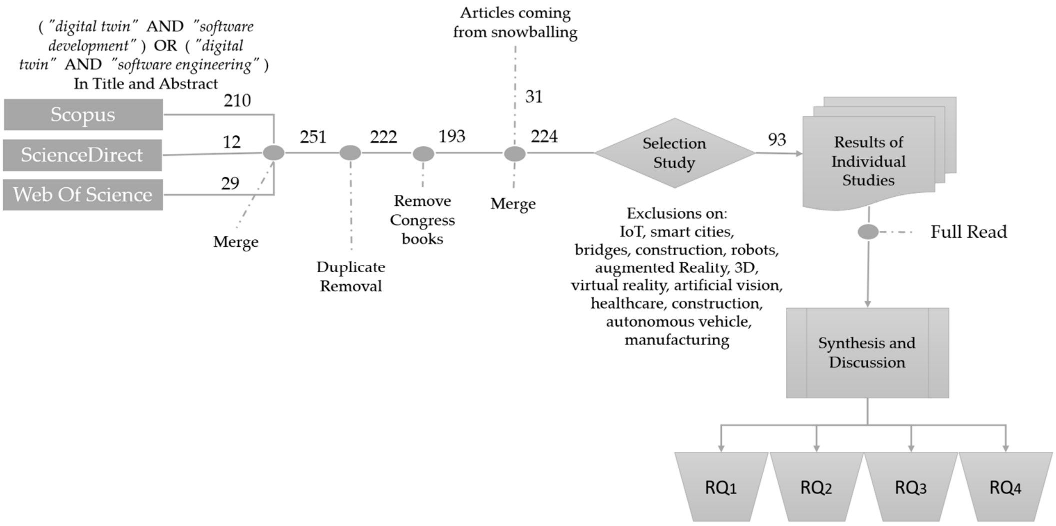

Figure 1. Method applied for the systematic literature review.

Table 3. Article selection.

PIS	Author	Year	Publication	Reference
Scopus	West et al.	2017	Conference	[119]
ScienceDirect	Hofmann et al.	2018	Conference	[120]
Web of Science	Bauer et al.	2019	Conference	[121]
Scopus	Cioroaica et al.	2019	Conference	[122]
Scopus	Loizou et al.	2019	Conference	[123]
Scopus	Eisentrager et al.	2019	Conference	[124]
ScienceDirect	Deuter et al.	2020	Journal	[6]
ScienceDirect	Caporuscio et al.	2020	Conference	[125]
ScienceDirect	Succar et al.	2020	Journal	[126]
Web of Science	Minerva et al.	2020	Journal	[127]
Web of Science	Gennady et al.	2020	Conference	[128]
Scopus	Pokhrel et al.	2020	Conference	[129]
Scopus	Hugues et al.	2020	Conference	[130]
Scopus	Dalibor et al.	2020	Conference	[131]
Scopus	Sun et al.	2020	Conference	[132]
Scopus	Xu et al.	2020	Conference	[133]
Scopus	Pileggi et al.	2020	Conference	[134]
ScienceDirect	Zhang et al.	2021	Journal	[135]
ScienceDirect	Davila Delgado et al.	2021	Journal	[136]
ScienceDirect	Bruneliere et al.	2021	Journal	[137]
ScienceDirect	Eiden et al.	2021	Conference	[138]
Web of Science	Nakagawa et al.	2021	Journal	[115]
Web of Science	Oakes et al.	2021	Conference	[139]
Web of Science	Cheng et al.	2021	Conference	[140]
Web of Science	Ahlgren et al.	2021	Conference	[141]
Web of Science	Asadi, AR	2021	Conference	[142]
Web of Science	Strandberg et al.	2021	Conference	[143]
Scopus	Autiosalo et al.	2021	Journal	[144]
Scopus	Brockhoff et al.	2021	Conference	[145]
Scopus	Jordan S.	2021	Conference	[146]
Scopus	Malakuti S.	2021	Conference	[147]
Scopus	Schroeder et al.	2021	Journal	[148]
Scopus	Engels G.	2021	Conference	[149]
Scopus	Poltronieri et al.	2021	Conference	[150]
Scopus	Muñoz et al.	2021	Conference	[151]
Scopus	Jones et al.	2021	Conference	[116]
Scopus	Fehlmann et al.	2021	Conference	[152]
ScienceDirect	Asikainen et al.	2022	Journal	[153]
ScienceDirect	Karagiannis et al.	2022	Journal	[154]
ScienceDirect	R. Subha et al.	2022	Journal	[155]

Table 3. *Cont.*

PIS	Author	Year	Publication	Reference
ScienceDirect	Ferreira et al.	2022	Journal	[156]
ScienceDirect	Vyhmeister et al.	2022	Journal	[157]
Web of Science	Das et al.	2022	Conference	[158]
Web of Science	Dobaj et al.	2022	Conference	[159]
Scopus	Rivera et al.	2022	Journal	[160]
Scopus	Kamburjan et al.	2022	Conference	[161]
Scopus	Lee et al.	2022	Conference	[87]
Scopus	Nakajima et al.	2022	Conference	[162]
Scopus	Bechu et al.	2022	Conference	[163]
Scopus	Guzina et al.	2022	Journal	[164]
Scopus	Frick et al.	2022	Journal	[165]
Scopus	Michael et al.	2022	Conference	[166]
Scopus	Bano et al.	2022	Journal	[167]
Scopus	Oliveira Antonino et al.	2022	Journal	[114]
Scopus	Kholkar et al.	2022	Conference	[168]
ScienceDirect	Epiphaniou et al.	2023	Journal	[169]
ScienceDirect	Hu et al.	2023	Journal	[170]
ScienceDirect	Alvarez-Rodríguez et al.	2023	Journal	[171]
ScienceDirect	Kügler et al.	2023	Journal	[172]
Scopus	Lu et al.	2023	Journal	[173]
Scopus	Lünnemann et al.	2023	Journal	[174]
Scopus	Ardito et al.	2023	Conference	[175]
Scopus	Frepoli et al.	2022	Conference	[176]
Scopus	Gorodetsky et al.	2020	Journal	[177]
Scopus	Newrzella et al.	2022	Journal	[178]
Scopus	Yue et al.	2023	Conference	[179]
Scopus	Dalibor et al.	2022	Journal	[7]
Scopus	AboElHassan et al.	2023	Journal	[180]
Other*	Rios et al.	2019	Conference	[181]
Scopus	Halenar et al.	2019	Conference	[182]
ScienceDirect	Hillenbrand et al.	2021	Conference	[183]
ScienceDirect	Liyanage et al.	2022	Conference	[184]
Scopus	Reiche et al.	2021	Conference	[185]
Scopus	Tisi et al.	2021	Conference	[186]
Scopus	Xia et al.	2019	Conference	[187]
Scopus	Feng et al.	2022	Conference	[188]
Scopus	Carver et al.	2022	Journal	[189]
Scopus	Al-Najjar et al.	2023	Journal	[190]
Scopus	Adams et al.	2022	Journal	[191]
Scopus	Lestingi et al.	2023	Journal	[192]
Scopus	Reed et al.	2021	Journal	[193]
Scopus	Djukić et al.	2023	Journal	[194]
Scopus	Khalajzadeh et al.	2021	Conference	[195]
Scopus	Kirchhof et al.	2020	Conference	[196]
Other*	Tsiatsis et al.	2019	Journal	[197]
WOS	Turk et al.	2020	Journal	[198]
Other*	Zheng et al.	2021	Conference	[199]
SCOPUS	Ferko et al.	2022	Journal	[200]
Other*	Boyes et al.	2022	Journal	[201]
Other*	Corradini et al.	2022	Journal	[202]
SCOPUS	Chaudhary et al.	2022	Conference	[203]
Other*	Schönig et al.	2022	Journal	[204]
Other*	Tekinerdogan et al.	2020	Journal	[205]

Other* in Table 3 means coming from other sources through snowballing.

A first analysis of the selected documents offered 51 works presented at conferences and 42 works in journals. With the resulting articles, a deeper analysis was implemented, driven by research questions. This creates a basis for shaping a general idea and conclusions. The set of research questions formulated in this study is described below:

- **RQ1**: Which assets can DTs cover in software development?
- **RQ2**: Is a DT necessary for SDLM and ALM in the software industry?
- **RQ3**: How can a DT be built in the context of the software industry?
- **RQ4**: What are the uses of DTs that can be applied to the software engineering area, specifically ALM?

During the exposition, there will be some references to the general research focused on smart manufacturing for comparison.

4. Results

4.1. RQ1: Which Assets Can DTs Cover in Software Development?

Returning to the definition of DTs, are there impediments to having DTs integrated into the software industry? According to Semeraro et al. [15], most definitions consider DTs a digital representation of a physical entity, as in Grieves' original definition. However, Dalibor et al. [7] extend the DT counterpart to physical entities and biological beings, individual beings, processes, products, a system of systems (SoS), and others, all coming from the real world. Even in the manufacturing domain, some artifacts can be potentially virtualizable entities.

Ahlgren et al. [141] state that the "physical context" restricts the capacity for DTs, being necessary to expand it to a more general concept as a digital asset. For instance, they tackle the benefits that cyber–cyber and cyber–physical DTs can deliver on Facebook. However, does it make sense to have DTs modeling synthetic or digital assets?

In the referred literature, some examples are commented on by Ahlgren et al. [141] for Facebook cyber–cyber DTs. Calvo-Bascones et al. [16] also address virtual models of synthetic physical entities with DTs, such as linked behaviors between entities that are part of the same system. This information is a valuable source for DT technology to detect anomalies.

According to the Industrial Internet Consortium (IIC), a DT is defined as a digital replica of an asset that captures the attributes and behaviors of that asset. From an information security perspective, the term asset under ISO/IEC 27001 [111] is defined as "anything that has value for the organization".

From Oakes [139], we observe the replacement of the physical entity into something more generic called an SuS (system under study). Oakes also includes the context (environment) as part of the SuS. This context may include humans or agents. Our premise is that this provides a good starting point to build a realistic model for the first stage for DTs in the software industry. So, we propose to use a modified version of that of Semeraro et al. [15]: "A set of adaptive models that emulate the behavior of a System under Study (SuS) in a virtual system getting real-time data to update itself along its life cycle. The DT replicates the SuS to predict failures and opportunities for change, to prescribe real-time actions for optimizing and mitigating unexpected events observing and evaluating the operating system profile."

DTs help to acquire comprehension of the SuS and exert control over it. Furthermore, DTs are rather adaptive in the sense that they can achieve the following:

- Join efforts into a network of DTs to have an SuS DT, as with Autiosalo et al. [144];
- Cooperate with multi-agent systems to achieve a broader capacity, as with Latsou et al. [17];
- A capacity to add AI and machine learning to DTs, as with Ricci et al. [206].

In this context, the DT will help with the governance and siloed data (Malakuti [143]) from different elements to be combined and used effectively. Then, aspects such as IT governance, application lifecycle management, or product lifecycle management can be covered more effectively by DTs. Xu [133] establishes a research agenda using a three-dimensional framework to investigate DTs' ecosystem context.

An example could be that proposed by Fehlmann et al. [152], using DTs in agile real-time testing to achieve the DevTestOps approach, in which test plans are created in real time and automized to evaluate working software from the beginning. Therefore, this approach extends TDD (Test Drive Development) principles.

4.2. RQ2: Is a DT Necessary for SDLM and ALM in the Software Industry?

Kephart and Chess [207] stated on the vision of autonomic computing based on the 2001 IBM manifesto that due to the complexity of modern systems, they need to be self-managed. DTs were usually adopted in areas unrelated to software development, such as manufacturing, construction, mechanical engineering, and health. In these areas, DTs became powerful tools to tackle complexity, substituting or enhancing expensive steps with a digital counterpart capable of improving the process involved with a reduction in the overall cost and risk on the system.

From the publications analyzed, five main reasons are identified by authors to sustain an investment in DTs. These reasons define the main areas in the software industry where DTs can have a positive impact:

1. The complexity of current systems: The complexity of the SDLC requires simulation capabilities for the automation efficiency that is needed for the continuous improvement of aspects of the value stream and the product. Al-Najjar et al. [190] use virtual infrastructure DTs to help with the complexity of complex workflow ecosystems. In contrast, Oliveira Antonino et al. [114] highlight the need for appropriate methods and tools to enable continuous and accurate assessments of the quality of system architectures so that it is not a siloed territory based on the expertise of a few engineers. Also, Asikainen [153] observes that the complexity of software process management grows as the number of related decisions increases, offering a potential framework to tackle this complexity during software processes. Having expertise in all the areas of the lifecycle of products is quite complex and requires enough resources with a level of infrequent expertise to cover all needs. Ardito et al. [175] comment on the need to rethink the interplay between human–computer interactions and software engineering for a rapid response to the evolution of technologies. It also set the DT as a protagonist of the digital transformation process.
2. Analysis, design, prognosis, planning, and rapid response, even for heterogeneous vendors: DTs can adaptively monitor operations and value streams and improve reaction times. For instance, Brockhoff [145] explores combining process mining techniques with model-driven DTs to efficiently combine data and models at runtimes applied to conformance-checking techniques. Caporuscio et al. [125] speak about smart troubleshooting to analyze information from various sources and find relationships with troubleshooting instructions and software fixes. The whole product cycle is suggested to be covered by Halenar et al. [182] and Reiche et al. [185] with an approach inspired by DevOps. Frepoli et al. [176] present the creation of an agile digital platform to facilitate the orchestration of complex workflows to identify risks in the design process. The benefits of applying DTs in different use cases can be appreciated in this paragraph. Planning the prioritization of these US resources also requires a methodology, as exposed by Newsrella et al. [178], that helps give a response to business objectives and challenges.
3. Knowledge sharing to improve collaborative processes: DTs help to support engineering by reinforcing knowledge-sharing practices and maintaining information isolation, transparency, and evolution tracking on the product side. Jordan [146] applies DTs to mitigate the risk of poorly documented architectures and again discusses the complexity of the software development process.
4. Replication of human skills: Some recent works propose the development of DTs linked to replicating human profiles. This area could grow significantly by including massive trends, such as LLM (large language model) systems like ChatGPT. In this way, Asadi et al. [142] comment on cognitive DTs to turn users' data into a future DT of users, while Ahlgren et al. explore [141] the simulation of a cyber entity rather

than the traditional approach of mirroring a physical entity. In this case, the Facebook www platform can map users' relationships and social interactions.

5. Ethics and review on decision making from algorithms and automatic processes. This is linked to auditing purposes. Lu et al. [173] propose the application of ethical DTs to artificial intelligence (AI) to evaluate transparency, trustworthiness, and bias in decisions. Similarly, Yue [179] follows the same approach of using DTs to verify decision making under uncertainty. Likewise, Cioroaica et al. [122] use DTs as intelligent agents to assess the runtime behavior of real system components. Furthermore, Muñoz et al. [151] also propose a framework to build and test DTs to transparently verify their expected behaviors in their early development stages and validate their effectiveness.

Evaluating the reasons covered by the analyzed works in Table 4 to include or manage DTs in the software industry, the weight of importance of each area can be determined according to the percentage of works found for each target. Thus, Figure 2 graphically shows the percentage of each area. According to it, areas 1 and 2 capitalize 64% of the importance of using DT in the software industry. That is, the fundamental reasons are focused on mitigating the complexity of software and improving the analysis, design, prognosis, planning, and rapid response to the software development process.

Table 4. Main areas in ALM for using DTs.

Area	Articles	References
Analysis, design, prognosis, planning, and rapid response	26	[87,125,126,129,131,134,136,145,149,150,158,165,168, 169,174,176,178,180–188]
Replicate cognitive capabilities	3	[141,142,189]
Complexity and level of abstraction demand automation, efficiency, and simulation capabilities	21	[114,115,121,131,135,137,141,152,153,156,159,160,162, 163,175,177,190–193,200]
Ethics and review of decision making of algorithms and teams	9	[120,122,131,151,157,168,171,173,179]
Knowledge sharing	15	[123,124,128,130,138,146,147,165,172,194–199]

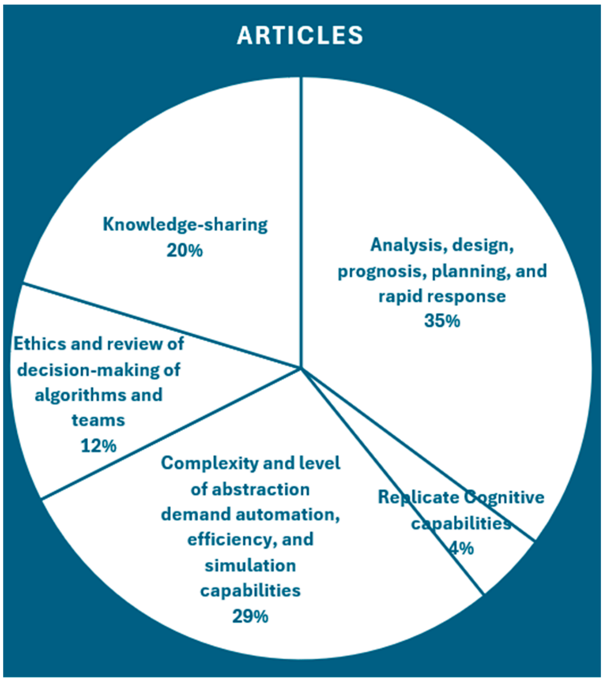

Figure 2. Identified areas for sustaining the usage of digital twins.

For our study, we consider the above five reasons an excellent fit to justify DTs within ALM. They all cover vital aspects of the three main ALM areas: governance, development, and operations. As DTs can mitigate risks and reduce costs—including cybersecurity critical issues—this also would impact the sustainability of the software industry.

DTs can cope with the claim pursued by digital transformation in which all software development and management processes should be digitized and automated. To demonstrate the possible connection, the popularity of the term "Digital Twin" in the last five years in a side-by-side comparison with the digital transformation concept using Google Trends was analyzed in Figure 3. Digital transformation enables the creation and use of DTs regarding advanced technology, skills, and culture. On the other hand, the DTs help to create a digital representation of the physical model to offer advanced capabilities. Consequently, it makes sense to see certain similarities in the evolution of both terms.

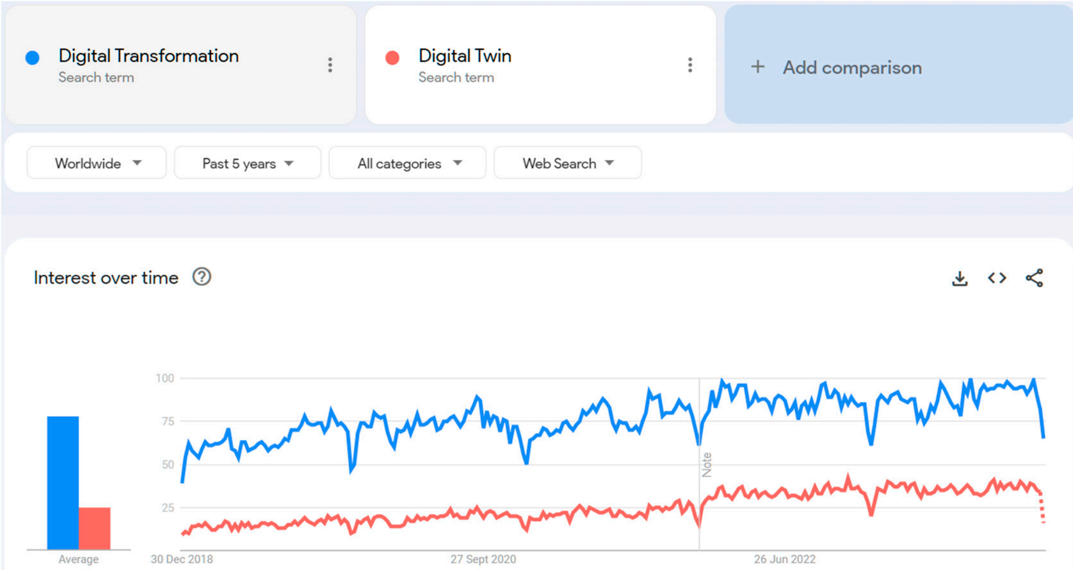

Figure 3. DT and digital transformation search terms and trends.

By default, it is impossible to carry out ALM using a single tool for project management. ALM covers the SDLC, and the potential ecosystem of toolsets with different vendors is enormous. Finding time to understand the correlation between the changes and outcomes throughout the whole ALM in real time and how to make the flow more efficient is somewhat problematic. The value stream mapping technique aims to explore the flow offline and identify where the waste is to optimize it. DT technology aims to create digital capabilities that synthesize the critical areas of the flow in real time to enable automatic optimizations.

We consider DTs a consequence of digital transformation to support a growing complexity value stream with a good balance of speed to release new features and deliver products to the market. As remarked, the smart manufacturing area is an excellent place to extract findings. From the overall picture of the works analyzed in this study, one of the critical factors to have a successful twin transition—as a world with a normalized usage of DTs—is related to the level of investment required to create virtual models of the existing systems, and the different possibilities that could bring. Understanding the cases and focusing on obtaining significant gains is crucial in this matter.

Chen et al. [41] comment on enabling twin transition to obtain a sustainable industry by pushing companies to innovate. Governments and administrations can support this push to obtain economic support. Recently, in Europe, The Digital Europe Program's campaign has been helping to boost the adoption of digitalization opportunities to provide financial support for digitizing medium and small-size companies' ecosystems, administrations, and citizens. Gartner stated that we are moving towards an API economy as an enabler for turning a business or organization into a platform.

In Spain, the Digital Kit campaign within the European Digitalization Program helped many organizations to enter into digital transformation programs. This trend must continue to enable Software Industry 4.0, focusing on a more sustainable software production in terms of energy and waste. DTs also offer a fantastic opportunity for funding organisms to have a simulation/virtual model of the funding outcomes and increase the audit capabilities available for multiple-year investments. A more assimilated usage of DTs could help administrations and organizations better control their ALM when multiple contractors and suppliers are involved.

After a cost analysis, West et al. [119] draw bold conclusions: the required investment may be high for the expected benefit. If the drawbacks from Table 2 are reviewed, several issues could require a significant investment. Nevertheless, it is imperative to find a sustainable approach for organizations. DTs in software engineering could require less investment than in manufacturing, as IoT physical sensors and actuators are not required because the assets are digital.

Another perspective is about using different technologies to optimize ALM as agent systems. Minerva et al. [127] establish similarities between multi-agent systems and DTs as they offer similar services. While DTs base their taxonomy of elements on the underlying concept of mirroring an SuS, the agents do not need to mirror the assets. They are built to accomplish an objective. There is a lack of real cases on the possibility of interaction between agents and DTs, which could help make DT collaboration easier through agent symbiosis. So, even when complementary, the goals are different, and DTs can offer learning from the past and present to create an optimized future.

As in any disruption, there is a need to determine the best implementation strategy. Roger's bell curve can be seen as the general innovation adoption lifecycle in which the laggards risk losing their customers' portfolios, as in Bohlen et al.'s study [208]. In contrast, Clayton Christensen coined the term disruptive technology in 1995. He recommended that large companies maintain minor, nimble divisions that attempt to replicate the usage of disruption internally, as in Bower et al.'s study [209]. Hence, DTs must offer a base to replicate, scale, and accept diverse situations and maturity to succeed.

In RQ4, the use cases are detailed. The focus is mainly on security, quality, and monitoring. These aspects can help ALM to gain efficiency.

4.3. RQ3: How Can a DT Be Built in the Context of the Software Industry?

There are attempts at standardization like ISO 23247 [210], mainly focused on manufacturing and other sectors, which include the IoT and M2M communication. Other remarkable attempts to standardize DTs are described by Autiosalo et al. [144]: (a) the Web of Things Description from the World Wide Web Consortium; (b) Digital Twin Definition Language by Microsoft; (c) Asset Administration Shell by Platform Industrie 4.0; (d) ETSI NGSI-LD; (e) PADI Connection Profiles; and (f) the Eclipse Ditto Platform. However, they are not specifically designed for creating DTs, and none of the mentioned ones has received widespread popularity to become DT standards.

There is no consensus about which architecture, framework, or methodology is best for generating or creating DTs. Minerva et al. [127] summarize DTs and some areas of interest with different architectures. Ferko et al. [200] analyze the existing architectures for DTs—using the ISO 25010 [211] standard on software product quality as guidance—resulting in 56.42% of the architectural solutions being reference models. The analyzed solutions use a catalog of 10 architectural patterns, with half of the solutions focusing on two patterns specifically: layered and SOA patterns. The preferred approach is using a layered approach over a service-oriented one. Boyes et al. [201] review architectures with different numbers of layers.

In many cases, a mechanism to generate DTs can be attractive to replace a digital or physical counterpart easily when a digital representation of the process, tools, and models is required during the system lifecycle in the context of PLM or ALM. In such cases, both systems, the DT and the counterpart, are executed simultaneously and should be managed and synchronized. Deuter et al. [6] theorize about the possibility of generating DTs for PLM/ALM using Open Services for Lifecycle Collaboration (OSLC) [212]. OSLC provides a standard interface for interoperability between different software tools and systems throughout product development and lifecycle management.

In contrast, Tekinerdogan et al. [205] center their analysis on specific architectural patterns to apply DTs for different lifecycle stages, such as the conception, development, production, utilization, support, and retirement stages. The catalog is composed of nine architectural patterns: (a) the digital model, digital generator, digital shadow for concept, development, and production stages; (b) the digital matching, digital restoration, digital monitoring, digital control, and digital autonomy for utilization and support stages; and, (c) the digital proxy for utilization, support, and retirement stages.

From an ALM perspective, three pillars define the infrastructure in the software industry: governance, development, and operations [112], as well as, more recently, security. In this context, DTs can be generated to have a digital replica of tools and processes currently managed in an organization. An infrastructure architecture based on the digital control pattern can assist the construction of DTs. From a connectivity perspective, DTs can cooperate in a system of systems (SoS) approach, as reflected in the architecture. This is similar to Autiosalo et al. [144].

In this way, the four critical capabilities for enabling DTs for manufacturing described by Guo [213] can be covered:

- Digital modeling: DTs must be capable of generating virtual models.
- Analytics support: DTs should provide services to understand anomalies more precisely and the relationships between the anomalies and the whole value chain.
- Timeliness update: DTs must be able to update the virtual models and data storage platform in near-real time, parallel to the asset system's operation.
- Control: DTs must supply the capacity to autonomously take action to control the assets based on conducted analyses from a process perspective and product operation.

The DT generated through this approach can cover general scenarios to gain efficiency and save costs for most organizations, such as the following:

- Test plan predictions and the impact of issues expected with the changes in a specific release cycle;
- Risk evaluation of issues in performance or by customers;
- Predictions on the time to complete a release cycle;
- An evaluation of the requirements and architecture changes involved;
- An evaluation of dependencies;
- An evaluation of budget to simulate budget consumption for project pipelines in the products roadmap;
- Anomaly detection with the flow of the release.

The exchange of information between a DT and its real counterpart must be decoupled to switch on the DT or the real counterpart indistinctly. Autiosalo et al. [144] propose the construction of DTs based on web technologies with a common skeleton and a way to cooperate. The inferencing skills of the DTs can improve with this cooperation among DTs. The information exchange between the DT and the real counterpart can be achieved based on OSLC (similar to AAS).

Moreover, the DT must manage the persistence and processing of data information extracted from the assets or counterparts using data repositories such as data lakes or graph databases. This ensures more capabilities for reasoning over the linked data, as with Bano et al. [167]. The data extraction and preparation can take significant time in the total implementation when the expected sync rate and the harmonization of information are also important topics. Message queue systems and data lake-creating channels to operate, as in [145,151,165], might be required.

Creating relatively rigid DTs—or DTs excessively bound to a specific vendor—would be problematic for their maintenance [134,135,144,145,148,151,153,161,165,167,202]. A total of 64% of the articles speak about accelerated DT generation using metamodel languages and tools to build DTs based on a reference architecture. This approach to generating DTs in a low-code or automatized way matches the context of constantly changing organizations and the DevOps culture flows, encouraging short feedback loops. Hence, changes are pretty frequent, and a way to validate the model is implicit in the metamodel restrictions. This can also help to quickly generate DTs based on a cockpit template for different purposes. Based on the analyzed articles, metamodels are an excellent approach. They must include information to cover the following:

- Specific notation for DT description: A level of abstraction about the definition, the status of the DT, the description, and operation (deployment, configuration, installation, and instantiation) (Autiosalo et al. [144], Muñoz et al. [151], Oakes et al. [139], Gennady et al. [128], and Bechu et al. [163]). Jones [116] characterizes the most relevant attributes of a DT. In contrast, Oakes et al. [139] also introduce a way to describe DTs with three layers and 14 characteristics and tested the approach in different scenarios. Some examples are as follows:
    - A micro-language called SMOL [161];
    - DT definition documents, such as DTDL, the Web of Things Description Language, or AAS (Autiosalo [144]);
    - MDD with Montigem with UML, as with Brockhoff [145];
    - UML + OCL, as in [151];
    - A metamodel with ADOxxx, as in [202].
- Domain context: It includes the information managed in the SuS and its environment as humans, or agents to consider. Kamburjan et al. [161] explore using knowledge graphs so they can be queried algorithmically. There are even projects to create knowledge graphs for the world avatar, as in Akroyd et al.'s study [214]. When the SuS is the developed software, Oakes et al. [139] comment on product architecture with DTs enabled by design, allowing products to add DTs seamlessly and offering patterns to include as part of the architectural drivers on the design of products. From an ALM perspective, some systems, such as application performance management (APM), can track the product's performance. However, data insights from the usage of the application will come from the software itself.

Apart from establishing the core of the data persistence and how to analyze it to achieve the established goals, there is a need to define the methodology to build DTs. Zhang et al. [135] include model engineering for DTs with different steps: requirement definition, model construction verification/validation and accreditation (VV&A), model application, model reuse, and model maintenance are their metrics to focus on building the right DT. Applying MBSE practices comes from the origin of DTs linked to PLM and knowledge-rich enterprises. Pileggi offers a different approach to mingling systems engineering and information technology called the double helix model in [134]. The article also argues that modeling is not well-suited to deal with the dynamics of DTs for software and merges MBSE (model-based system engineering) with IT DevOps, with three stages: design, deploy, and operate. In this context, it eliminates the need to track the changes over the product lifecycle for product data management. For deployment, Hughes et al. [130] speak about ModDevOps and TwinOps to standardize deployment and evolution.

Chaudhary et al. [203] use MDD (model-driven development) to synchronize different elements in the value creation chain. Frick et al. [165] introduce a specific framework for digital value-chain twins. A structured approach to building DT solutions depends on the domain, so the methodology and phases to be covered will be specific for this solution.

To fight uncertainty, one remarkable characteristic of the DT design is its capacity to distinguish different cycles and take the learnings from previous ones, following MAPE-K principles. This topic is touched on by Pileggi et al. [134], Rivera et al. [160], and Engels [149].

Another capacity is mining process data to incorporate the information into the DT, as analyzed by Bano et al. [167]. In our approach, the learnings are ingested via collaboration from the multi-agent system and knowledge graph. Agents can develop their knowledge during their lifecycle.

Once a DT automates a decision (for instance, assuming a digital automation pattern, as with Tekinerdogan et al. [205]), assigning responsibilities or perceiving bias may be challenging. The more autonomy is given to DTs, the more significant the concern about the audit type of decisions they can take. A coding bias could be problematic and provoke failures in the decisions. Some articles focused on validating the level of trust in their results are [120,122,131,143,171,173]. On the other hand, DTs can be used to validate the decisions made by decision-making algorithms. Vyhmeister et al. [157] discuss ethical by-design principles and share the ideas in a framework.

Several types of technical debt in DTs are explored by Malakuti [147]. This is important to understand, since the cost of their implementation and maintenance could be impacted by how fast the technical debt is accumulated.

Assuming a digital control pattern, it is possible to consider a non-fully automated solution that can iterate towards more autonomy after the validation of the results towards maturity metrics. Hu et al. [171] establish a maturity model based on DT value, function, and reliability dimensions. Zhang et al. [135] also cover metrics for every step of the defined build method with specific dimensions. As stated, the DevOps approach of Pileggi et al. [134] seems to be more aligned with software engineering. Consequently, a general maturity model could offer a view of the overall situation of implemented DTs.

To enable this approach, the layered architecture contains entry points for users to interact with the DT. The data from lakehouses and the knowledge graph can be retro-propagated, so there are ways to enable more analysis with diverse types of information. This is useful to set up models and adapt from every iteration. Specific roles like data scientists and AI specialists can interact with DTs to extend the capacity of the models and the DT capabilities.

Regarding security and privacy, all the information related to DTs needs to follow the same risk management principles as the original asset regarding regulatory compliance. Hence, encryption, privacy by design, strong APIs, and credentials are required by Autiosalo [144]. Using a lakehouse with a semantic layer adds security capabilities to the system.

Once the proposed reference architecture and the storage and analytic dimension of the architecture tackle different aspects of the solution, namely scalability, connectivity, continuous improvement, or security, there is a need to define a conceptual framework to describe the potential dynamics of this architecture. Therefore, the definition of an ontology to be used as the conceptual framework of the system to guide integrations of SuS data into services and their partial code generation is critical due to the acceleration and flexibility to cover different scenarios.

4.4. RQ4: What Are the Uses of DTs That Can Be Applied to the Software Engineering Area, Specifically ALM?

Transforming businesses or organizations into optimization platforms is an ideal context for using DTs. Of the publications reviewed, 40% cover aspects of DT generation due to the urgency of having a robust, general-purpose approach. As in [197], there is a need to differentiate the problem domain (needs, problems, constraints) from the solution domain (concepts, base technologies, system solutions).

From their survey results, the 16th Annual State of Agile Report [215] stated that four out of five respondents use agile, and half use a combination of agile, waterfall, or interactive. On top of that, DevOps practices are achieving 83% of the aim of organizations for combining software development with operations, including delivery. In this context, we also have adaptations such as agile ALM as a good reference for agilism, as pointed out by Huettermann [216]. Then, ALM is suitable for traditional and agile methodologies.

Furthermore, ALM encourages the seamless integration of tools without barriers. Our constraints in the software industry are linked to the SDLC and ALM's incredible diversity of ecosystems.

Regarding ALM in terms of maintenance and innovation areas, the survey conducted by the National CIO Review publication among their readers in 2023 concluded that 30% of companies focused on maintenance, another 30% included some innovation when there was a clear business need, and 25% bet on a balanced approach when allocating resources. In contrast, 15% were innovation-first companies with punctual maintenance. According to this survey, most of the resources are invested in maintenance. In most cases, the existing products provide a stable foundation for a company; consequently, optimizing their value streams is essential for most organizations.

As shown in RQ3, there is no general agreement on how to design DTs. Specific use cases of DTs in the software lifecycle are frequently seen to evidence their potential value and applicability. These cases are shown in Table 5.

Table 5. Specific use cases for DTs in software engineering.

Usage	Title	Reference
Insider Thread DT	Creating a DT of an Insider Threat Detection Enterprise Using Model-Based Systems Engineering	[87]
Ethical DT	Responsible-AI-by-Design: A Pattern Collection for Designing Responsible AI Systems	[173]
DT State of Quality in Agile Process	Concept of Quality DT in Agile Development	[162]
Architecture maintenance by DTs	Co-evolving digital architecture twins	[146]
Cockburn procedure for soft dev: User Stories > Scenarios	Implementing DTs in existing infrastructures	[174]
Continuous Monitoring of the Value Stream	The Digital Value Stream Twin	[165]
Chaos Twins to create anomalies.	Chaos Twin: A Chaos Engineering and DT Approach for the Design of Resilient IT Services	[150]
PADTCs	Process-aware DT cockpit synthesis from event logs	[167]
Architecture Maturity Evaluation and Improvement	Continuous engineering for Industry 4.0 architectures and systems	[114]
Cybersecurity	DT for Cybersecurity Incident Prediction: A Multivocal Literature Review	[129]
Version Control DTs	Integrated Version Control of physical and virtual artifacts	[116]
SOC/Compliance	Towards Process-Oriented IIoT Security Management: Perspective and Challenges	[217]
DTs for TDD, test cases' combinatory algebra	ART for Agile: Autonomous Real-Time Testing in the Product Development Cycle	[152]

According to Table 5, few cases are identified in the literature specifically from a software engineering perspective. ALM is a relatively new concept, similarities may be appreciated with the purpose of this document taking a look to Frick et al. [165]. In Figure 4, a list of researched usages of DTs mapped in ALM areas is shown, along with the studies in

Table 5. This evidences the current applicability state through study analysis that fits the ALM process.

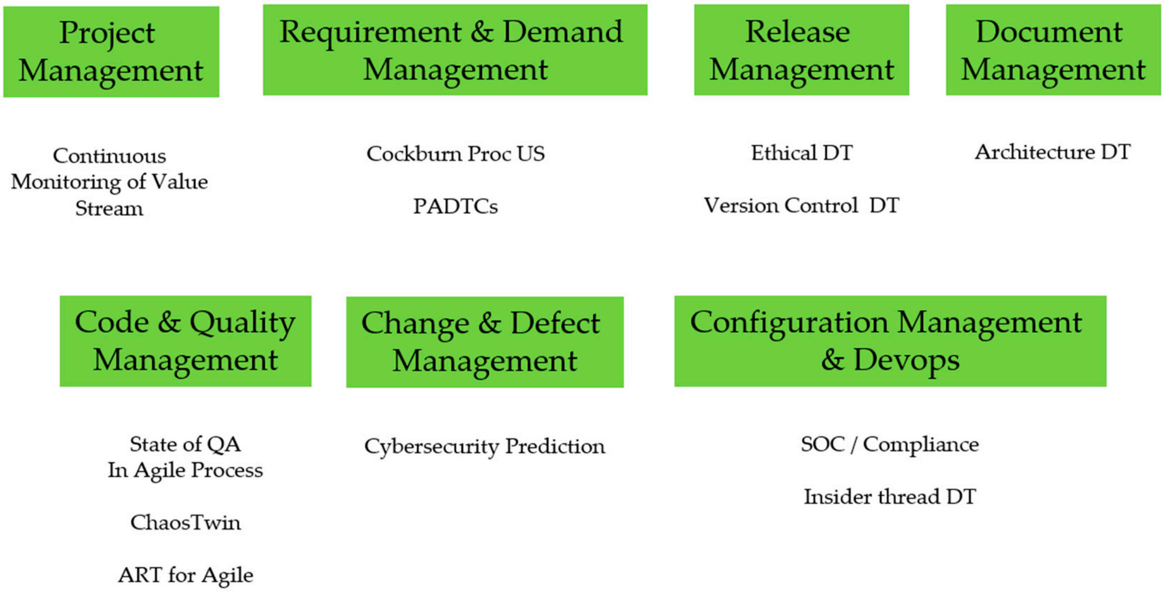

Figure 4. Use cases detected from articles mapped into ALM areas.

Among the exposed cases, some DTs could combine efforts to cover more ground in ALM. This combination of multiple DTs could be a roadmap for organizations starting their journey with DTs as part of their digital transformation process. The evaluation and prioritization of this roadmap is a critical activity to understand the most beneficial path in terms of ROI. Every organization could have different needs based on its maturity and context.

5. Discussion

DTs can help drive business into the digital thread. According to 74% of engineering, manufacturing, service, and IT department leaders surveyed in PTC's recent State of Digital Thread, improving a company's ability to leverage data across the enterprise would be effective or highly effective at addressing disruption.

There are many similarities between ALM and PLM value streams. Tons of findings can be taken from manufacturing and put into the software industry. Nevertheless, practices for PLM and ALM are different, and every domain has its own needs and constraints, even in different markets or verticals. Setting a reference architecture and a methodology to build DTs efficiently is a great goal to help organizations climb up the digital thread faster.

Although this fact is quite unifying and offers an excellent starting point, the context of each organization is unique. How can the suitability of DTs for the software industry in the context of ALM be evaluated? Next, several relevant topics will be highlighted.

(a) Digital transformation

Due to digital transformation processes, DTs are hard to apply in non-digitalized SuSs. The level of adoption of digital transformation is crucial to enable DT usage. If the digitalization of the processes is very basic, a vast amount of information is neglected, not updated, or kept in siloed data sources that are not well interconnected.

COVID pushed companies to adopt digital transformation practices and provoked a significant investment to grow resilience in a world where companies needed to be able to market from home rather than offices. Tools, practices, and how information was used needed a review to adapt to the uncertainty and flexibility of new ways of working at a massive scale.

As shown in Figure 3, there is a relationship between digital transformation and the activation of DTs. Cloud adoption also contributed to handling the data explosion that simulations of DTs work with. A solid digital transformation program and strategy implemented relying on strong cybersecurity principles is a critical dependency.

(b) Maturity

Even though the Agile Manifesto states individuals and interactions over processes and tools as one of the axioms, gaining efficiency means including different forms of formalization through these interactions. This formalization can be a significant advantage in integrating DTs. Suppose that a DT needs conversion from real-world data into semantic data connected in the SoS DT to analyze simulations. In that case, the capacity to have this information digitally available and with standardization is genuinely relevant.

DT implementations will need to have process mining-specific services. In case the information is digitalized but not formalized, process mining can be less meaningful and make extracting insights impossible at a profound scale in the value streams. Companies with solid maturity levels can also have a significant capacity to automate their processes. This is a clue to identify the applicability of DTs at a more beneficial level.

On the other hand, in case there is a deficiency in this level of maturity, and as evidenced in the systematic literature review of this work, DTs can also help to reinforce maturity, making compliance with processes more efficient. Less-mature companies must add more steps to their programs to accelerate innovation and response to current challenges.

(c) Cultural Aspects

There is a level of friction between teams and departments to be scrutinized in software engineering. From an ethical perspective, the goal of DTs must be transparent and support the growth of the organization and all people involved. As some articles work on tracking emotion to identify fatigue, DTs could also be used to simulate processes and identify potential risks for companies such as potential leavers, low-performers, and other sensible cases. This type of analysis requires a solid approach for the confidentiality of the data results and the evidence, but also a way to guarantee that the decisions or actions extracted from DTs are trackable and solid.

However, agilism has shown that not all metrics and results from them make a difference between successful companies and laggards. Consequently, there is friction from the development team with being monitored or compared with other teams or their value in the process.

To be successful, DTs must be people-centered, -designed, and -implemented. This implies working on objectives relevant to people at scale.

(d) Technological Ecosystem

The technological choices selected by organizations are pretty unique. They depend on internal evaluations, context, management, and leadership. Those factors vary from year to year as there is no decision on a technological choice selected in aeternum.

DTs need to build an abstraction level able to extract the system under study information with the independence of the technology but focused on the purpose, function, and potential behaviors of the SuS. Selecting tools linked to open standards, such as OSLC, helps to significantly reduce the automation and DT adoption cost in evaluating tools to integrate into the ALM. It is imperative to understand the compatibility and interoperability. This can be reached by using specifications like OSLC or exposing APIs.

The update cycle for vendors about the software integrated into companies' ALMs is also essential, since backward compatibility can help to maintain the consistency of all information ingested in the DTs or the automation implemented on top of these APIs.

(e) Skills and Investment

The scarce profiles for implementing complex DT aspects in an organization could be a setback and a concern to slow down the adoption of and investment in this technology. This is one of the reasons why a wide variety of articles use MDD to help reduce friction, simplifying the adherence to defining and changing DTs. As seen in the previous paragraphs, building DTs means integrating and ingesting a wide range of data sources. The data quality and level of transformation can make the investment level significantly grow.

DTs in ALMs for the software industry are not very common. Finding profiles and engineers that are able to carry out this task can imply having researchers and data engineers on the team, but also crucial stakeholders in the company to identify, track, and interpret critical sources of information.

From the factors commented, it is clear there is a risk of exclusion for those organizations that are not able to leverage their strategies towards a successful digital transformation and, consequently, enable advanced engineering practices like DT adoption. This could impact their capacity to compete against organizations in the long run.

The good news is that DTs could benefit by increasing maturity and collaborating more efficiently with third parties.

In summary, DTs are desirable for companies moving forward with their digitalization. The applicability and number of use cases depend on different factors, as seen in this section. Nowadays, generative AI is considered a disruptive technology; consequently, a holistic approach toward the digital thread is strategic for companies' survival.

6. Digital Twin Development Guidelines for the Software Industry

This section aims to establish some guidelines on how to face the development and management of DTs when applied to ALM. It contains some essential suggestions about the foundations of the DT architecture, the key elements to consider in ALM and the SDLC, and several issues to consider in their design, deployment, and execution.

6.1. DT Architecture

DTs have become essential components in the digital transformation of the software industry in many stages of the lifecycle. Therefore, the development and execution of DTs that are complementary to the tools managed in the software process could be essential to standardize. A reference architecture can be set as the basis for constructing a new DT.

The reference architecture of a DT based on a layered architecture approach aligns well with the research findings. The proposed architecture takes ISO 23247-2:2021 [210] as the reference (Figure 5) from the available options as an inspiration for the targeted layered reference architecture. The proposed architecture adheres to a reference architecture with five different domains, as commented on by Duan et al. [35]: a user domain, a DT domain with three subdomains (ops and management, application and services, and interoperability sensing and controlling), a physical domain, and a cross-domain.

This assumption has benefits: a solid standard usage, integration between data and systems with clear responsibilities, privacy and cybersecurity, quality and reliability, a trusted approach for collaboration and communication, and finally, scalability. However, as ISO 23247 [210] is mainly thought to be for manufacturing environments and physical systems, there is a need for an adaptation to operate in the ALM context.

The user entity part corresponds to the purpose of the DT. Dalibor et al. [7] extract that the main usages of DTs are in monitoring, behavior optimization, and prediction, aligned with what is shown in Table 5. In the ALM context, optimizing the workflow to become more efficient is essential. This goal is taken as the main driver. There is a need to consider that maintenance is much more frequent than innovation. This might have less singular information for simulation and learning, making it more relevant for DT simulations in the development and operation steps.

The core entity part is the heart of the DT reference architecture, which is used for creating the information and simulations and the capacity to interact with other entities. This area will work with all semantic data extracted from the SuS in federated models in which all concepts are linked and can be queried.

The concept of observable SuS can replace the observable manufacturing elements. The OSLC connector layer can replace all the areas of the data collection entity to enable the capacity to extract information in real time about the state of the SuS, and the device control part can be combined with SuS-specific APIs.

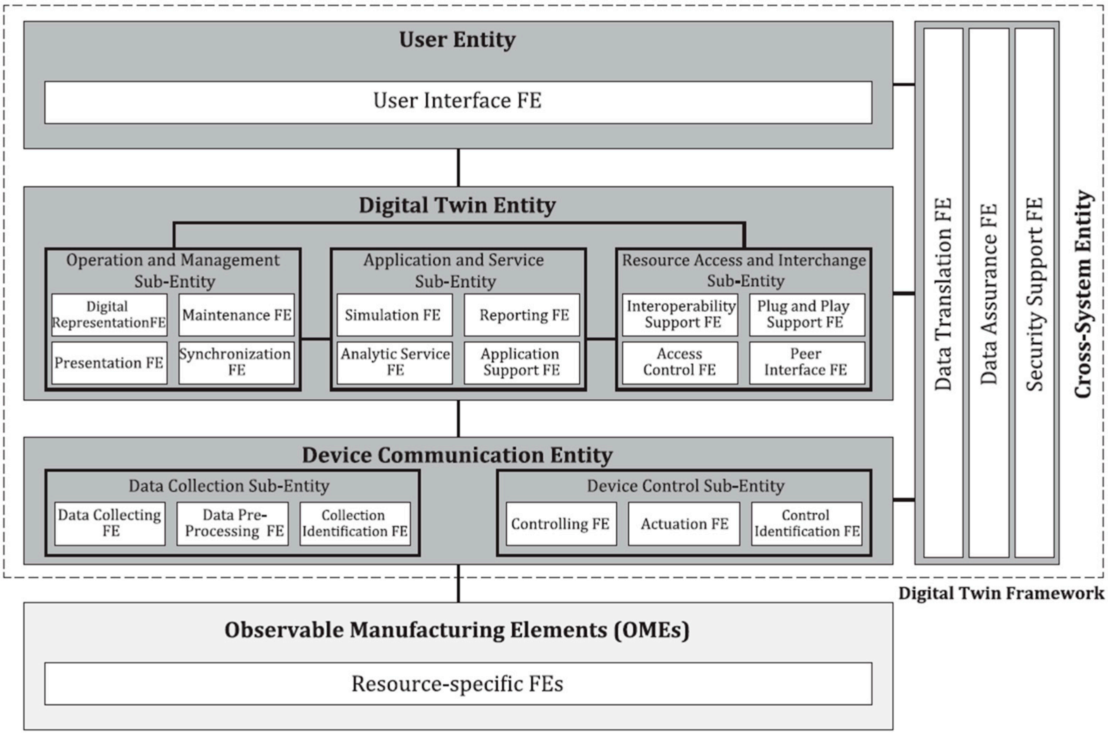

Figure 5. ISO 23247-2:2021 reference architecture. Reprinted with permission from [210]. 2021, ISO.

Zhang et al. [135] establish information syncing as one of the critical challenges in DT engineering. Even for an event-driven architecture, time harmonization could be essential. Regarding ALM, relative time must be meaningful to the dimensions driven by the ALM cycle: phases, releases, and hotfixes can help link the data with a context for reasoning and bring new information.

To use the approach introduced to solve the identified problem, it is vital to establish the base mechanisms for persistence and work on the data by combining all potential diverse sources. Our proposal from a specific framework consists of using three information tools:

- Knowledge graphs: OSLC is based on RDF; the usage of a metamodel built on top of the OSLC APIs and connected using a graph DB will enable semantic queries, reasoning, and other techniques applied to the connected model of the current elements in the SuS to mine new knowledge. From a primary usage for semantic search, this approach allows the application of reasoning rules, traversing the connected data, or even machine learning. Knowledge graphs represent a model of reality as a powerful tool to explore the space of complex problems, such as with Akroyd et al. [214]. The knowledge graph combines the information extracted and mapped into OSLC specification from the diverse sources of information. Hence, the data capabilities of our approach adopt a semantic layer to democratize the data usage via the DT.
- Lakehouses: This element aims to provide scalable data processing capabilities to the DT. The lakehouse keeps the door open to apply transformations before ingesting into a query layer, the knowledge graph. The transformations will tackle some adjustments into the different SuS to unify aspects of the interpretation and for the normalization of the data. Lakehouses also open the possibility of having dissimilar sources of information from OSLC so the transformations can be applied directly to transform the original data into RDF before ingesting into the semantic layer. This expands the

capacity of the architecture to tackle dissimilar sources of information. Furthermore, using a lakehouse reduces cost and focuses on scalability to work with the data.
- Multi-agent systems: Multi-agent systems will enable a solid simulation tool, taking advantage of the power of multi-agent systems and the knowledge to learn from the knowledge graph. This approach differs from statistical methods like Montecarlo. This is seen with Latsou et al. [17], for instance. Also, Gorodetsky et al. [177] comment on the suitability of MASs to complex problems due to their adaptive skills and foresees an opportunity in combination with DTs. Adding MASs, which can be distributed by extending the layered architecture, is an opportunity to decouple and give the different simulated contexts more possibilities.

This section will cover some issues while defining DTs for ALM in the software industry. Going into detail about the final decisions and technological choices is out of the scope of this review.

6.2. Mutable Credentials and Identities

An essential part of ALM involves people: roles, teams, departments, and organizations. In terms of identity management, there are several topics to consider:

- All of these elements change over time. An entity change provoked by a human or a service can vary, and the role of the person or service can be relevant enough.
- On top of that, from the different cyber entities such as version control systems, ERP, and others, a user can have different credentials, but, in the end, they are the same person. This can be a severe obstacle to making a virtual model that mimics reality. Mapping credentials into a unified and federated view of everyone is crucial.
- Preserving privacy and confidentiality: when there is sensitive information in the systems to be mirrored, different techniques need to be applied so that the information is only available for its purpose, and the design makes it impossible to obtain relevant information about someone: opinions, feedback, or performance.
- Handling several types of accounts: service accounts, emergency accounts, application accounts, and user accounts changing over time could be a potential issue.

It is not strange to see some researchers using blockchain frameworks for DTs, like Hunhevicz et al. [204], Borowski [218], or Ahmad et al. [219]. The last one uses a blockchain for audit logs.

With a lakehouse implementation in the DT architecture, transformations can be set before building the semantic layer to unify and resolve the multiple credential issues.

6.3. ALM Systems under Study Consideration

Based on the fact that 47% of agile teams are measured based on on-time delivery and another 44% based on business objectives achieved, according to the 16th Annual State of Agile Report, prioritizing scaling organizations towards high performance is the focus of research after this document.

There are many cases in the software industry of smart elements developed by vendors. This trend is expected to grow with generative AI and ML usage. Nevertheless, DTs need to be differentiated from smart assets.

Smart assets do not help with understanding and improving the unique ecosystem in every organization from a holistic perspective, and they do not use dark data and siloed information, since the culture, processes, and technological choices could be vastly different.

At this stage, and to use as a base, it is vital to identify the critical elements of ALM that can contribute to improving an organization's results. There is exciting research like that of Forsgren et al. [220] identifying key capabilities for software delivery.

Forsgren identifies five categories with 24 capabilities to be enabled for organizations' acceleration. These categories are as follows:

- Continuous delivery groups the capabilities related to empowering delivery in the organizations;

- Architecture: its evolution by teams in a decoupled and proper way unlocks the potential for seamless operations for the catalog of services offered and maintained;
- Product and process: how the flow of product evolution and customer feedback is managed in the value stream is at the heart of an organization;
- Lean management and monitoring: the capacity to eliminate the waste of the value chain strongly aligns with Kaizen's and Deming's methods;
- Cultural factors: the values of organizations and teams' cooperation are the first pieces to enable the rest.

The capacity of the proposed architecture to include this information in the system relies on the ability to have OSLC connectors that are linked to assets or at least be able to map into OSLC specifications for storage. The evidence of this diverse classification by Forsgren makes clear the need for tools to manage the complexity of analyzing and extracting insights from all the assets involved.

6.4. Ontology as a Conceptual Framework

Defining the ontology to be used as a conceptual framework involves several steps.

- Identify the domain: from ISO 23247 [210], the layer and domains can be extracted.
- Gather knowledge: these are related to all the sources, either in OSLC or in different formats, through lakehouse data ingestion and understanding the relationships and constraints among them.
- Conceptual model: Since ISO 23247 [210] is very stuck to manufacturing, OSLC specifications are the ones to help here. For every identified SuS, the OSLC API and mappings to be used will be determined, and the scenarios described for the essentials of entities, attributes, relationships, and constraints will be identified and used as the foundation of the ontology.
- Definitions: with the conceptual model as a base, the extension towards all the aspects, including classes, properties, taxonomies, constraints, and rules, must be detailed and translated into a language.
- Refinement: the evolution of APIs (OSLC) or understanding the semantics defined in the ontology will trigger the iteration through the stages to redefinition the concepts presented.

As commented, an ontology for the domain context and the DT definition (instantiation, deployment, configuration, maturity) is desirable.

In the first case, the ontology defines the potential dynamics of the simulations and analysis features from the system, so this is an active element that will drive the instantiation in operation, help to validate it, support the definition, and control the constraints. Meanwhile, the DT definition is a passive element that will be part of the system's configuration and support the DT system setup.

7. Conclusions

A systematic literature review was conducted to discover how DTs have been applied to the software industry, especially in current software infrastructures and organizations in the context of the ALM. DTs are one of the key pieces of innovation in the digital transformation that is currently taking place to achieve the much-desired Industrial Revolution 4.0, which will make possible the optimization of systems, the prevention of inefficiencies, the improvement of productivity, and the reduction in costs.

The status of the adoption of DTs has been growing continuously in recent years in specific areas such as manufacturing and in other industrial sectors, but not so much in the software industry. However, an incipient number of works from manufacturing will establish a solid base for their application to the software industry, especially at the first level in the software development process and, to a lesser degree, at the second level in application lifecycle management.

The traditional methods, methodology, and tools used by organizations for developing software have changed significantly towards a software infrastructure that manages the entire process from inception to the end of the application's life, including the governance, development, operations, quality, security, and business in the context of ALM. In this context, digital twins are the tools required to face the complexity of the application lifecycle path, even making the automation of the entire process possible.

DTs are the enablers for the next steps of the software industry's digital transformation. The level of maturity, the internal culture, and the level of investment are crucial factors to allow organizations to avoid being laggards on the digital divide, but are also a remarkable opportunity for auditing purposes, as DTs could act as independent evaluators of the status of the processes. The benefits of automation and a reduction in efforts in the value stream of the products can cover the ROI of their implementation.

This work describes in detail the significance of digital twins in the software industry, the specific domains and areas where they can be applied in the software lifecycle, and the proposed approaches explored to build powerful DTs to cover ALM for developing, deploying, and maintaining software systems, taking into account the papers analyzed in this work.

There are still some restrictions to be overcome for DTs to be exploited in the software development domain, such as the lack of standard methodologies, tools, and technologies, and more approaches to create and generate DTs with adequate costs for software development, but also other processes, tools, and methods of its lifecycle. However, on the contrary, this work presents an updated status on the adoption of DTs in the software industry.

Finally, we have included a list of guidelines for developing DTs in the software industry that provides ideas on how the reader or community can progress their research using DTs.

8. Future Steps

Nowadays, an effort to define a solid DT to sustain the ALM process in the software industry is still in progress. There is a strong parallelism between PLM and smart manufacturing that can benefit massively from this research, as evidenced during the review. The concept of DTs starting from a physical asset is not working for ALM, since many artifacts are digital. Consequently, a specific definition and a way to model them are required. ISO 23247 [210] is used as a starting point. The persistence of the exchanged data and the generation of services are achieved using data lakehouses and the semantic layer based on a knowledge graph as a central paradigm from the architecture perspective. This helps to cover critical aspects such as security, scalability, inferencing and mining information, breaking silos in information, and democratizing usage. A semantic web technology is used for the architecture's dynamics, definition, and service description. So, an ontology as a conceptual framework and defining DTs to enable partial code generation and automation is also in progress.

Using multi-agent systems to replicate the dynamics of ALM through a DT is the basis of our proposal. The agents operate as part of the simulation in the DT. Thereby, an implementation initiative for this approach and the validation cases are in progress. The results will be shared in future articles.

Companies are also experiencing the disruption of AI, more recently through the usage of large generative AI models. The DT can be an excellent enabler to have a mirrored view of the reality of the software companies' value streams with a consolidated view of the data, relationships, and dynamics of the processes. Generative AI can help to enrich this view with different scenarios and test agents in different contexts. On top of that, the interaction experience between agents and the users of DTs can be more natural with the usage of generative AI.

Concretely, implementing large AI models into DT capabilities is an option to explore in our base architecture. Despite the fact of the challenges associated with its usage, there may be two perspectives to enable generative AI within the proposed model:

- Through MOS, including generative AI as part of the agents. This could make the capabilities of every interacting agent more sophisticated.
- At the simulation level, as generative AI can elaborate different sets of data and scenarios, it could support a better understanding of the model's situation and the transitions and the testing of the agents in a broader range of situations.

The following steps explore this architecture's possibilities and applicability to real use cases, connecting it with other disruptive technologies. This approach interprets the next steps and evolution of digital transformation as a combination of the technologies enabled. DTs are an entry point not only for AI applications, but also to make the processes and their status and trends more visible.

Author Contributions: Conceptualization, M.A.G.-C. and J.A.H.-T.; methodology, M.A.G.-C. and J.A.H.-T.; software, M.A.G.-C. and J.A.H.-T.; validation, M.A.G.-C. and J.A.H.-T.; formal analysis, M.A.G.-C. and J.A.H.-T.; investigation, M.A.G.-C. and J.A.H.-T.; resources, M.A.G.-C. and J.A.H.-T.; data curation, M.A.G.-C. and J.A.H.-T.; writing—original draft preparation, M.A.G.-C.; writing—review and editing, M.A.G.-C. and J.A.H.-T.; visualization, M.A.G.-C. and J.A.H.-T.; supervision, J.A.H.-T.; project administration, J.A.H.-T.; funding acquisition, J.A.H.-T. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the University of Granada.

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

References

1. Schwab, K. *The Fourth Industrial Revolution. World Economic Forum*, 1st ed.; Penguin Books: London, UK, 2017.
2. Tao, F.; Qi, Q.; Wang, L.; Nee, A. Digital Twins and Cyber–Physical Systems toward Smart Manufacturing and Industry 4.0: Correlation and Comparison. *Engineering* **2019**, *5*, 653–661. [CrossRef]
3. Khan, S.; Arslan, T.; Ratnarajah, T. Digital Twin Perspective of Fourth Industrial and Healthcare Revolution. *IEEE Access* **2022**, *10*, 25732–25754. [CrossRef]
4. Grieves, M.; Vickers, J. Origins of the Digital Twin Concept; Working Paper. 2016. Available online: https://www.researchgate.net/publication/307509727_Origins_of_the_Digital_Twin_Concept (accessed on 1 December 2023).
5. Deuter, A.; Rizzo, S. A Critical View on PLM/ALM Convergence in Practice and Research. *Procedia Technol.* **2016**, *26*, 405–412. [CrossRef]
6. Deuter, A.; Imort, S. PLM/ALM Integration with The Asset Administration Shell. *Procedia Manuf.* **2020**, *52*, 234–240. [CrossRef]
7. Dalibor, M.; Jansen, N.; Rumpe, B.; Schmalzing, D.; Wachtmeister, L.; Wimmer, M.; Wortmann, A. A Cross-Domain Systematic Mapping Study on Software Engineering for Digital Twins. *J. Syst. Softw.* **2022**, *193*, 111361. [CrossRef]
8. Kritzinger, W.; Kerner, M.; Traar, G.; Henjes, J.; Sihn, W. Digital Twin in manufacturing: A categorical literature review and classification. *IFAC-PapersOnLine* **2018**, *51*, 1016–1022. [CrossRef]
9. Korenhof, P.; Blok, V.; Kloppenburg, S. Steering Representations-Towards a Critical Understanding of Digital Twins. *Philos. Technol.* **2021**, *34*, 1751–1773. [CrossRef]
10. Sadiku, M.N.O.; Yonghui, W.; Suxia, C.; Sarhan, M.M. Cyber-Physical Systems: A Literature Review. *Eur. Sci. J.* **2017**, *13*, 52–58. [CrossRef]
11. Dafflon, B.; Moalla, N.; Ouzrout, Y. The challenges, approaches, and used techniques of CPS for manufacturing in Industry 4.0: A literature review. *Int. J. Adv. Manuf. Technol.* **2021**, *113*, 2395–2412. [CrossRef]
12. Fraga-Lamas, P.; Barros, D.; Lopez, S.; Fernández-Caramés, T. Mist and Edge Computing Cyber-Physical Human-Centered Systems for Industry 5.0: A Cost-Effective IoT Thermal Imaging Safety System. *Sensors* **2022**, *22*, 8500. [CrossRef]
13. Maruyama, T.; Ueshiba, T.; Tada, M.; Toda, H.; Endo, Y.; Domae, Y.; Nakabo, Y.; Mori, T.; Suita, K. Digital Twin-Driven Human Robot Collaboration Using a Digital Human. *Sensors* **2021**, *21*, 8266. [CrossRef] [PubMed]
14. Somers, R.; Douthwaite, J.; Wagg, D.; Walkinshaw, N.; Hierons, R.M. Digital-twin-based testing for cyber–physical systems: A systematic literature review. *Inf. Softw. Technol.* **2023**, *156*, 107145. [CrossRef]
15. Semeraro, C.; Lezoche, M.; Panetto, H.; Dassisti, M. Digital twin paradigm: A systematic literature review. *Comput. Ind.* **2021**, *130*, 103469. [CrossRef]
16. Calvo-Bascones, P.; Voisin, A.; Do, P.; Sanz-Bobi, M. A collaborative network of digital twins for anomaly detection applications of complex systems. *Snitch Digital Twin concept. Comput. Ind.* **2023**, *144*, 103767.
17. Latsou, C.; Farsi, M.; Erkoyuncu, J. Digital twin-enabled automated anomaly detection and bottleneck identification in complex manufacturing systems using a multi-agent approach. *J. Manuf. Syst.* **2023**, *67*, 242–264. [CrossRef]
18. Dietz, M.; Pernul, G. Digital Twin: Empowering Enterprises Towards a System-of-Systems Approach. *Bus. Inf. Syst. Eng.* **2020**, *62*, 179–184. [CrossRef]

19. Zheng, X.; Psarommatis, F.; Petrali, P.; Turrin, C.; Jinzhi, L.; Kiritsis, D.A. Quality-Oriented Digital Twin Modelling Method for Manufacturing Processes Based on A Multi-Agent Architecture. *Procedia Manuf.* **2021**, *51*, 309–315. [CrossRef]
20. Nielsen, C.P.; Yu, F. Product Design for Matrix-Structured Manufacturing Systems. *Procedia CIRP* **2022**, *109*, 407–412. [CrossRef]
21. Cimino, A.; Gnoni, M.; Longo, F.; La Rosa, A. Digital Twin (DT) based methodology to support effective design of industrial production lines. *Procedia Comput. Sci.* **2023**, *217*, 1896–1907. [CrossRef]
22. Chiurco, A.; Elbasheer, M.; Longo, F.; Nicoletti, L.; Solina, V. Data Modeling and ML. Practice for Enabling Intelligent Digital Twins in Adaptive Production Planning and Control. *Procedia Comput. Sci.* **2023**, *217*, 1908–1917. [CrossRef]
23. Negri, E.; Cattaneo, L.; Pandhare, V.; Macchi, M.; Lee, J. Integrating PHM into production scheduling through a Digital Twin-based framework. *IFAC-PapersOnLine* **2022**, *55*, 31–36. [CrossRef]
24. Neto, A.A.; Carrijo, B.S.; Romanzini Brock, J.G.; Deschamps, F.; de Lima, P.E. Digital twin-driven decision support system for opportunistic preventive maintenance scheduling in manufacturing. *Procedia Manuf.* **2021**, *55*, 439–446. [CrossRef]
25. Villalonga, A.; Negri, E.; Biscardo, G.; Castano, F.; Haber, R.E.; Fumagalli, L.; Macchi, M. A decision-making framework for dynamic scheduling of cyber-physical production systems based on digital twins. *Annu. Rev. Control* **2021**, *51*, 357–373. [CrossRef]
26. Laukotka, F.; Rennpferdt, C.; Krause, D. Digital Twins and Product-Service Systems: A Synergy with Challenges and Opportunities. *Proc. Des. Soc.* **2022**, *2*, 1639–1648. [CrossRef]
27. Wilking, F.; Schleich, B.; Wartzack, S. Digital Twins—Definitions, classes and business scenarios for different industry sectors. *Proc. Des. Soc.* **2021**, *1*, 1293–1302. [CrossRef]
28. Rasor, R.; Göllner, D.; Bernijazov, R.; Kaiser, L.; Dumitrescu, R. Towards collaborative life cycle specification of digital twins in manufacturing value chains. *Procedia CIRP* **2021**, *98*, 229–234. [CrossRef]
29. Moder, P.; Ehm, H.; Ramzy, N. Digital Twin for Plan and Make Using Semantic Web Technologies–Extending the JESSI/SEMATECH MIMAC Standard to the Digital Reference. In *Digital Transformation in Semiconductor Manufacturing. EADTC EADTC 2018 2019*; Keil, S., Lasch, R., Lindner, F., Lohmer, J., Eds.; Lecture Notes in Electrical Engineering; Springer: Cham, Switzerland, 2020; Volume 670.
30. Papacharalampopoulos, A.; Michail, C.; Stavropoulos, P. Manufacturing resilience and agility through processes digital twin: Design and testing applied in the LPBF case. *Procedia CIRP* **2021**, *103*, 164–169. [CrossRef]
31. Empl, P.; Schlette, D.; Zupfer, D.; Pernul, G. SOAR4IoT: Securing IoT Assets with Digital Twins. In *Proceedings of the 17th International Conference on Availability, Reliability and Security (ARES '22), Vienna, Austria, 23–26 August 2022*; Association for Computing Machinery: New York, NY, USA, 2022; Article 4; pp. 1–10.
32. Umeda, Y.; Ota, J.; Shirafuji, S.; Kojima, F.; Saito, M.; Matsuzawa, H.; Sukekawa, T. Exercise of digital kaizen activities based on 'digital triplet' concept. *Procedia Manuf.* **2020**, *45*, 325–330. [CrossRef]
33. Ferriol-Galmés, M.; Suárez-Varela, J.; Paillissé, J.; Shi, X.; Xiao, S.; Cheng, X.; Barlet-Ros, P.; Cabellos-Aparicio, A. Building a Digital Twin for network optimization using Graph Neural Networks. *Comput. Netw.* **2022**, *217*, 109329. [CrossRef]
34. Petri, I.; Rezgui, Y.; Ghoroghi, A.; Alzahrani, A. Digital twins for performance management in the built environment. *J. Ind. Inf. Integr.* **2023**, *33*, 100445. [CrossRef]
35. Duan, H.; Tian, F. The development of standardized models of digital twin. *IFAC-PapersOnLine* **2020**, *53*, 726–731. [CrossRef]
36. Göllner, D.; Rasor, R.; Anacker, H.; Dumitrescu, R. Collaborative Modeling of Interoperable Digital Twins in a SoS Context. *Procedia CIRP* **2022**, *107*, 1089–1094. [CrossRef]
37. Kugler, S.; Kern, A.; Anderl, R. Method for the generation of use case related views for Digital Twins. *Procedia CIRP* **2021**, *104*, 1896–1900. [CrossRef]
38. Farsi, M.; Ariansyah, D.; Erkoyuncu, J.; Harrison, A. A digital twin architecture for effective product lifecycle cost estimation. *Procedia CIRP* **2021**, *100*, 506–511. [CrossRef]
39. Hickey, B.; Gachon, D.; Cosgrove, D. Digital Twin—A Tool for Project Management in Manufacturing. *Procedia Comput. Sci.* **2023**, *217*, 720–727. [CrossRef]
40. Mouthaan, M.; Frenken, K.; Piscicelli, L.; Vaskelainen, T. Systemic sustainability effects of contemporary digitalization: A scoping review and research agenda. *Futures* **2023**, *149*, 103142. [CrossRef]
41. Chen, X.; Kurdve, M.; Johansson, B.; Despeisse, M. Enabling the twin transitions: Digital technologies support environmental sustainability through lean principles. *Sustain. Prod. Consum.* **2023**, *38*, 13–27. [CrossRef]
42. Maschler, B.; Braun, D.; Jazdi, N.; Weyrich, M. Transfer learning as an enabler of the intelligent digital twin. *Procedia CIRP* **2021**, *100*, 127–132. [CrossRef]
43. Florea, A.; Lobov, A.; Lanz, M. Emotions-aware Digital Twins For Manufacturing. *Procedia Manuf.* **2020**, *51*, 605–612. [CrossRef]
44. Barari, A.; de Sales Guerra, M.; Cohen, Y.; Macchi, M. Editorial: Intelligent manufacturing systems towards industry 4.0 era. *J. Intell. Manuf.* **2021**, *32*, 1793–1796. [CrossRef]
45. Kuehner, K.J.; Scheer, R.; Strassburger, S. Digital Twin: Finding Common Ground—A Meta-Review. *Procedia CIRP* **2021**, *104*, 1227–1232. [CrossRef]
46. Terkaj, W.; Qi, Q.; Urgo, M.; Scott, P.; Jiang, X. Multi-scale modelling of manufacturing systems using ontologies and delta-lenses. *CIRP Ann.* **2021**, *70*, 361–364. [CrossRef]
47. Psarommatis, F. A generic methodology and a digital twin for zero defect manufacturing (ZDM) performance mapping towards design for ZDM. *J. Manuf. Syst.* **2021**, *59*, 507–521. [CrossRef]

48. Bouleux, G.; El Haouzi, H.; Cheutet, V.; Demesure, G.; Derigent, W.; Moyaux, T.; Trilling, L. Requirements for a Digital Twin for an Emergency Department. In *Service Oriented, Holonic and Multi-Agent Manufacturing Systems for Industry of the Future, Proceedings of the SOHOMA 2022, Bucharest, Romania, 22–23 September 2022*; Springer: Cham, Switzerland, 2023; Volume 1083.
49. Greco, A.; Caterino, M.; Fera, M.; Gerbino, S. Digital Twin for Monitoring Ergonomics during Manufacturing Production. *Appl. Sci.* **2020**, *10*, 7758. [CrossRef]
50. Fera, M.; Greco, A.; Caterino, M.; Gerbino, S.; Caputo, F.; Macchiaroli, R.; D'Amato, E. Towards Digital Twin Implementation for Assessing Production Line Performance and Balancing. *Sensors* **2020**, *20*, 97. [CrossRef]
51. Stojadinovic, S.; Vidosav, M.; Durakbasa, N.; Stanic, D. Contribution to the development of a digital twin based on CMM to support the inspection process. *Meas. Sens.* **2022**, *22*, 100372. [CrossRef]
52. Ruzsa, C. Digital twin technology—External data resources in creating the model and classification of different digital twin types in manufacturing. *Procedia Manuf.* **2021**, *54*, 209–215. [CrossRef]
53. Maes, V.K.; Potter, K.; Kratz, J. Features and defects characterisation for virtual verification and certification of composites: A review. *Compos. Part B Eng.* **2022**, *246*, 110282. [CrossRef]
54. Eramo, R.; Bordeleau, F.; Combemale, B.; van den Brand, M.; Wimmer, M.; Wortmann, A. Conceptualizing Digital Twins. *IEEE Softw.* **2022**, *39*, 39–46. [CrossRef]
55. Ahmadi, H.; Avishek, N.; Zaheer, K.; Kamran, S.; Susanto, R. Networked Twins and Twins of Networks: An Overview on the Relationship Between Digital Twins and 6G. *IEEE Commun. Stand. Mag.* **2021**, *5*, 154–160. [CrossRef]
56. Li, S.; Wang, J.; Rong, J.; Wei, W. A digital twin framework for product to-be-designed analysis based on operation data. *Procedia CIRP* **2022**, *109*, 179–184. [CrossRef]
57. Tang, W.; Tian, L.; Zheng, X.; Yan, K. Analyzing topics in social media for improving digital twinning based product development. *Digit. Commun. Netw.* 2022. [CrossRef]
58. Bellalouna, F. Case study for design optimization using the digital twin approach. *Procedia CIRP* **2021**, *100*, 595–600. [CrossRef]
59. Wang, Y.; Wang, X.; Ang, L. Digital Twin-Driven Analysis of Design Constraints. *Procedia CIRP* **2020**, *91*, 716–721. [CrossRef]
60. Shan Ng, M.; Hall, D.; Schmailzl, M.; Linner, T.; Bock, T. Identifying enablers and relational ontology networks in design for digital fabrication. *Autom. Constr.* **2022**, *144*, 104592.
61. Riesener, M.; Dölle, C.; Perau, S.; Lossie, P.; Schuh, G. Methodology for iterative system modeling in agile product development. *Procedia CIRP* **2021**, *100*, 439–444. [CrossRef]
62. Benfer, M.; Peukert, S.; Lanza, G. A Framework for Digital Twins for Production Network Management. *Procedia CIRP* **2021**, *104*, 1269–1274. [CrossRef]
63. Talkhestani, B.A.; Braun, D.; Schloegl, W.; Weyrich, M. Qualitative and quantitative evaluation of reconfiguring an automation system using Digital Twin. *Procedia CIRP* **2020**, *93*, 268–273. [CrossRef]
64. Feldt, J.; Kourouklis, T.; Kontny, H.; Wagenitz, A. Digital twin: Revealing potentials of real-time autonomous decisions at a manufacturing company. *Procedia CIRP* **2020**, *88*, 185–190. [CrossRef]
65. Mueller-Zhang, Z.; Oliveira Antonino, P.; Kuhn, T. Integrated Planning and Scheduling for Customized Production using Digital Twins and Reinforcement Learning. *IFAC-PapersOnLine* **2021**, *54*, 408–413. [CrossRef]
66. Ragazzini, L.; Negri, E.; Macchi, M. A Digital Twin-based Predictive Strategy for Workload Control. *IFAC-PapersOnLine* **2021**, *54*, 743–748.
67. Reichardt, P.; Lang, S.; Reggelin, T. Procedure model for the development and launch of intelligent assistance systems. *Procedia Comput. Sci.* **2021**, *180*, 968–977. [CrossRef]
68. Herkes, M.C.; Oversluizen, G. Using a systems approach to model a process digital twin. *IFAC-PapersOnLine* **2022**, *55*, 1906–1911. [CrossRef]
69. Kumbhar, M.; Ng, A.H.C.; Bandaru, S. A digital twin based framework for detection, diagnosis, and improvement of throughput bottlenecks. *J. Manuf. Syst.* **2023**, *66*, 92–106. [CrossRef]
70. Zhang, L.; He, S.; Li, B.; Mao, X.; Liang, K.; Hao, C. Research on the Modelling and Development of Flexibility in Production System Design Phase Driven by Digital Twins. *Appl. Sci.* **2022**, *12*, 2537. [CrossRef]
71. Mo, F.; Rehman, H.R.; Monetti, F.M.; Chaplin, J.C.; Sanderson, D.; Popov, A.; Maffei, A.; Ratchev, S. A framework for manufacturing system reconfiguration and optimisation utilising digital twins and modular artificial intelligence. *Robot. Comput. Integr. Manuf.* **2023**, *82*, 102524. [CrossRef]
72. Perno, M.; Hvam, L.; Haug, A. Implementation of digital twins in the process industry: A systematic literature review of enablers and barriers. *Comput. Ind.* **2022**, *134*, 103558. [CrossRef]
73. Eunike, A.; Wang, K.; Chiu, J.; Hsu, Y. Real-time resilient scheduling by digital twin technology in a flow-shop manufacturing system. *Procedia CIRP* **2022**, *107*, 668–674. [CrossRef]
74. Serrano-Ruiz, J.C.; Mula, J.; Poler, R. Smart manufacturing scheduling: A literature review. *J. Manuf. Syst.* **2021**, *61*, 265–287. [CrossRef]
75. Serrano-Ruiz, J.C.; Mula, J.; Poler, R. Development of a multidimensional conceptual model for job shop smart manufacturing scheduling from the Industry Industry 4.0 perspective. *J. Manuf. Syst.* **2022**, *63*, 185–202. [CrossRef]
76. Sahlab, N.; Braun, D.; Köhler, C.; Jazdi, N.; Weyrich, M. Extending the Intelligent Digital Twin with a context modeling service: A decision support use case. *Procedia CIRP* **2022**, *107*, 463–468. [CrossRef]
77. Wagner, S.B.; Milde, M.; Reinhart, G. The Digital Twin in Order Processing. *Procedia CIRP* **2021**, *104*, 863–868. [CrossRef]

78. Panagou, S.; Fruggiero, F.; Lerra, M.; del Vecchio, C.; Menchetti, F.; Piedimonte, L.; Natale, O.R.; Passariello, S. Feature investigation with Digital Twin for predictive maintenance following a machine learning approach. *IFAC-PapersOnLine* **2022**, *55*, 132–137. [CrossRef]
79. Zhong, D.; Xia, Z.; Zhu, Y.; Duan, J. Overview of predictive maintenance based on digital twin technology. *Heliyon* **2023**, *9*, 14534. [CrossRef] [PubMed]
80. D'Amico, D.R.; Erkoyuncu, J.A.; Addepalli, S.; Penver, S. Cognitive digital twin: An approach to improve the maintenance management. *CIRP J. Manuf. Sci. Technol.* **2022**, *38*, 613–630. [CrossRef]
81. van Dinter, R.; Tekinerdogan, B.; Catal, C. Predictive maintenance using digital twins: A systematic literature review. *Inf. Softw. Technol.* **2022**, *151*, 107008. [CrossRef]
82. You, Y.; Chen, C.; Hu, F.; Liu, Y.; Ji, Z. Advances of Digital Twins for Predictive Maintenance. *Procedia Comput. Sci.* **2022**, *200*, 1471–1480. [CrossRef]
83. Davies, O.; Makkattil, A.; Jiang, C.; Farsi, M. A Digital Twin Design for Maintenance Optimization. *Procedia CIRP* **2022**, *109*, 395–400. [CrossRef]
84. Frantzén, M.; Bandaru, S.; Ng, A.H.C. Digital-twin-based decision support of dynamic maintenance task prioritization using simulation-based optimization and genetic programming. *Decis. Anal. J.* **2022**, *3*, 100039. [CrossRef]
85. Nota, G.; Postiglione, A.; Carvello, R. Text mining techniques for the management of predictive maintenance. *Procedia Comput. Sci.* **2022**, *200*, 778–792. [CrossRef]
86. Santos, C.; Lima, R.; Fabiano, L.; Queiroz, J.; Balestrassi, P.; Montevechi, J.A.B. A decision support tool for operational planning: A Digital Twin using simulation and forecasting methods. *Produção* **2020**, *30*, e20200018. [CrossRef]
87. Lee, L.; Alghamdi, A.; Zaidi, A.K. Creating a Digital Twin of an Insider Threat Detection Enterprise Using Model-Based Systems Engineering. In Proceedings of the 2022 IEEE International Systems Conference (SysCon), Montreal, QC, Canada, 25–28 April 2022; pp. 1–7.
88. Novikov, D.; Bakhtadze, N.; Elpashev, D.; Suleykin, A. Integrated Resource Management in the Digital Ecosystem of the Enterprise Based on Intelligent Consorts. *IFAC-PapersOnLine* **2022**, *55*, 2330–2335. [CrossRef]
89. Pires, F.; Leitão, P.; Moreira, A.P.; Ahmad, B. Reinforcement learning based trustworthy recommendation model for digital twin-driven decision-support in manufacturing systems. *Comput. Ind.* **2023**, *148*, 103884. [CrossRef]
90. Attaran, M.; Celik, B.G. Digital Twin: Benefits, use cases, challenges, and opportunities. *Decis. Anal. J.* **2023**, *6*, 100165. [CrossRef]
91. Silva, H.; Soares, A.L. Digital Platforms as Enablers of Smart Product-Service Systems. In *Smart and Sustainable Collaborative Networks 4.0, Proceedings of the 22nd IFIP WG 5.5 Working Conference on Virtual Enterprises, PRO-VE 2021, Saint-Étienne, France, 22–24 November 2021*; Camarinha-Matos, L.M., Boucher, X., Afsarmanesh, H., Eds.; Springer: Cham, Switzerland, 2021; Volume 629.
92. Bertoni, M.; Bertoni, A. Designing solutions with the product-service systems digital twin: What is now and what is next? *Comput. Ind.* **2022**, *138*, 103629. [CrossRef]
93. Wang, Y.; Wang, X.; Liu, A. Digital Twin-driven Supply Chain Planning. *Procedia CIRP* **2020**, *93*, 198–203. [CrossRef]
94. McCausland, T. Digital Twins. *Res. Technol. Manag.* **2021**, *65*, 69–71. [CrossRef]
95. Van der Valk, H.; Strobel, G.; Winkelmann, S.; Hunker, J.; Tomczyk, M. Supply Chains in the Era of Digital Twins—A Review. *Procedia Comput. Sci.* **2022**, *204*, 156–163. [CrossRef]
96. Maheshwari, P.; Kamble, S. The Application of Supply Chain Digital Twin to Measure Optimal Inventory Policy. *IFAC-PapersOnLine* **2022**, *55*, 2324–2329. [CrossRef]
97. Longo, F.; Mirabelli, G.; Padovano, A.; Solina, V. The Digital Supply Chain Twin paradigm for enhancing resilience and sustainability against COVID-like crises. *Procedia Comput. Sci.* **2023**, *217*, 1940–1947. [CrossRef]
98. Rocha, A.; Barata, J. Digital twin-based optimiser for self-organised collaborative cyber-physical production systems. *Manuf. Lett.* **2021**, *29*, 79–83. [CrossRef]
99. Arunachalam, K.; Thangamuthu, S.; Shanmugam, V.; Mukesh, R.; Premraj, K. Deep learning and optimisation for quality of service modelling. *J. King Saud Univ. Comput. Inf. Sci.* **2022**, *34*, 5998–6007. [CrossRef]
100. Stamer, F.; Maier, S.; Peukert, S.; Lanza, G. Adaptive and Dynamic Feedback Loops between Production System and Production Network based on the Asset Administration Shell. *Procedia CIRP* **2022**, *112*, 79–84. [CrossRef]
101. Caccamo, C.; Pedrazzoli, P.; Eleftheriadis, R.; Magnanini, M. Using the Process Digital Twin as a tool for companies to evaluate the Return of Investment of manufacturing automation. *Procedia CIRP* **2022**, *107*, 724–728. [CrossRef]
102. Ma, S.; Wei, D.; Liu, Y.; Ren, S.; Yang, H. Digital twin and big data-driven sustainable smart manufacturing based on information management systems for energy-intensive industries. *Appl. Energy* **2022**, *326*, 119986. [CrossRef]
103. Langlotz, P.; Klar, M.; Yi, L.; Hussong, M.; Sousa, F.; Aurich, J. Concept of hybrid modeled digital twins and its application for an energy management of manufacturing systems. *Procedia CIRP* **2022**, *112*, 549–554. [CrossRef]
104. Ralph, B.; Schwarz, A.; Stockinger, M. An Implementation Approach for an Academic Learning Factory for the Metal Forming Industry with Special Focus on Digital Twins and Finite Element Analysis. *Procedia Manuf.* **2020**, *45*, 253–258. [CrossRef]
105. Ulmer, J.; Braun, S.; Cheng, C.; Dowey, S.; Wollert, J. Usage of digital twins for gamification applications in manufacturing. *Procedia CIRP* **2022**, *107*, 675–680. [CrossRef]
106. Neghina, M.; Matei, A.; Zamfirescu, B. Multimodal emotion detection from multiple data streams for improved decision-making. *Procedia Comput. Sci.* **2022**, *214*, 1082–1089.

107. Ehrhardt, J.; Hoffmann, C. The Digital Shadow: Developing a universal model for the automated optimization of cyber-physical production systems based on real-time data. *Procedia CIRP* **2022**, *93*, 304–310. [CrossRef]
108. Kubie, E.C. Recollections of the first software company. *IEEE Ann. Hist. Comput.* **1994**, *16*, 65–71. [CrossRef] [PubMed]
109. Sommerville, I. *Software Engineering*, 9th ed.; Pearson Education Inc.: Boston, MA, USA, 2011.
110. Poppendieck, M.; Poppendieck, T. *Lean Software Development: An Agile Toolkit*; Addison-Wesley Professional: Boston, MA, USA, 2003.
111. *ISO/IEC 27001:2022*; Information Security, Cybersecurity and Privacy Protection—Information Security Management Systems Requirements. ISO: Geneva, Switzerland, 2022.
112. Chappell, D. What Is Application Lifecycle Management. 7 December 2014. Available online: http://davidchappell.com/ (accessed on 1 December 2023).
113. *ISO/IEC/IEEE 12207:2017*; Systems and Software Engineering—Software Life Cycle Processes. ISO: Geneva, Switzerland, 2017.
114. Antonino, P.O.; Capilla, R.; Kazman, R.; Kuhn, T.; Schnicke, F.; Treichel, T.; Bachorek, A.; Müller-Zhang, Z.; Salamanca, V. Continuous engineering for Industry 4.0 architectures and systems. *Softw. Pract. Exp.* **2022**, *52*, 2241–2262. [CrossRef]
115. Nakagawa, E.; Oliveira Antonino, P.; Schnicke, F.; Kuhn, T.; Liggesmeyer, P. Continuous Systems and Software Engineering for Industry 4.0: A disruptive view. *Inf. Softw. Technol.* **2021**, *135*, 106562. [CrossRef]
116. Jones, D.; Nassehi, A.; Snider, C.; Gopsill, J.; Rosso, P.; Real, R.; Goudswaard, M.; Hicks, B. Towards integrated version control of virtual and physical artifacts in new product development: Inspirations from software engineering and the digital twin paradigm. *Procedia CIRP* **2021**, *100*, 283–288. [CrossRef]
117. Kofod-Petersen, A. *How to Do a Structured Literature Review in Computer Science*; (Version 0.2); Alexandra Institute: Copenhague, Denmark, 2014.
118. Page, M.; McKenzie, J.E.; Bossuyt, P.M.; Boutron, I.; Hoffmann, T.C.; Mulrow, C.D.; Shamseer, L.; Tetzlaff, J.M.; Akl, E.A.; Brennan, S.E.; et al. Declaración PRISMA 2020: Una guía actualizada para la publicación de revisiones sistemáticas. *Rev. Española De Cardiol.* **2021**, *74*, 790–799. [CrossRef]
119. West, T.; Blackburn, M.R. Is Digital Thread/Digital Twin Affordable? A Systemic Assessment of the Cost of DoD's Latest Manhattan Project. *Procedia Comput. Sci.* **2017**, *114*, 47–56. [CrossRef]
120. Hofmann, C.; Lauber, S.; Haefner, B.; Lanza, G. Development of an agile development method based on Kanban for distributed part-time teams and an introduction framework. *Procedia Manuf.* **2018**, *23*, 45–50. [CrossRef]
121. Bauer, T.; Oliveira Antonino, P.; Kuhn, T. Towards Architecting Digital Twin-Pervaded Systems. In Proceedings of the 2019 IEEE/ACM 7th International Workshop on Software Engineering for Systems-of-Systems (SESoS) and 13th Workshop on Distributed Software Development, Software Ecosystems and Systems-of-Systems (WDES), Montreal, QC, Canada, 28 May 2019; pp. 66–69.
122. Cioroaica, E.; Kuhn, T.; Buhnova, B. (Do Not) Trust in Ecosystems. In Proceedings of the 2019 IEEE/ACM 41st International Conference on Software Engineering: New Ideas and Emerging Results (ICSE-NIER), Montreal, QC, Canada, 25–31 May 2019; pp. 9–12.
123. Loizou, S.; Elgammal, A.; Kumara, I.; Christodoulou, P.; Papazoglou, M.P.; Andreou, A.S. A Smart Product Co-design and Monitoring Framework Via Gamification and Complex Event Processing. In Proceedings of the International Conference on Enterprise Information Systems, Heraklion, Crete, Greece, 3–5 May 2019.
124. Eisenträger, M.; Adler, S.; Fischer, E. Rethinking Software Development for Collaboration Technologies. In Proceedings of the 2019 IEEE International Conference on Engineering, Technology and Innovation (ICE/ITMC), Valbonne Sophia-Antipolis, France, 17–19 June 2019; pp. 1–9.
125. Caporuscio, M.; Edrisi, F.; Hallberg, M.; Johannesson, A.; Kopf, C.; Perez-Palacin, D. Architectural Concerns for Digital Twin of the Organization. In *Software Architecture, Proceedings of the 14th European Conference, ECSA 2020, L'Aquila, Italy, 14–18 September 2020*; Jansen, A., Malavolta, I., Muccini, H., Ozkaya, I., Zimmermann, O., Eds.; Lecture Notes in Computer Science; Springer: Cham, Switzerland, 2020; Volume 12292.
126. Succar, B.; Poirier, E. Lifecycle information transformation and exchange for delivering and managing digital and physical assets. *Autom. Constr.* **2020**, *112*, 103090. [CrossRef]
127. Minerva, R.; Lee, G.M.; Crespi, N. Digital Twin in the IoT Context: A Survey on Technical Features, Scenarios, and Architectural Models. *Proc. IEEE* **2020**, *108*, 1785–1824. [CrossRef]
128. Gennady, K.; Nadezhda, N.; Vyacheslav, A.; Maria, S.; Rodionova, L. Formal Representation of the Model of the Designed Software-Analytical Complex Based on the Principle of the Necessary Variety of Structural Relationships. In Proceedings of the 8th Scientific Conference on Information Technologies for Intelligent Decision Making Support (ITIDS 2020), Ufa, Russia, 6–9 October 2020; pp. 222–227.
129. Pokhrel, A.; Katta, V.; Colomo-Palacios, R. Digital Twin for Cybersecurity Incident Prediction: A Multivocal Literature Review. In *Proceedings of the IEEE/ACM 42nd International Conference on Software Engineering Workshops (ICSEW'20), Seoul, Republic of Korea, 27 June–19 July 2020*; Association for Computing Machinery: New York, NY, USA, 2020; pp. 671–678.
130. Hughes, J.; Hristozov, A.; Hudak, J.; Yankel, J. TwinOps–DevOps meets model-based engineering and digital twins for the engineering of CPS. In *MODELS'20, Proceedings of the 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems: Companion Proceeding, Virtual Event, Canada, 18–23 October 2020*; Association for Computing Machinery: New York, NY, USA, 2020; Volume 94, pp. 1–5. [CrossRef]

131. Dalibor, M.; Michael, J.; Rumpe, B.; Varga, S.; Wortmann, A. Towards a Model-Driven Architecture for Interactive Digital Twin Cockpits. In *Conceptual Modeling, Proceedings of the 39th International Conference, ER 2020, Vienna, Austria, 3–6 November 2020*; Dobbie, G., Frank, U., Kappel, G., Liddle, S.W., Mayr, H.C., Eds.; Lecture Notes in Computer Science; Springer: Cham, Switzerland, 2020; Volume 12400, pp. 377–387.
132. Sun, L.; Pei, A.; Qi, X.; Cao, S.; Yang, R.; Liu, X. Dynamic Analysis of Digital Twin System Based on Five-Dimensional Model. *J. Phys. Conf. Ser.* **2019**, *1486*, 072038. [CrossRef]
133. Xu, Y.; Päivärinta, T.; Kuvaja, P. Digital Twins as Software and Service Development Ecosystems in Industry 4.0: Towards a Research Agenda. In *Big Data and Security, Proceedings of the First International Conference, ICBDS 2019, Nanjing, China, 20–22 December 2019*; Communications and Information Science; Springer: Singapore, 2020; Volume 1210.
134. Pileggi, P.; Lazovik, E.; Broekhuijsen, J.; Borth, M.; Verriet, J. Lifecycle Governance for Effective Digital Twins: A Joint Systems Engineering and IT Perspective. In Proceedings of the 2020 IEEE International Systems Conference (SysCon), Montreal, QC, Canada, 24 August–20 September 2020; pp. 1–8.
135. Zhang, L.; Longfei, Z.; Horn, B. Building a right digital twin with model engineering. *J. Manuf. Syst.* **2021**, *59*, 151–164. [CrossRef]
136. Davila Delgado, J.M.; Oyedele, L. Digital Twins for the built environment: Learning from conceptual and process models in manufacturing. *Adv. Eng. Inform.* **2021**, *49*, 101332. [CrossRef]
137. Bruneliere, H.; Muttillo, V.; Eramo, R.; Berardinelli, L.; Gómez, A.; Bagnato, A.; Sadovykh, A.; Cicchetti, A. AIDOaRt: AI-augmented Automation for DevOps, a model-based framework for continuous development in Cyber–Physical Systems. *Microprocess. Microsyst.* **2022**, *94*, 104672. [CrossRef]
138. Eiden, A.; Eickhoff, T.; Gries, J.; Göbel, J.C.; Psota, T. Supporting semantic PLM by using a lightweight engineering metadata mapping engine. *Procedia CIRP* **2021**, *100*, 690–695. [CrossRef]
139. Oakes, B.; Parsai, A.; Van Mierlo, S.; Demeyer, S.; Denil, J.; De Meulenaere, P.; Vangheluwe, H. Improving Digital Twin Experience Reports. In Proceedings of the International Conference on Model-Driven Engineering and Software Development, Virtual Event, 8–10 February 2021; pp. 179–190.
140. Cheng, Z.; Guo, J. Team Analysis Based on Digital Twin Within RoboCup 2D Simulation. In Proceedings of the 2021 2nd International Conference on Big Data & Artificial Intelligence & Software Engineering (ICBASE), Zhuhai, China, 24–26 September 2021; pp. 691–695.
141. Ahlgren, J.; Bojarczuk, K.; Drossopoulou, S.; Dvortsova, I.; George, J.; Gucevska, N.; Harman, M.; Lomeli, M.; Lucas, S.M.M.; Meijer, E.; et al. Facebook's Cyber–Cyber and Cyber–Physical Digital Twins. In *Evaluation and Assessment in Software Engineering (EASE 2021)*; Association for Computing Machinery: New York, NY, USA, 2021; pp. 1–9.
142. Asadi, A.R. Cognitive Ledger Project: Towards Building Personal Digital Twins Through Cognitive Blockchain. In Proceedings of the 2021 2nd International Informatics and Software Engineering Conference (IISEC), Ankara, Turkey, 16–17 December 2021; pp. 1–5.
143. Strandberg, P.; Frasheri, M.; Enoiu, E. Ethical AI-Powered Regression Test Selection. In Proceedings of the IEEE International Conference on Artificial Intelligence Testing (AITest), Oxford, UK, 23–26 August 2021; pp. 83–84.
144. Autiosalo, J.; Siegel, J.; Tammi, K. Twinbase: Open-Source Server Software for the Digital Twin Web. *IEEE Access* **2021**, *9*, 140779–140798. [CrossRef]
145. Brockhoff, T.; Heithoff, M.; Koren, I.; Michael, J.; Pfeiffer, J.; Rumpe, B.; Uysal, M.S.; Aalst, W.; Wortmann, A. Process Prediction with Digital Twins. In Proceedings of the 2021 ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C), Fukuoka, Japan, 10–15 October 2021; pp. 182–187.
146. Jordan, S. Co-evolving Digital Architecture Twins (short paper). In Proceedings of the European Conference on Software Architecture, Virtual, 13–17 September 2021.
147. Malakuti, S. Emerging Technical Debt in Digital Twin Systems. In Proceedings of the 26th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), Vasteras, Sweden, 7–10 September 2021; pp. 1–4.
148. Schroeder, G.; Steinmetz, C.; Rodrigues, R.; Henriques, R.; Rettberg, A.; Pereira, C. A Methodology for Digital Twin Modeling and Deployment for Industry 4.0. *Proc. IEEE* **2020**, *109*, 556–567. [CrossRef]
149. Engels, G. Predict the Future: Preventing unanticipated changes is the ultimate challenge for self-adaptive systems. In Proceedings of the 2021 International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS), Madrid, Spain, 18–24 May 2021; pp. 260–261.
150. Poltronieri, F.; Tortonesi, M.; Stefanelli, C. ChaosTwin: A Chaos Engineering and Digital Twin Approach for the Design of Resilient IT Services. In Proceedings of the 2021 17th International Conference on Network and Service Management (CNSM), Izmir, Turkey, 25–29 October 2021; pp. 234–238.
151. Muñoz, P.; Troya, J.; Vallecillo, A. Using UML and OCL Models to Realize High-Level Digital Twins. In Proceedings of the 2021 ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C), Fukuoka, Japan, 10–15 October 2021; pp. 212–220.
152. Fehlmann, T.; Eberhard, K. ART for Agile—Autonomous Real-Time Testing in the Product Development Cycle. In Proceedings of the European Conference on Software Process Improvement (2021), Krems, Austria, 1–3 September 2021; pp. 377–390.
153. Asikainen, T.; Männistö, T. Undulate: A framework for data-driven software engineering enabling soft computing. *Inf. Softw. Technol.* **2022**, *152*, 107039. [CrossRef]

154. Karagiannis, D.; Buchmann, R.A.; Utz, W. The OMiLAB Digital Innovation environment: Agile conceptual models to bridge business value with Digital and Physical Twins for Product-Service Systems development. *Comput. Ind.* **2022**, *138*, 103631. [CrossRef]
155. Subha, R.; Zhang, J. An optimal construction of smart aged homes based on SDLC using smart sensors and agent networks. *Int. J. Intell. Netw.* **2022**, *3*, 138–142. [CrossRef]
156. Ferreira, B.; Marques, S.; Kalinowski, M.; Lopes, H.; Barbosa, S. Lessons Learned to Improve the UX Practices in Agile Projects Involving Data Science and Process Automation. *Inf. Softw. Technol.* **2023**, *155*, 107106. [CrossRef]
157. Vyhmeister, E.; Castañé, G.G.; Buchholz, J.; Östberg, P. Lessons learn on responsible AI implementation: The ASSISTANT use case. *IFAC-PapersOnLine* **2022**, *55*, 377–382. [CrossRef]
158. Das, S.; Ahuja, M.; Singi, K.; Dey, K.; Kaulgud, V.; Raman, M.; Tung, T. Digital Twin-based Fault Analysis in Hybrid-cloud Applications. In Proceedings of the 2022 IEEE/ACM 10th International Workshop on Software Engineering for Systems-of-Systems and Software Ecosystems (SESoS), Pittsburgh, PA, USA, 16 May 2022; pp. 29–32.
159. Dobaj, J.; Riel, A.; Krug, T.; Seidl, M.; Macher, G.; Egretzberger, M. Towards Digital Twin-enabled DevOps for CPS providing Architecture-Based Service Adaptation & Verification at Runtime. In Proceedings of the 2022 International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS), Pittsburgh, PA, USA, 22–24 May 2022; pp. 132–143.
160. Rivera, L.; Jiménez, M.; Villegas, N.; Tamura, G.; Muller, H. Toward Autonomic, Software-Intensive Digital Twin Systems. *IEEE Softw.* **2021**, *39*, 20–26. [CrossRef]
161. Kamburjan, E.; Johnsen, E.B. Knowledge Structures Over Simulation Units. In Proceedings of the Annual Modeling and Simulation Conference (ANNSIM), San Diego, CA, USA, 18–20 July 2022; pp. 78–89.
162. Nakajima, T.; Simonetta, A. Concept of Quality Digital Twin in Agile Development. In Proceedings of the 2022 4th International Workshop on Experience with SQuaRE Series and Its Future Direction, Tokyo, Japan, 6 December 2022.
163. Bechu, G.; Beugnard, A.; Cao, C.; Perez, Q.; Urtado, C.; Vauttier, S. A software engineering point of view on digital twin architecture. In Proceedings of the 2022 IEEE 27th International Conference on Emerging Technologies and Factory Automation (ETFA), Stuttgart, Germany, 6–9 September 2022.
164. Guzina, L.; Ferko, E.; Bucaioni, A. Investigating Digital Twin: A Systematic Mapping Study. *Adv. Transdiscipl. Eng.* **2022**, *21*, 449–460.
165. Frick, N.; Metternich, J. The Digital Value Stream Twin. *Systems* **2022**, *10*, 102. [CrossRef]
166. Michael, J.; Pfeiffer, J.; Rumpe, B.; Wortmann, A. Integration Challenges for Digital Twin Systems-of-Systems. In Proceedings of the 2022 IEEE/ACM 10th International Workshop on Software Engineering for Systems-of-Systems and Software Ecosystems (SESoS), Pittsburgh, PA, USA, 16 May 2022; pp. 9–12.
167. Bano, D.; Michael, J.; Rumpe, B.; Varga, S.; Weske, M. Process-aware digital twin cockpit synthesis from event logs. *J. Comput. Lang.* **2022**, *70*, 101121. [CrossRef]
168. Kholkar, D.; Roychoudhury, S.; Kulkarni, V.; Reddy, S. Learning to Adapt–Software Engineering for Uncertainty. In Proceedings of the ISEC 2022: 15th Innovations in Software Engineering Conference, Gandhinagar, India, 24–26 February 2022; pp. 1–5.
169. Epiphaniou, G.; Hammoudeh, M.; Yuan, H.; Maple, C.; Ani, U. Digital twins in cyber effects modelling of IoT/CPS points of low resilience. *Simul. Model. Pract. Theory* **2023**, *125*, 102744. [CrossRef]
170. Hu, W.; Fang, J.; Zhang, T.; Liu, Z.; Tan, J. A new quantitative digital twin maturity model for high-end equipment. *J. Manuf. Syst.* **2023**, *66*, 248–259. [CrossRef]
171. Alvarez-Rodríguez, J.M.; Mendieta, R.; Cibrián, E.; Llorens, J. Towards a method to quantitatively measure toolchain interoperability in the engineering lifecycle: A case study of digital hardware design. *Comput. Stand. Interfaces* **2023**, *86*, 103744. [CrossRef]
172. Kügler, P.; Dworschak, F.; Schleich, B.; Wartzack, S. The evolution of knowledge-based engineering from a design research perspective: Literature review 2012–2021. *Adv. Eng. Inform.* **2023**, *55*, 101892. [CrossRef]
173. Lu, Q.; Zhu, L.; Xu, X.; Whittle, J. Responsible-AI-by-Design: A Pattern Collection for Designing Responsible AI Systems. *IEEE Softw.* **2023**, *40*, 63–71. [CrossRef]
174. Lünnemann, P.; Lindow, K.; Goßlau, L. Implementing digital twins in existing infrastructures. *Forsch. Im Ingenieurwesen* **2023**, *87*, 421–429. [CrossRef]
175. Ardito, C.; Bernhaupt, R.; Sauer, S. Human-Centered Software Engineering: Rethinking the Interplay of Human–Computer Interaction and Software Engineering in the Age of Digital Transformation. In Proceedings of the INTERACT 2023: 19th IFIP TC13 International Conference, York, UK, 28 August–1 September 2023; Proceedings, Part IV. Springer: Cham, Switzerland, 2023; pp. 638–643.
176. Frepoli, C.; Valeri, J.; Martin, R.P. Development of an Enterprise Digital Platform for Risk-Informed Design. In Proceedings of the Probabilistic Safety Assessment and Management Conference (PSAM 16), Honolulu, HI, USA, 26 June–1 July 2022.
177. Gorodetsky, V.; Skobelev, P. System engineering view on multi-agent technology for industrial applications: Barriers and prospects. *Cybern. Phys.* **2020**, *9*, 13–30. [CrossRef]
178. Newrzella, S.R.; Franklin, D.W.; Haider, S. Methodology for Digital Twin Use Cases: Definition, Prioritization, and Implementation. *IEEE Access* **2022**, *10*, 75444–75457. [CrossRef]
179. Yue, T.; Ali, S. Evolve the Model Universe of a System Universe. In Proceedings of the 2023 38th IEEE/ACM International Conference on Automated Software Engineering (ASE), Luxembourg, 13–15 September 2023; pp. 1726–1731.

180. AboElHassan, A.; Yacout, S. A digital shadow framework using distributed system concepts. *J. Intell. Manuf.* **2022**, *34*, 3579–3598. [CrossRef]
181. Rios, J.; Staudter, G.; Weber, M.; Anderl, R. A Review, Focused on Data Transfer Standards of the Uncertainty Representation in the Digital Twin Context. In Proceedings of the 2019 16th IFIP WC 5.1 International Conference, PLM 2019, Moscow, Rusia, 8–12 July 2019; pp. 24–33.
182. Halenar, I.; Juhás, M.; Juhásová, B.; Borkin, D. Virtualization of Production Using Digital Twin Technology. In Proceedings of the 20th International Carpathian Control Conference (ICCC), Krakow-Wieliczka, Poland, 26–29 May 2019; pp. 1–5.
183. Hillenbrand, J.; Gönnheimer, P.; Gerlitz, E.; Fleischer, J. Design and implementation of a holistic framework for data integration in industrial machine and sensor networks. *Procedia CIRP* **2021**, *104*, 1771–1776. [CrossRef]
184. Liyanage, R.; Tripathi, N.; Päivärinta, T.; Xu, Y. Digital Twin Ecosystems: Potential Stakeholders and Their Requirements. In *Software Business, Proceedings of the 13th International Conference, ICSOB 2022, Bolzano, Italy, 8–11 November 2022*; Carroll, N., Nguyen-Duc, A., Wang, X., Stray, V., Eds.; Lecture Notes in Business Information Processing; Springer: Cham, Switzerland, 2022; Volume 463.
185. Reiche, F.; Timinger, H. Process Model for Integrated Product Lifecycles Using Digital Twins and Predictive Analytics. In Proceedings of the 2021 IEEE Technology & Engineering Management Conference-Europe (TEMSCON-EUR), Dubrovnik, Croatia, 17–20 May 2021; pp. 1–5.
186. Tisi, M.; Bruneliere, H.; Lara, J.; Di Ruscio, D.; Kolovos, D. Towards Twin-Driven Engineering: Overview of the State-of-The-Art and Research Directions. In *IFIP Advances in Information and Communication Technology, Proceedings of the IFIP WG 5.7 International Conference, APMS 2021, Nantes, France, 5–9 September 2021*; Springer: Cham, Switzerland, 2021; Volume 630.
187. Xia, K.; Sacco, C.; Kirkpatrick, M.; Harik, R.; Bayoumi, A. Virtual Commissioning of Manufacturing System Intelligent Control. In Proceedings of the SAMPE 2019, Charlotte, NC, USA, 20–23 May 2019.
188. Feng, H.; Gomes, C.; Gil, S.; Mikkelsen, P.; Tola, D.; Larsen, P.; Sandberg, M. Integration of the Mape-K Loop in Digital Twins. In Proceedings of the 2022 Annual Modeling and Simulation Conference (ANNSIM), San Diego, CA, USA, 18–20 July 2022; pp. 102–113.
189. Carver, J.; Staron, M.; Capilla, R.; Muccini, H.; Hochstein, L. Digital Twins. *IEEE Softw.* **2022**, *39*, 97–99. [CrossRef]
190. Al-Najjar, A.; Rao, N. Virtual Infrastructure Twin for Computing-Instrument Ecosystems: Software and Measurements. *IEEE Access* **2023**, *11*, 20254–20266. [CrossRef]
191. Adams, M.; Li, X.; Boucinha, L.; Kher, S.S.; Banerjee, P.; Gonzalez, J.L. Hybrid Digital Twins: A Primer on Combining Physics-Based and Data Analytics Approaches. *IEEE Softw.* **2022**, *39*, 47–52. [CrossRef]
192. Lestingi, L.; Zerla, D.; Bersani, M.; Rossi, M. Specification, stochastic modeling, and analysis of interactive service robotic applications. *Robot. Auton. Syst.* **2023**, *163*, 104387. [CrossRef]
193. Reed, S.; Löfstrand, M.; Andrews, J.D. Modeling cycle for simulation digital twins. *Manuf. Lett.* **2021**, *28*, 54–58. [CrossRef]
194. Djukić, V.; Popovic, A. Handling complex representations in visual modeling tools for MDSD/DSM by means of code generator languages. *J. Comput. Lang.* **2023**, *75*, 101208. [CrossRef]
195. Khalajzadeh, H.; Simmons, A.; Verma, T.; Abdelrazek, M.; Grundy, J.; Hosking, J.; He, Q.; Ratnakanthan, P.; Zia, A.; Law, M. BiDaML in Practice: Collaborative Modeling of Big Data Analytics Application Requirements. In *ENASE 2020: Evaluation of Novel Approaches to Software Engineering, Proceedings of the 15th International Conference, Prague, Czech Republic, 5–6 May 2020*; Springer: Cham, Switzerland, 2020; Volume 1375.
196. Kirchhof, J.C.; Michael, J.; Rumpe, B.; Varga, S.; Wortmann, A. Model-driven Digital Twin Construction: Synthesizing the Integration of Cyber-Physical Systems with Their Information Systems. In Proceedings of the 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems, Virtual Event, 16–23 October 2020; pp. 90–101.
197. Tsiatsis, V.; Karnouskos, S.; Höller, J.; Boyle, D.; Mulligan, C. An Architecture Perspective. In *Internet of Things*, 2nd ed.; Academic Press: Cambridge, MA, USA, 2019; pp. 49–63.
198. Turk, Ž.; Klinc, R. A social–product–process framework for construction. *Build. Res. Inf.* **2019**, *48*, 1–16. [CrossRef]
199. Zheng, C.; An, Y.; Wang, Z.; Qin, X.; Yu, F.; Yicha, Z. Heterogeneous requirement gathering for generative design of robotic manufacturing systems. *Procedia CIRP* **2021**, *104*, 1861–1866. [CrossRef]
200. Ferko, E.; Bucaioni, A.; Behnam, M. Architecting Digital Twins. *IEEE Access* **2022**, *10*, 50335–50350. [CrossRef]
201. Boyes, H.; Watson, T. Digital twins: An analysis framework and open issues. *Comput. Ind.* **2022**, *143*, 103763. [CrossRef]
202. Corradini, F.; Silvestri, M. Design and testing of a digital twin for monitoring and quality assessment of material extrusion process. *Addit. Manuf.* **2022**, *51*, 102633. [CrossRef]
203. Chaudhary, H.; Guevara, I.; John, J.; Singh, A.; Ghosal, A.; Pesch, D.; Margaria, T. Model-Driven Engineering in Digital Thread Platforms: A Practical Use Case and Future Challenges. In *Leveraging Applications of Formal Methods, Verification and Validation. Practice, Proceedings of the 11th International Symposium, ISoLA 2022, Rhodes, Greece, 22–30 October 2022*; Lecture Notes in Computer Science; Springer: Cham, Switzerland, 2022; Volume 13704.
204. Hunhevicz, J.; Motie, M.; Hall, D. Digital building twins and blockchain for performance-based (smart) contracts. *Autom. Constr.* **2022**, *133*, 103981. [CrossRef]
205. Tekinerdogan, B.; Cor, V. Systems Architecture Design Pattern Catalog for Developing Digital Twins. *Sensors* **2020**, *20*, 5103. [CrossRef]

206. Ricci, A.; Croatti, A.; Mariani, S.; Montagna, S.; Picone, M. Web of Digital Twins. *ACM Trans. Internet Technol.* **2022**, *22*, 1–30. [CrossRef]
207. Kephart, J.O.; Chess, D.M. The vision of autonomic computing. *Computer* **2003**, *36*, 41–50. [CrossRef]
208. Bohlen, J.M.; Beal, G.M. *The Diffusion Process. Increasing Understanding of Public Problems and Policies*; Farm Foundation: Bloomington, IL, USA, 1956; pp. 111–121.
209. Bower, J.L.; Christensen, C.M. Disruptive Technologies: Catching the Wave. *Harvard Business Review*, 1 January 1995.
210. *ISO 23247-2:2021*; Automation Systems and Integration—Digital Twin Framework for Manufacturing—Part 2: Reference Architecture. ISO: Geneva, Switzerland, 2021.
211. *ISO/IEC 25010:2011*; Systems and Software Engineering—Systems and Software Quality Requirements and Evaluation (SQuaRE). ISO: Geneva, Switzerland, 2011.
212. Open Services for Lifecycle Collaboration. Oasis Open Project. 2019. Available online: https://open-services.net/ (accessed on 1 December 2023).
213. Guo, J.; Lv, Z. Application of Digital Twins in multiple fields. *Multimed. Tools Appl.* **2022**, *81*, 26941–26967. [CrossRef] [PubMed]
214. Akroyd, J.; Mosbach, S.; Bhave, A.; Kraft, M. Universal Digital Twin—A Dynamic Knowledge Graph. *Data-Centric Eng.* **2021**, *2*, e14. [CrossRef]
215. State of Agile. Available online: https://stateofagile.com/ (accessed on 1 December 2023).
216. Huettermann, M. *Agile ALM, Lightweight Tools, and Agile Strategies*; Manning: Shelter Island, NY, USA, 2011; ISBN 9781935182634.
217. Schönig, S.; Hornsteiner, M.; Stoiber, C. Towards Process-Oriented IIoT Security Management: Perspectives and Challenges. In *Enterprise, Business-Process and Information Systems Modeling, Proceedings of the 23rd International Conference BPMDS 2022 and 27th International Conference, EMMSAD 2022, Leuven, Belgium, 6–7 June 2022*; Lecture Notes in Business Information Processing; Springer: Cham, Switzerland, 2022; Volume 450.
218. Borowski, P. Digitization, Digital Twins, Blockchain, and Industry 4.0 as Elements of Management Process in Enterprises in the Energy Sector. *Energies* **2021**, *14*, 1885. [CrossRef]
219. Ahmad, A.; Saad, M.; Bassiouni, M.; Mohaisen, D. Towards Blockchain-Driven, Secure, and Transparent Audit Logs. In Proceedings of the MobiQuitous '18: 15th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services, New York, NY, USA, 5–7 November 2018; pp. 443–448.
220. Forsgren, N.; Humble, J.; Kim, G. *Accelerate: The Science of Lean Software and DevOps Building and Scaling High Performing Technology Organizations*, 1st ed.; IT Revolution Press: Portland, OR, USA, 2018.