The Systematic Literature Review on Digital Twin Architectures for TwinArch Design

Alessandra Somma
Università degli studi di Napoli Federico II, Italy
alessandra.somma@unina.it

Domenico Amalfitano
Università degli studi di Napoli Federico II, Italy
domenico.amalfitano@unina.it

Alessandra De Benedictis
Università degli studi di Napoli Federico II, Italy
alessandra.debenedictis@unina.it

Patrizio Pelliccione
Gran Sasso Science Institute (GSSI), Italy
patrizio.pelliccione@gssi.it

ABSTRACT

The Systematic Literature Review (SLR) presented in this document is part of the research conducted for the *"TwinArch: A Digital Twin Reference Architecture"* paper. The review examines the current state of Digital Twin architectures, with a focus on how researchers design and document them. The discussion of architectural perspectives is guided by the ISO 42010 standard and the Software Engineering Institute's (SEI) Views&Beyond methodology for designing and documenting software system architectures. Please ensure that the TwinArch work is properly cited when referencing these findings. For access to the replication package, visit: https://alessandrasomma28.github.io/twinarch/slr.html .

KEYWORDS

Systematic Literature Review, Digital Twin, Software Architecture, Views&Beyond

1 MOTIVATION

Digital Twins (DTs) have emerged as a transformative technology across multiple domains, including manufacturing, healthcare, smart cities, and transportation. By creating digital replicas of physical entities, DTs enable advanced monitoring, simulation, and predictive capabilities that drive innovation and optimize processes. Despite the growing body of research, existing studies on DTs have primarily focused on their definitions, characteristics, and applications, with comparatively limited attention given to their underlying architectural design.

Early work by Harper *et al.* [24] examined DT capabilities and operations but did not explore the architectural structures that support them. Similarly, Jones *et al.* [27] conducted a systematic literature review (SLR) identifying key characteristics of DTs without addressing architectural aspects. While recent surveys by Minerva et al. [34], Stefan *et al.* [33], and Ofosu *et al.* [36] have contributed to understanding DT architectures in specific contexts, these studies often remain limited to specific domains or focus on high-level classifications without offering standardized frameworks for design and documentation. Studies such as those by Ferko *et al.* [21] have shown that existing DT architectures often fail to fully adhere to relevant standards, such as ISO 23247, underscoring the need for a more structured approach.

To address these challenges, this Systematic Literature Review (SLR) investigates the state-of-the-art in DT architecture design and documentation. The review applies the *ISO 42010* standard [26] and the Software Engineering Institute's (SEI) *Views&Beyond* methodology [7] to evaluate how current DT architectures are designed, documented, and aligned with established software architecture practices. This structured evaluation is essential for identifying best practices, recurring architectural patterns, and existing gaps, providing a foundation for the development of standardized approaches.

This SLR is conducted as part of the *"TwinArch: A Digital Twin Reference Architecture"* work, which aims to design a reference architecture for DTs by extracting key architectural elements from the literature. This review adopts Kitchenham's guidelines for conducting systematic reviews [28] and formulates research questions based on the Goal-Question-Metric (GQM) approach [45]. The research questions investigate key aspects of DT architectural design, including the types of architectural views, styles, and notations employed in the selected primary studies. By systematically analyzing these dimensions, we aim to identify the architectural elements typically used in Digital Twin architectures that will be the ground for the TwinArch design (*please if using these findings remember to cite the TwinArch work*). This review identifyed also recurring patterns, gaps, and limitations that could guide future research and practice.

2 STUDY DESIGN

This section delves into the steps executed for conducting the Systematic Literature Review according to Kitchenam's guidelines [28].

2.1 Research Goal

This study aims to characterize the state-of-the-art of Digital Twin architectures with respect to the ISO 42010 standard and the SEI View & Beyond method. The research investigates how current architectural models are designed and documented, adopting the Goal-Question-Metric (GQM) approach [45] to define its core research objective.

Purpose	Characterize
Issue	Digital Twin architectures for
Object	understanding their design and documentation
Viewpoint	from researchers and developers perspective.

2.2 Research Questions

To achieve these goals, we formulated four Research Questions (RQs).

RQ1: *How are the primary studies categorized according to the SEI architectural views?*
Rationale: This research question aims to identify which SEI architectural views are most commonly used to guide the documentation of Digital Twin architectures. Understanding the alignment between the architectural views presented in the primary studies and the ones advocated by SEI (e.g., Module, Component & Connector, and Allocation views) is critical to assessing the adherence of DT architectures to standardized documentation practices.

RQ2: *How are the primary studies classified based on the SEI architectural styles?*
Rationale: The goal of this research question is to determine which architectural styles are most frequently employed and to identify the core elements and relationships commonly found in DT architectures. Architectural styles play a key role in defining system organization, communication, and constraints, and different views may require distinct styles tailored to specific purposes. By examining how these styles are applied, we can uncover patterns in element selection, common relationships, and constraints, offering insights into the design decisions made in the development of DT architectures.

RQ3: *What type of notation (informal, semi-formal, or formal) is mostly used to document DT architectures?*
Rationale: this question focuses on assessing the types of notations and modeling languages employed for documenting DT architectures. Notations can be classified into three categories: informal (e.g., textual descriptions or sketches), semi-formal (e.g., UML diagrams), and formal (e.g., Architecture Description Languages). Understanding the preferred documentation approaches within the field is essential for evaluating how well these notations convey key architectural aspects. By identifying trends in notation usage, we aim to assess how they influence the clarity, precision, and consistency of DT architecture documentation.

2.3 Initial search

The research questions were broken down into facets, and a list of synonyms, abbreviations, and alternative spellings was created for each term following the guidelines of Kitchenham and Charters [28]. Additional terms were derived from subject headings used in journals and scientific databases. The search string was constructed by combining terms using conjunctions (AND) and disjunctions (OR) for each main concept. To validate its effectiveness, the search string was tested using a set of five control studies [5, 6, 13, 47, 50, 51] previously identified by one of the authors. The accuracy of the search string was evaluated by verifying if these control studies were successfully retrieved when applied to the Scopus search engine. The finalized search string is the following:

("Digital Twin" **OR** "Virtual Twin" **OR** "Digital Replica" **OR** "Virtual Replica") **AND** (Architect* **OR** Framework **OR** Platform **OR** Document* **OR** View **OR** Style)

The selection process is illustrated in Figure 1. It began by executing the search string in the Scopus database on June 2024 that resulted in 5508 studies. The selection process we followed is shown in Figure 1, representing the steps executed and their results. In the *automated search* step, a researcher executed the search string in the electronic database Scopus[1] on June 2024. The search string was filtered by title and abstract and configured for retrieving studies published after 2019 in the areas of computer science and engineering, yielding 5508 studies.

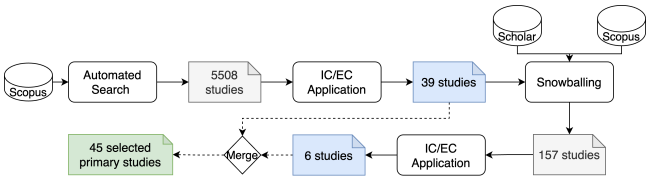

Figure 1: Systematic Literature Review Process.

2.4 Application of selection criteria

To support the selection of retrieved studies by the automated search strategy, we defined inclusion and exclusion criteria to include or discard a manuscript.

Inclusion Criteria. To be included in our analysis, a study must accomplish all the following inclusion criteria:

IC1. The study is related to the topic under investigation, i.e., defining and documenting Digital Twin architectures.
IC2. The study is written in English.
IC3. The study is peer-reviewed.
IC4. The study is a primary study.
IC5. The study has been published after 2019.
Rationale: Despite Digital Twins have over a 20-year history, publications on DT architectures have significantly increased since 2019 [20].
IC6. The study was published in journals or conference proceedings.
IC7. The study was published in high-ranking venues.

Exclusion Criteria. We excluded a publication if it satisfies at least one of the five exclusion criteria listed below:

EC1. The study is an earlier version of a more recent or complete version that has been identified.
EC2. The study treats digital twins as software characterized solely by simulated models.
EC3. The study conflates the concept of digital twins with the Metaverse.
Rationale: The Metaverse is a virtual world designed for social interactions and immersive experiences, while DTs replicate physical assets for operational use.
EC4. The study confuses the modeling aspects of Digital Twins with artificial intelligence models.
Rationale: Artificial intelligence models are tailored algorithms for specific analyses, whereas DTs offer holistic system representation integrating AI for analytics and prediction.

[1] https://www.scopus.com/

In the *Application of Inclusion and Exclusion Criteria* step, two researchers independently analyzed each study's title and abstract and, if necessary, read the full text, applying the IC/EC previously listed. Each manuscript was categorized as "yes", "no" or "doubt". Publications that received two "yes" votes were included, while studies with two "no" votes were excluded. The researchers discussed the remaining manuscripts to reach a consensus. At the end of this step, 5469 studies were excluded and the remaining 39 manuscripts were included in the final set of selected papers

2.5 Snowballing

To mitigate the risk of missing relevant literature, we further executed the complementary *snowballing search* step. For each of the 39 papers, we applied backward snowballing on Scopus to identify referenced studies. For forward snowballing, we used Google Scholar[2] to automate the retrieval of manuscripts citing selected ones. An author aggregated all the retrieved studies and duplicates were automatically removed, resulting in 157 papers. The inclusion/exclusion criteria step was repeated for these 157 papers, resulting in 6 additional studies. Finally, the two sets of papers were merged, resulting in the set of *45 selected primary studies* listed in Table 3.

Table 1: Data Extraction Form Design.

Research Question	*Evidence to be extracted*
RQ1: How are the primary studies categorized according to the SEI architectural views?	List of the extracted architectural views.
RQ2: How are the primary studies classified based on the SEI architectural styles?	List of the extracted architectural styles, elements and relations.
RQ3: What type of notation (informal, semi-formal, formal) is mostly used to document DT architectures?	List of the extracted notations used to document the proposal.

2.6 Data Extraction

To facilitate the data extraction process, we created the data extraction form detailed in Table 1. This form was utilized to report the evidence extracted from the selected papers to answer the research questions.

3 DATA EXTRACTION AND ANALYSIS

Two researchers used the extraction form shown in Table 1 to extract evidence from the selected primary studies for answering the research questions. Finally, they shared their findings with other authors and discussed discrepancies to reach a final consensus on the extracted data.

3.1 RQ1. How are the primary studies classified according to the SEI architectural views?

To identify the architectural views that have been adopted to document Digital Twin architectures, a thorough analysis was conducted on sentences extracted from the selected primary studies to answer this research question. Figure 2 highlights the number of publications that present one or more architectural views, revealing that 41 out of 45 studies (91%) use a single architectural view to document Digital Twin architectures, while only four studies [11–13, 53] apply multiple views. Specifically, these four studies combine two views to document the proposed architecture.

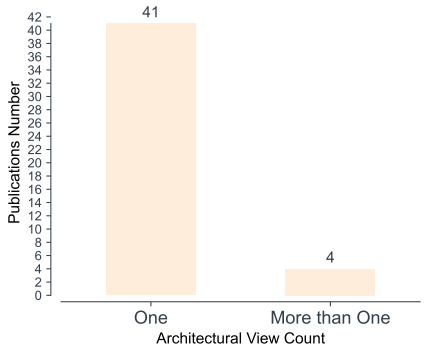

Figure 2: View count.

Figure 3 classifies these studies according to SEI's module, component and allocation views [7]. It shows that 66% (30 out of 45) of studies rely on the module view, mainly for documenting Digital Twins at a high level of detail. Another 29% (13 out of 45) use the Component-and-Connector view to depict development aspects, such as the interaction of components at runtime for implementing specific features. Only 13% (6 out of 45) utilize the allocation view to map architectural elements to hardware devices or Commercial-off-the-shelf (COTS) software components.

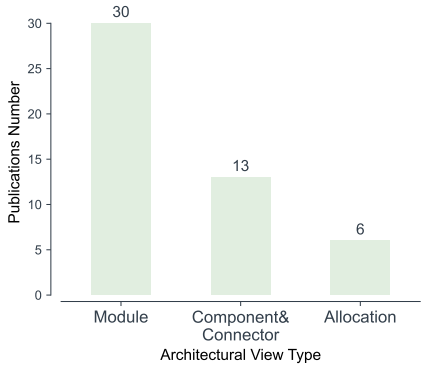

Figure 3: View type distribution.

Figure 4 illustrates the occurrence of categories like basics, software interface, and behavior which are the beyond aspects defined by the SEI. Figure indicates that 17% (8 out of 45) of studies include basic elements such as context diagrams to define the scope of the Digital Twin. Additionally, 31% (14 out of 45) describe software interfaces to explain interactions between components, including data flows or API calls. Finally, 86% (39 out of 45) document behavioral aspects by detailing execution steps through events and actions. These include performing specific use cases, such as wind turbine predictive maintenance [1] or a Smart City Digital Twin [2].

[2]https://scholar.google.com/

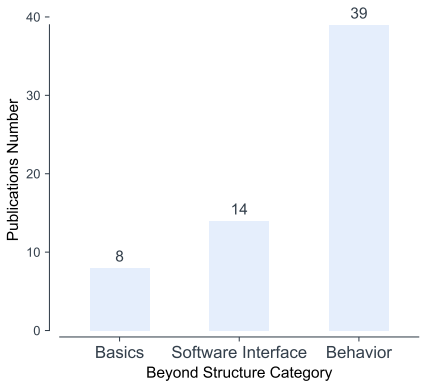

Figure 4: Beyond aspects distribution.

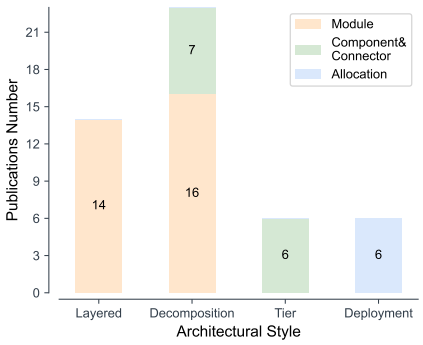

Figure 5: Architectural styles distribution.

This analysis highlights the significant need to describe complex architectures like Digital Twins at a high level of detail, to document fundamental elements and their interrelationships. Furthermore, the results suggest that at lower abstraction levels, authors document specific functionalities by illustrating the interactions between software components. When detailing the implementation of Digital Twins for specific application domains or technologies, the Allocation View proved to be an effective model for design.

3.2 RQ2. How are the primary studies classified based on the SEI architectural styles?

To answer RQ2, we analyzed the evidence collected from the selected primary studies regarding the architectural styles used in modeling the architectural views of Digital Twin documentation. The architectural styles we consider are those defined by the SEI for each type of view [7]. Since SEI specifies multiple styles per view, we focus on the specific styles applied within DT architectures.

The histograms in Figure 5 illustrate the number of architectural styles applied to each view type across the selected studies. It can be observed that the *layered style* is used in 31% (14 out of 45) of the studies. This style is exclusively applied within module views, where it helps demonstrate the separation of concerns in the proposed architectures. For instance, authors distinguish between a data manager class, responsible for managing the data flow within the Digital Twin, and a digital model class, which focuses on modeling the structural and behavioral aspects of the physical twin.

The layered style not only highlights this separation of concerns but also shows how architectural elements are organized into distinct layers (e.g. modeling layer, data ingestion layer) and how they interact through relationships such as data flows [41], dependencies [23], or communication directions [56]. By dividing the system into layers with clearly defined responsibilities, this style simplifies the management of complex architectures by isolating different aspects of the system, making it easier to design, maintain, and scale.

The *decomposition style* is applied in both module and component views. Our analysis revealed that this style is commonly used to enhance the understanding of proposed architectures. By breaking the system into smaller, manageable parts, it facilitates better comprehension of the architecture and its functionalities for both developers and stakeholders. For example, in Digital Twin architectures, decomposition may be used to separate the system into core modules, such as a data acquisition module, a processing module, and a visualization module, each focusing on a distinct responsibility.

The *tier style* is applied in 7 out of 45 studies and, as expected, is used exclusively in Component&Connector views, where it is typically employed to model runtime interactions. Similar to the layered style in module views, the tier style also provides a form of separation of concerns. It divides the system into distinct tiers, each handling a specific aspect of the application, such as presentation, business logic, and data access [2, 3]. In Digital Twin architectures, we observed examples where elements are divided into tiers, such as a sensor input tier for collecting data from the physical environment, a processing tier for analyzing the data, and an output tier for reporting results or triggering actions.

Lastly, the *deployment style* is applied in 6 out of 45 studies and is used within allocation views. This style provides a clear representation of the physical distribution of software components across hardware nodes and defines resource allocation by mapping software components to specific hardware resources [1, 25]. For instance, in a Smart City Digital Twin, a deployment style might show how traffic monitoring components are distributed across roadside devices, edge servers, and central control systems, ensuring efficient resource utilization and data flow.

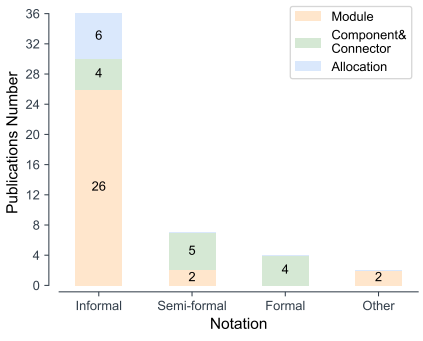

Figure 6: Architectural views notation.

Table 2: Limitations identified in the selected primary studies.

ID	Name	Description
L1.1	Unstructured documentation	The proposed architecture lacks a structured approach with distinct architectural views.
L1.2	Misused elements for views	The proposed architecture mixes elements meant for distinct architectural views.
L1.3	Unclear model elements	The model elements of the proposed architecture lack clear definitions.
L1.4	Unclear model element responsibilities	In the proposed architecture, the responsibilities assigned to model elements are poorly described.
L1.5	Informal notations only	The architecture documentation relies solely on informal notation.
L1.6	Lack of dynamic interaction illustration	In the proposed architecture, the behavior relies solely on case study descriptions, not dynamic interactions.
L1.7	Lack of data styles	The proposed architecture neglects data-related architectural styles essential for integrating DT data.
L2.1	Lack of empirical evaluation	The proposed architecture remains theoretical without empirical evaluation.
L2.2	Data not publicly accessible	The experimental data or the design or the implementation of the proposed architecture are not publicly accessible.
L3.1	Domain-specific	The proposed architecture primarily addresses specific challenges within application domains.
L3.2	Platform-specific	The proposed architecture presents only the deployment of widely adopted DT technologies.

3.3 RQ3. What type of notation (informal, semi-formal, or formal) is mostly used to document DT architectures?

To address this research question, we analyzed the selected studies to identify the types of notations used to describe architectural documentations. Figure 6 illustrates the distribution of notations used to represent elements and relationships within the architectural views for documenting Digital Twin architectures.

The results show that *informal notations*, particularly box-and-arrow diagrams, are the most commonly used (36 out of 45 studies). These notations typically represent Digital Twin elements as boxes and illustrate relationships such as information flow, control flow, or dependencies through arrows. We also found that only a few studies use *semi-formal notations* (7 out of 45) or formal notations (4 out of 45).

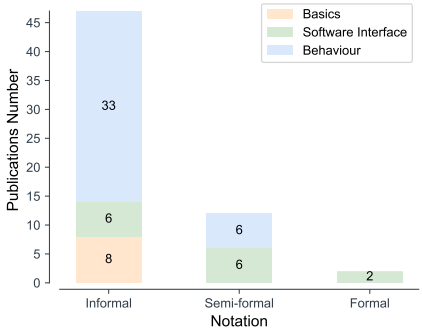

Figure 7: Beyond aspects notation.

Among the semi-formal notations, UML is the most frequently used modeling language. Specifically, within the module view, UML class and object diagrams (e.g., in study [44]) and package diagrams (e.g., in study [53]) are commonly employed. For the C&C views, Digital Twin components are typically modeled using UML component diagrams (e.g., study [13]). *Formal notations*, such as MontiArc, are used exclusively in manuscripts presenting the component-and-connector view [3, 4, 38, 47]. Additionally, we observed that taxonomies and other generic languages are used in very few studies to describe module views (e.g., studies [5] and [18]).

We also examined the types of notations used to represent beyond aspects of Digital Twin architectures. Figure 7 presents the distribution of notation types used to model various beyond aspects. Similar to the architectural views, informal notations dominate and are used to model all beyond aspects. Box-and-arrow models are frequently employed to depict context diagrams for basic aspects (e.g., studies [1, 25, 49]), communication points for software interface aspects (e.g., studies [23, 41]), and use case architectures for behavioral documentation (e.g., studies [9, 15]).

Semi-formal notations are also used in specific contexts, such as modeling context diagrams and interfaces using UML lollipop graphical notation. Behavioral aspects are often documented through diagrams such as UML sequence diagrams or UML activity diagrams (e.g., studies [3, 10]). Only two studies use a more formal approach to describe interfaces, employing a Java-like syntax (e.g., studies [4, 47]).

The findings indicate a strong preference for intuitive and visual representations in Digital Twin architecture documentation, with informal notations (such as text and box-and-arrow diagrams) being the most commonly used. This preference can be attributed to their simplicity and their ability to convey complex ideas in a clear and expressive manner. However, the analysis also highlights a growing need for more structured modeling languages, particularly when representing specific aspects of the architecture, because informal notations suffer of the lack of universally recognized meanings.

4 DISCUSSION

We analyzed the selected primary studies to identify the limitations associated with the design and documentation of Digital Twin architectures. These limitations were classified into three main categories:

- lack of structured architectural documentation (*L1.x*);
- lack of experimentation and unavailable design or implementation artifacts (*L2.x*);
- non-generalizable design (*L3.x*).

Table 2 lists the identified limitation in the literature review, Figure 8 displays the distribution of limitations across the selected studies. The most common limitation is the lack of structured multi-view architectural documentation ($L1.1$), observed in 38 out of 45 studies (e.g., [48, 56]). This gap often results in confusion and leads to the misuse of architectural elements across different views ($L1.2$), noted in 29 studies (e.g., [50, 55]).

Additionally, unclear definitions of model elements ($L1.3$) affect 22 out of 45 studies (e.g., [11, 42]), and ambiguities in assigning responsibilities to these elements ($L1.4$) are present in 44% of the

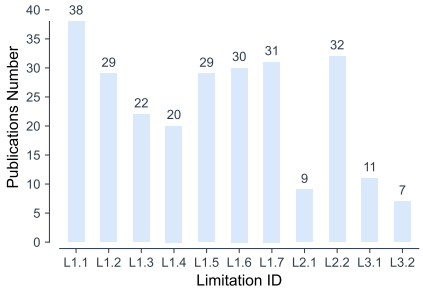

Figure 8: Limitations distribution.

studies (e.g., [29, 53]). These ambiguities hinder the understanding of how components interrelate within the architecture. A lack of proper documentation of system behavior further exacerbates the issue, with 45% of the studies failing to illustrate dynamic interactions ($L1.6$) effectively (e.g., [43, 52]).

Study Replicability. To replicate the findings, please refer to https://alessandrasomma28.github.io/twinarch/slr.html where the replication package with Scopus query results, manuscript selection and selected studies data extraction and analysis is available.

REFERENCES

[1] Ibrahim Abdullahi, Stefano Longo, and Mohammad Samie. 2024. Towards a Distributed Digital Twin Framework for Predictive Maintenance in Industrial Internet of Things (IIoT). *Sensors* 24, 8 (2024). https://doi.org/10.3390/s24082663

[2] Lorenzo Adreani, Pierfrancesco Bellini, Marco Fanfani, Paolo Nesi, and Gianni Pantaleo. 2024. Smart City Digital Twin Framework for Real-Time Multi-Data Integration and Wide Public Distribution. *IEEE Access* 12 (2024), 76277–76303. https://doi.org/10.1109/ACCESS.2024.3406795

[3] Pascal Bibow, Manuela Dalibor, Christian Hopmann, Ben Mainz, Bernhard Rumpe, David Schmalzing, Mauritius Schmitz, and Andreas Wortmann. 2020. *Model-Driven Development of a Digital Twin for Injection Molding.* Springer International Publishing, Cham, 85–100. https://doi.org/10.1007/978-3-030-49435-3_6

[4] Tim Bolender, Gereon Burvenich, Manuela Dalibor, Bernhard Rumpe, and Andreas Wortmann. 2021. Self-Adaptive Manufacturing with Digital Twins. In *2021 International symposium on software engineering for adaptive and self-managing systems (SEAMS).* IEEE, 156–166. https://doi.org/10.1109/SEAMS51251.2021.00029

[5] Hugh Boyes and Tim Watson. 2022. Digital twins: An analysis framework and open issues. *Computers in Industry* 143 (2022), 103763. https://doi.org/10.1016/j.compind.2022.103763

[6] Tobias Brockhoff, Malte Heithoff, István Koren, Judith Michael, Jérôme Pfeiffer, Bernhard Rumpe, Merih Seran Uysal, Wil M. P. Van Der Aalst, and Andreas Wortmann. 2021. Process Prediction with Digital Twins. In *2021 ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C).* 182–187. https://doi.org/10.1109/MODELS-C53483.2021.00032

[7] Paul Clements, Felix Bachmann, Len Bass, David Garlan, James Ivers, Reed Little, Paulo Merson, Robert Nord, and Judith Stafford. 2010. *Documenting Software Architectures: Views and Beyond.* Addison-Wesley.

[8] Javier Conde, Andres Munoz-Arcentales, Álvaro Alonso, Gabriel Huecas, and Joaquín Salvachúa. 2022. Collaboration of Digital Twins Through Linked Open Data: Architecture With FIWARE as Enabling Technology. *IT Professional* 24, 6 (2022), 41–46. https://doi.org/10.1109/MITP.2022.3224826

[9] Javier Conde, Andrés Munoz-Arcentales, Álvaro Alonso, Sonsoles López-Pernas, and Joaquín Salvachúa. 2022. Modeling Digital Twin Data and Architecture: A Building Guide With FIWARE as Enabling Technology. *IEEE Internet Computing* 26, 3 (2022), 7–14. https://doi.org/10.1109/MIC.2021.3056923

[10] Alessandro Costantini, Giuseppe Di Modica, Jean Christian Ahouangonou, Doina Cristina Duma, Barbara Martelli, Matteo Galletti, Marica Antonacci, Daniel Nehls, Paolo Bellavista, Cedric Delamarre, and Daniele Cesini. 2022. IoTwins: Toward Implementation of Distributed Digital Twins in Industry 4.0 Settings. *Computers* 11, 5 (2022). https://doi.org/10.3390/computers11050067

[11] Salvador Cuñat Negueroles, Raúl Reinosa Simón, Matilde Julián, Andreu Belsa, Ignacio Lacalle, Raúl S-Julián, and Carlos E. Palau. 2024. A Blockchain-based Digital Twin for IoT deployments in logistics and transportation. *Future Generation Computer Systems* 158 (2024), 73–88. https://doi.org/10.1016/j.future.2024.04.011

[12] Alessandra De Benedictis, Francesco Flammini, Nicola Mazzocca, Alessandra Somma, and Francesco Vitale. 2023. Digital Twins for Anomaly Detection in the Industrial Internet of Things: Conceptual Architecture and Proof-of-Concept. *IEEE Transactions on Industrial Informatics* 19, 12 (2023), 11553–11563. https://doi.org/10.1109/TII.2023.3246983

[13] Alessandra De Benedictis, Nicola Mazzocca, Alessandra Somma, and Carmine Strigaro. 2023. Digital Twins in Healthcare: An Architectural Proposal and Its Application in a Social Distancing Case Study. *IEEE Journal of Biomedical and Health Informatics* 27, 10 (2023), 5143–5154. https://doi.org/10.1109/JBHI.2022.3205506

[14] Lorenzo De Donato, Ruth Dirnfeld, Alessandra Somma, Alessandra De Benedictis, Francesco Flammini, Stefano Marrone, Mehdi Saman Azari, and Valeria Vittorini. 2023. Towards AI-assisted digital twins for smart railways: preliminary guideline and reference architecture. *Journal of Reliable Intelligent Environments* 9, 3 (2023), 303 – 317. https://doi.org/10.1007/s40860-023-00208-6

[15] Pavlos Eirinakis, Stavros Lounis, Stathis Plitsos, George Arampatzis, Kostas Kalaboukas, Klemen Kenda, Jinzhi Lu, Jože M. Rožanec, and Nenad Stojanovic. 2022. Cognitive Digital Twins for Resilience in Production: A Conceptual Framework. *Information* 13, 1 (2022). https://doi.org/10.3390/info13010033

[16] Haya Elayan, Moayad Aloqaily, and Mohsen Guizani. 2021. Digital Twin for Intelligent Context-Aware IoT Healthcare Systems. *IEEE Internet of Things Journal* 8, 23 (2021), 16749–16757. https://doi.org/10.1109/JIOT.2021.3051158

[17] Romina Eramo, Francis Bordeleau, Benoit Combemale, Mark van den Brand, Manuel Wimmer, and Andreas Wortmann. 2022. Conceptualizing Digital Twins. *IEEE Software* 39, 2 (2022), 39–46. https://doi.org/10.1109/MS.2021.3130755

[18] John Ahmet Erkoyuncu, Iñigo Fernández del Amo, Dedy Ariansyah, Dominik Bulka, Rok Vrabič, and Rajkumar Roy. 2020. A design framework for adaptive digital twins. *CIRP Annals* 69, 1 (2020), 145–148. https://doi.org/10.1016/j.cirp.2020.04.086

[19] Nafise Eskandani and Sten Grüner. 2023. Cloud-Native Architecture for Mixed File-Based and API-Based Digital Twin Exchange. In *Software Architecture.*

Springer Nature Switzerland, Cham, 292–299.

[] Enxhi Ferko, Alessio Bucaioni, and Moris Behnam. 2022. Architecting Digital Twins. *IEEE Access* 10 (2022), 50335–50350. https://doi.org/10.1109/ACCESS.2022.3172964

[21] Enxhi Ferko, Alessio Bucaioni, Patrizio Pelliccione, and Moris Behnam. 2023. Standardisation in Digital Twin Architectures in Manufacturing. In *2023 IEEE 20th International Conference on Software Architecture (ICSA)*. 70–81. https://doi.org/10.1109/ICSA56044.2023.00015

[22] Moritz Glatt, Chantal Sinnwell, Li Yi, Sean Donohoe, Bahram Ravani, and Jan C. Aurich. 2021. Modeling and implementation of a digital twin of material flows based on physics simulation. *Journal of Manufacturing Systems* 58 (2021), 231–245. https://doi.org/10.1016/j.jmsy.2020.04.015 Digital Twin towards Smart Manufacturing and Industry 4.0.

[23] Sri Nikhil Gupta Gourisetti, Sraddhanjoli Bhadra, David Jonathan Sebastian-Cardenas, Md Touhiduzzaman, and Osman Ahmed. 2023. A Theoretical Open Architecture Framework and Technology Stack for Digital Twins in Energy Sector Applications. *Energies* 16, 13 (2023). https://doi.org/10.3390/en16134853

[24] K. Eric Harper, Christopher Ganz, and Somayeh Malakuti. 2019. Digital Twin Architecture and Standards. *IIC Journal of Innovation* (11 2019), 1–12.

[25] Naser Hossein Motlagh, Martha Arbayani Zaidan, Lauri Lovén, Pak Lun Fung, Tuomo Hänninen, Roberto Morabito, Petteri Nurmi, and Sasu Tarkoma. 2024. Digital Twins for Smart Spaces—Beyond IoT Analytics. *IEEE Internet of Things Journal* 11, 1 (2024), 573–583. https://doi.org/10.1109/JIOT.2023.3287032

[26] ISO/IEC/IEEE 42010:2022(E) 2022. *International Standard for Software, systems and enterprise–Architecture description*. Standard. IEEE/ISO/IEC. 1–74 pages. https://doi.org/10.1109/IEEESTD.2022.9938446

[27] David Jones, Chris Snider, Aydin Nassehi, Jason Yon, and Ben Hicks. 2020. Characterising the Digital Twin: A systematic literature review. *CIRP Journal of Manufacturing Science and Technology* 29 (2020), 36–52. https://doi.org/10.1016/j.cirpj.2020.02.002

[28] Barbara Kitchenham, O Pearl Brereton, David Budgen, Mark Turner, John Bailey, and Stephen Linkman. 2009. Systematic literature reviews in software engineering–a systematic literature review. *Information and software technology* 51, 1 (2009), 7–15.

[29] Daniel Lehner, Jérôme Pfeiffer, Erik-Felix Tinsel, Matthias Milan Strljic, Sabine Sint, Michael Vierhauser, Andreas Wortmann, and Manuel Wimmer. 2022. Digital Twin Platforms: Requirements, Capabilities, and Future Prospects. *IEEE Software* 39, 2 (2022), 53–61. https://doi.org/10.1109/MS.2021.3133795

[30] Aurora Macéas, Elena Navarro, Carlos E. Cuesta, and Uwe Zdun. 2023. Architecting Digital Twins Using a Domain-Driven Design-Based Approach*. In *2023 IEEE 20th International Conference on Software Architecture (ICSA)*. 153–163. https://doi.org/10.1109/ICSA56044.2023.00022

[31] Montaser Mahmoud, Concetta Semeraro, Mohammad Ali Abdelkareem, and Abdul Ghani Olabi. 2024. Designing and prototyping the architecture of a digital twin for wind turbine. *International Journal of Thermofluids* 22 (2024), 100622. https://doi.org/10.1016/j.ijft.2024.100622

[32] Somayeh Malakuti, Johannes Schmitt, Marie Platenius-Mohr, Sten Grüner, Ralf Gitzel, and Prerna Bihani. 2019. A Four-Layer Architecture Pattern for Constructing and Managing Digital Twins. In *Software Architecture*. Springer International Publishing, Cham, 231–246.

[33] Stefan Mihai, Mahnoor Yaqoob, Dang V. Hung, William Davis, Praveer Towakel, Mohsin Raza, Mehmet Karamanoglu, Balbir Barn, Dattaprasad Shetve, Raja V. Prasad, Hrishikesh Venkataraman, Ramona Trestian, and Huan X. Nguyen. 2022. Digital Twins: A Survey on Enabling Technologies, Challenges, Trends and Future Prospects. *IEEE Communications Surveys & Tutorials* 24, 4 (2022), 2255–2291. https://doi.org/10.1109/COMST.2022.3208773

[34] Roberto Minerva, Gyu Myoung Lee, and Noël Crespi. 2020. Digital Twin in the IoT Context: A Survey on Technical Features, Scenarios, and Architectural Models. *Proc. IEEE* 108, 10 (2020), 1785–1824. https://doi.org/10.1109/JPROC.2020.2998530

[35] Nader Mohamed, Jameela Al-Jaroodi, Imad Jawhar, and Nader Kesserwan. 2023. Leveraging Digital Twins for Healthcare Systems Engineering. *IEEE Access* 11 (2023), 69841–69853. https://doi.org/10.1109/ACCESS.2023.3292119

[36] Rosemary Ofosu, Amin Hosseinian-Far, and Dilshad Sarwar. 2022. *Digital Twin Technologies, Architecture, and Applications: A Comprehensive Systematic Review and Bibliometric Analysis*. Springer International Publishing, Cham, 105–142. https://doi.org/10.1007/978-3-030-98225-6_5

[37] Hector D. Perez, John M. Wassick, and Ignacio E. Grossmann. 2022. A digital twin framework for online optimization of supply chain business processes. *Computers & Chemical Engineering* 166 (2022), 107972. https://doi.org/10.1016/j.compchemeng.2022.107972

[38] Jérôme Pfeiffer, Daniel Lehner, Andreas Wortmann, and Manuel Wimmer. 2023. Towards a Product Line Architecture for Digital Twins. In *2023 IEEE 20th International Conference on Software Architecture Companion (ICSA-C)*. 187–190. https://doi.org/10.1109/ICSA-C57050.2023.00049

[39] Anro Redelinghuys, Anton Basson, and Karel Kruger. 2020. A six-layer architecture for the digital twin: a manufacturing case study implementation. *Journal of Intelligent Manufacturing* 31 (08 2020). https://doi.org/10.1007/s10845-019-01516-6

[40] Luis F. Rivera, Miguel Jiménez, Norha M. Villegas, Gabriel Tamura, and Hausi A. Müller. 2022. The Forging of Autonomic and Cooperating Digital Twins. *IEEE Internet Computing* 26, 5 (2022), 41–49. https://doi.org/10.1109/MIC.2021.3051902

[41] Julia Robles, Cristian Martín, and Manuel Díaz. 2023. OpenTwins: An open-source framework for the development of next-gen compositional digital twins. *Computers in Industry* 152 (2023), 104007. https://doi.org/10.1016/j.compind.2023.104007

[42] Carlos Rodríguez-Alonso, Iván Pena-Regueiro, and Óscar García. 2024. Digital Twin Platform for Water Treatment Plants Using Microservices Architecture. *Sensors* 24, 5 (2024). https://doi.org/10.3390/s24051568

[43] Hassan Sartaj, Shaukat Ali, Tao Yue, and Kjetil Moberg. 2024. Model-based digital twins of medicine dispensers for healthcare IoT applications. *Software: Practice and Experience* 54, 6 (2024), 1172–1192. https://doi.org/10.1002/spe.3311

[44] Greyce N. Schroeder, Charles Steinmetz, Ricardo Nagel Rodrigues, Renato Ventura Bayan Henriques, Achim Rettberg, and Carlos Eduardo Pereira. 2021. A Methodology for Digital Twin Modeling and Deployment for Industry 4.0. *Proc. IEEE* 109, 4 (2021), 556–567. https://doi.org/10.1109/JPROC.2020.3032444

[45] Rini Solingen, Vic Basili, Gianluigi Caldiera, and Dieter Rombach. 2002. *Goal Question Metric (GQM) Approach*. https://doi.org/10.1002/0471028959.sof142

[46] Alessandra Somma, Alessandra De Benedictis, Christiancarmine Esposito, and Nicola Mazzocca. 2024. The convergence of Digital Twins and Distributed Ledger Technologies: A systematic literature review and an architectural proposal. *Journal of Network and Computer Applications* 225 (2024), 103857. https://doi.org/10.1016/j.jnca.2024.103857

[47] Patrick Spaney, Steffen Becker, Robin Ströbel, Jürgen Fleischer, Soraya Zenhari, Hans-Christian Möhring, Ann-Kathrin Splettstößer, and Andreas Wortmann. 2023. A Model-Driven Digital Twin for Manufacturing Process Adaptation. In *2023 ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C)*. 465–469. https://doi.org/10.1109/MODELS-C59198.2023.00081

[48] Charles Steinmetz, Greyce N. Schroeder, Ricardo N. Rodrigues, Achim Rettberg, and Carlos E. Pereira. 2022. Key-Components for Digital Twin Modeling With Granularity: Use Case Car-as-a-Service. *IEEE Transactions on Emerging Topics in Computing* 10, 1 (2022), 23–33. https://doi.org/10.1109/TETC.2021.3131532

[49] Chang Su, Yong Han, Xin Tang, Qi Jiang, Tao Wang, and Qingchen He. 2024. Knowledge-based digital twin system: Using a knowledge-driven approach for manufacturing process modeling. *Computers in Industry* 159-160 (2024), 104101. https://doi.org/10.1016/j.compind.2024.104101

[50] Fei TAO, Xuemin SUN, Jiangfeng CHENG, Yonghuai ZHU, Weiran LIU, Yong WANG, Hui XU, Tianliang HU, Xiaojun LIU, Tingyu LIU, Zheng SUN, Jun XU, Jinsong BAO, Feng XIANG, and Xiaohui JIN. 2024. makeTwin: A reference architecture for digital twin software platform. *Chinese Journal of Aeronautics* 37, 1 (2024), 1–18. https://doi.org/10.1016/j.cja.2023.05.002

[51] Bedir Tekinerdogan and Cor Verdouw. 2020. Systems Architecture Design Pattern Catalog for Developing Digital Twins. *Sensors* 20, 18 (2020). https://doi.org/10.3390/s20185103

[52] Thomas Usländer, Michael Baumann, Stefan Boschert, Roland Rosen, Olaf Sauer, Ljiljana Stojanovic, and Jan Christoph Wehrstedt. 2022. Symbiotic Evolution of Digital Twin Systems and Dataspaces. *Automation* 3, 3 (2022), 378–399. https://doi.org/10.3390/automation3030020

[53] Raymon van Dinter, Bedir Tekinerdogan, and Cagatay Catal. 2023. Reference architecture for digital twin-based predictive maintenance systems. *Computers & Industrial Engineering* 177 (2023), 109099. https://doi.org/10.1016/j.cie.2023.109099

[54] Rok Vrabič, John Ahmet Erkoyuncu, Maryam Farsi, and Dedy Ariansyah. 2021. An intelligent agent-based architecture for resilient digital twins in manufacturing. *CIRP Annals* 70, 1 (2021), 349–352. https://doi.org/10.1016/j.cirp.2021.04.049

[55] Yang Wenqiang, Xiangyu Bao, Yu Zheng, Lei Zhang, Ziqing Zhang, Zhao Zhang, and Lin Li. 2022. A digital twin framework for large comprehensive ports and a case study of Qingdao Port. *The International Journal of Advanced Manufacturing Technology* 131 (12 2022). https://doi.org/10.1007/s00170-022-10625-1

[56] Haoyu Yu, Dong Yu, Chuting Wang, Yi Hu, and Yue Li. 2023. Edge intelligence-driven digital twin of CNC system: Architecture and deployment. *Robotics and Computer-Integrated Manufacturing* 79 (2023), 102418. https://doi.org/10.1016/j.rcim.2022.102418

Table 3: Selected primary studies.

ID	Title	Ref.
M01	A Blockchain-based Digital Twin for IoT deployments in logistics and transportation	[11]
M02	Knowledge-based digital twin system: Using a knowledge-driven approach for manufacturing process modeling	[49]
M03	Designing and prototyping the architecture of a digital twin for wind turbine	[31]
M04	The convergence of Digital Twins and Distributed Ledger Technologies: A systematic literature review and an architectural proposal	[46]
M05	A digital twin framework for large comprehensive ports and a case study of Qingdao Port	[55]
M06	Towards a Distributed Digital Twin Framework for Predictive Maintenance in Industrial Internet of Things (IIoT)	[1]
M07	Digital Twin Platform for Water Treatment Plants Using Microservices Architecture	[42]
M08	Smart City Digital Twin Framework for Real-Time Multi-Data Integration and Wide Public Distribution	[2]
M09	makeTwin: A reference architecture for digital twin software platform	[50]
M10	Digital Twins for Smart Spaces—Beyond IoT Analytics	[25]
M11	Digital Twins for Anomaly Detection in the Industrial Internet of Things: Conceptual Architecture and Proof-of-Concept	[12]
M12	OpenTwins: An open-source framework for the development of next-gen compositional digital twins	[41]
M13	Digital Twins in Healthcare: An Architectural Proposal and Its Application in a Social Distancing Case Study	[13]
M14	Towards AI-assisted digital twins for smart railways: preliminary guideline and reference architecture	[14]
M15	A Theoretical Open Architecture Framework and Technology Stack for Digital Twins in Energy Sector Applications	[23]
M16	Reference architecture for digital twin-based predictive maintenance systems	[53]
M17	Edge intelligence-driven digital twin of CNC system: Architecture and deployment	[56]
M18	A Model-Driven Digital Twin for Manufacturing Process Adaptation	[47]
M19	Towards a Product Line Architecture for Digital Twins	[38]
M20	Architecting Digital Twins Using a Domain-Driven Design-Based Approach*	[30]
M21	Digital twins: An analysis framework and open issues	[5]
M22	Collaboration of Digital Twins Through Linked Open Data: Architecture With FIWARE as Enabling Technology	[8]
M23	A digital twin framework for online optimization of supply chain business processes	[37]
M24	Symbiotic Evolution of Digital Twin Systems and Dataspaces	[52]
M25	Modeling Digital Twin Data and Architecture: A Building Guide With FIWARE as Enabling Technology	[9]
M26	IoTwins: Toward Implementation of Distributed Digital Twins in Industry 4.0 Settings	[10]
M27	Cognitive Digital Twins for Resilience in Production: A Conceptual Framework	[15]
M28	Digital Twin Platforms: Requirements, Capabilities, and Future Prospects	[29]
M29	Conceptualizing Digital Twins	[17]
M30	Key-Components for Digital Twin Modeling With Granularity: Use Case Car-as-a-Service	[48]
M31	The Forging of Autonomic and Cooperating Digital Twins	[40]
M32	Digital Twin for Intelligent Context-Aware IoT Healthcare Systems	[16]
M33	Self-Adaptive Manufacturing with Digital Twins	[4]
M34	A Methodology for Digital Twin Modeling and Deployment for Industry 4.0	[44]
M35	Process Prediction with Digital Twins	[6]
M36	An intelligent agent-based architecture for resilient digital twins in manufacturing	[54]
M37	Systems Architecture Design Pattern Catalog for Developing Digital Twins	[51]
M38	A six-layer architecture for the digital twin: a manufacturing case study implementation	[39]
M39	A design framework for adaptive digital twins	[18]
M40	Modeling and implementation of a digital twin of material flows based on physics simulation	[22]
M41	A Four-Layer Architecture Pattern for Constructing and Managing Digital Twins	[32]
M42	Model-driven development of a digital twin for injection molding	[3]
M43	Model-based digital twins of medicine dispensers for healthcare IoT applications	[43]
M44	Leveraging Digital Twins for Healthcare Systems Engineering	[35]
M45	Cloud-Native Architecture for Mixed File-Based and API-Based Digital Twin Exchange	[19]