network

MDPI

Review

Architectural Design for Digital Twin Networks

Jorg Wieme *, Mathias Baert and Jeroen Hoebeke

imec—IDLab, Ghent University, 9000 Gent, Belgium; mathias.baert@ugent.be (M.B.); jeroen.hoebeke@ugent.be (J.H.)
* Correspondence: jorg.wieme@ugent.be

Abstract

Digital Twin Networks are advanced digital replicas of physical network infrastructures, offering real-time monitoring, analysis, and optimization capabilities. Despite their potential, the absence of a standardized definition and implementation guidelines complicates practical deployment. The existing literature often lacks clarity on tool selection and implementation specifics. In response, this paper aims to address these challenges by providing a complete guide and reference list of essential tools to implement Digital Twin Networks. Following the current research and work-in-progress from the definition initiative, including our own contributions, we propose a structured approach to Digital Twin Network implementation. Our methodology integrates insights from diverse sources to establish a coherent framework for developers and researchers. By synthesizing insights from the literature and practical experience, we define key components and functionalities critical to Digital Twin Network architecture. Additionally, we highlight challenges inherent to Digital Twin Network implementation and offer strategic approaches and mindsets for addressing them. This includes considerations for scalability, interoperability, real-time communication, data modeling, and security, ensuring a holistic approach to building effective Digital Twin Network systems.

Keywords: Digital Twin Network; Network Digital Twin; network optimization; network management

Academic Editor: Martin Reisslein

Received: 4 June 2025
Revised: 3 July 2025
Accepted: 7 July 2025
Published: 9 July 2025

Citation: Wieme, J.; Baert, M.; Hoebeke, J. Architectural Design for Digital Twin Networks. *Network* **2025**, *5*, 24. https://doi.org/10.3390/network5030024

1. Introduction

Over the past decades, communication protocols have evolved from basic data exchange mechanisms into sophisticated systems tailored for diverse and dynamic network environments [1–3]. Modern protocols now support high-throughput applications and the heterogeneous demands of the Internet of Things (IoT), which involve varied device types, data rates, and reliability requirements. This transformation reflects the growing need for scalable, secure, and adaptable communication frameworks capable of sustaining complex, real-time interactions across increasingly interconnected systems.

As networks are increasingly required to support a growing number of users and applications, the complexity of managing these systems also rises. Network administrators must handle a broad spectrum of configurations, including energy efficiency measures like selectively activating sections of the network and ensuring stringent performance guarantees. This is true for both wireless and wired networks. The added demands for versatility, robust security, and seamless operation further complicate management tasks. Introducing new technologies into these dynamic environments can be costly and potentially disruptive, requiring significant resources to ensure continued efficiency and

reliability. This complexity has driven the evolution of software-managed networking solutions that integrate advanced techniques such as Artificial Intelligence (AI) [4–7] or the Digital Twin Network (DTN) [8–10]. These innovations are essential for navigating the complexities of modern networks and ensuring their scalability and resilience in the face of rapid technological advancements.

A digital twin is a virtual representation of a physical asset, process, or system that exists in the digital world. This concept, originating in academia and prominently adopted in Industry 4.0, involves creating a digital counterpart that mirrors the real-world entity in detail. Digital twins are used extensively to simulate and test processes and services before actual implementation, allowing for predictive analysis, optimization, and troubleshooting in a controlled virtual environment. By mimicking the behavior and performance of their physical counterparts, digital twins enable organizations to enhance efficiency, reduce costs, and accelerate innovation across various sectors, from manufacturing and logistics to healthcare and beyond [11,12].

In recent years, researchers have increasingly looked into the application of digital twin concepts to networks. Thus, the term Network Digital Twin or Digital Twin Network has emerged to describe these virtual replicas that attempt to simulate entire network infrastructures and continuously update themselves with real-time data. This approach allows these digital replicas to accurately mirror the current state and behavior of physical networks. Controlled digital environments provide invaluable insights into network performance, enable the prediction of potential issues, and support thorough testing of various scenarios. This innovative use of digital twins is poised to revolutionize network management practices, offering opportunities to optimize operations, enhance reliability, and drive advancements in communication infrastructure development. The term Network Digital Twin is occasionally used in the literature [13,14] to denote a network comprised of multiple digital twins, though this usage is becoming less common. Instead, the term is increasingly applied to refer to a digital twin specifically developed for modeling and simulating networks, which is the focus of this work.

The definition of Digital Twin Networks remains fluid, reflecting the rapid evolution of this technology. Efforts have been made to establish a standardized concept, currently a work in progress by the Internet Research Task Force (IRTF) [15]. While this ongoing effort provides a foundational framework aligning conceptually with digital twins but tailored specifically for networks, it primarily offers concepts and a reference architecture. Therefore, the focus of this study is to bridge the gap between practical implementation and standardization of DTNs. Building on our expertise in developing and deploying a fully operational DTN for managing a Bluetooth Mesh network [16], the aims of this work are as follows:

- Provide insights into the concepts and reference architecture of DTNs, offering practical guidance for their realization;
- Evaluate existing frameworks against our design, sharing best practices and lessons learned to help readers develop their own DTN implementations;
- Map state-of-the-art ideas to a standardized approach for DTNs;
- Offer a focused discussion on tools and technologies used in research and literature;
- Conceptualize unified security in the DTN by building upon already established, robust security principles.

The paper is structured as follows: Section 2 presents an overview of the existing literature on related work. In Section 3, we look into the DTN architecture as defined by the current standard, providing a foundational understanding. Section 4 outlines our functional implementation of the Digital Twin Network, demonstrating its mapping with the IRTF definition. Subsequently, in Section 5, we describe the implementation details and

design recommendations for constructing a Digital Twin Network. Following this, we will describe implementation strategies to address typical challenges in Section 6. Finally, in Section 7, we conclude with key focus areas and best practices essential for successful DTN implementation.

2. Related Works

The concept of Digital Twin Networks has gained significant attention in recent years, yet many research efforts tend to label them broadly as digital twins for networks without delving deeply into the detailed definitions like those proposed by the work in progress of IRTF. Most studies in the field focus on designing DTN frameworks or conducting surveys, but rarely do they involve real-life implementations or tests of a fully operational DTN. Additionally, discussions on implementation choices are not often addressed in the literature.

For instance, the survey by Wu et al. [8] provides in-depth insights into the key technologies, applications, challenges, and trends of DTNs. They propose a definition of a Digital Twin (DT) comprising three core components: a physical object, its virtual twin, and the continuous mapping between them that ensures their co-evolution. While this aligns with the foundational elements of a DT, their discussion of DTNs remains at a high level and builds upon the general concept of digital twins without addressing the specifics of network implementations in depth.

Similarly, works by Mozo et al. [17,18] propose DTNs enhanced with Machine Learning (ML) and AI for future 6G networks. These studies adhere somewhat to the IRTF definition but focus predominantly on the conceptual integration of ML and AI rather than providing detailed architectural guidelines or implementation strategies.

Other research, like that of Xie et al. [19], introduces a DTN approach using Graph Neural Networks (GNN). However, this work diverges from the IRTF definition, functioning more as an updated GNN framework rather than a full-fledged DTN model as outlined by the IRTF. Similarly, Hui et al. [20] describe a DTN that integrates modeling, simulation, and neural networks, closely mirroring the IRTF structure but without the same precision or focus on real-world applicability.

Zhao et al. [21] mainly explore the design aspects of DTNs, closely following the IRTF's guidelines but not extending them into practical implementation. Their work remains theoretical, offering valuable insights into design considerations without addressing the complexities of deployment.

Other studies take unique approaches that either align partially with or deviate from the IRTF definition. For example, Becattini et al. [22] apply a deep learning methodology to DTNs, devising their own structure that closely resembles the application aspects of the IRTF's model. Conversely, Bellavista et al. [23] refer to their work as a middleware solution for DTNs but focus on a network of digital twins rather than adhering to the IRTF's network-centric perspective.

In a different vein, Song et al. [24] propose a flow emulation framework for DTNs, similar to the IRTF's structure, yet they lack a detailed description of the implementation process. Other notable works include Zhao et al. [25], who define an Intelligent Digital Twin (IDT) that aligns with the IRTF's vision and employs a deep learning approach, and Kherbache et al. [26], who use the Cooja simulator [27] as a modeling tool. The latter is particularly notable for applying DTNs to the Industrial Internet of Things (IIoT), expanding beyond the common focus on 5G, 6G, or Wi-Fi networks.

Lu et al. [28] diverge significantly from the IRTF's framework by splitting their digital twin concept into separate entities for devices and services, thereby not following the structure proposed by the IRTF.

Lastly, Zaki-Hindi et al. [29] attempted to create a work similar to this one, providing a brief and limited discussion of the issues and challenges within DTNs. However, their work does not delve deeply into the subject, nor does it pair these challenges with concrete solutions or draw on extensive practical experience, leaving significant gaps in addressing the complexities of DTNs.

3. Digital Twin Network

The definition of a Digital Twin Network is heavily influenced and inspired by the IRTF draft [15].

3.1. Core Definition

DTNs are designed using four essential components that enable network management. These components, inspired by digital twins in other industries, allow DTNs to analyze, diagnose, emulate, and control real networks. Table 1 summarizes these core elements and their roles in DTN functionality.

Table 1. Core Elements of a Digital Twin Network, inspired by the draft and adapted for a more practical connection.

Element	Description
Data	Historical and/or real-time data about the real network, including configuration, operational state, topology, and process data. This data repository serves as the authoritative source for model development.
Models	Representations of the network, created using data collected from real-world sources. These models are essential for emulation, diagnosis, and understanding network operations. Examples include service models, data models, and knowledge graphs.
Interfaces	Standardized connections that ensure interoperability. One interface links the DTN platform with the real network for real-time data collection and control. Another interface connects the platform with applications, facilitating service requests and access to DTN features.
Mapping	Establishes real-time interactive relationships between the real network and its digital twin, or among multiple DTNs. This can involve continuous one-to-one synchronization or occasional one-to-many data exchanges.

These four core components, Data, Models, Interfaces, and Mapping, form the foundation of DTNs, enabling them to emulate networks and provide essential insights for analysis, diagnosis, and control across the network's lifecycle. Together with optimization algorithms, management strategies, and expert knowledge, these components ensure the support of both repeatability and reproducibility.

Proper DTN implementations also drive efficient resource allocation and configuration, enhancing overall performance. They streamline troubleshooting and maintenance by enabling effective issue diagnosis. Moreover, DTNs are invaluable for planning and deployment, allowing for the simulation of new designs and configurations to evaluate their impact before implementation.

3.2. Reference Architecture

Based on the definitions provided in previous section, a Digital Twin Network architecture is outlined in Figure 1, consisting of three main layers, each described below.

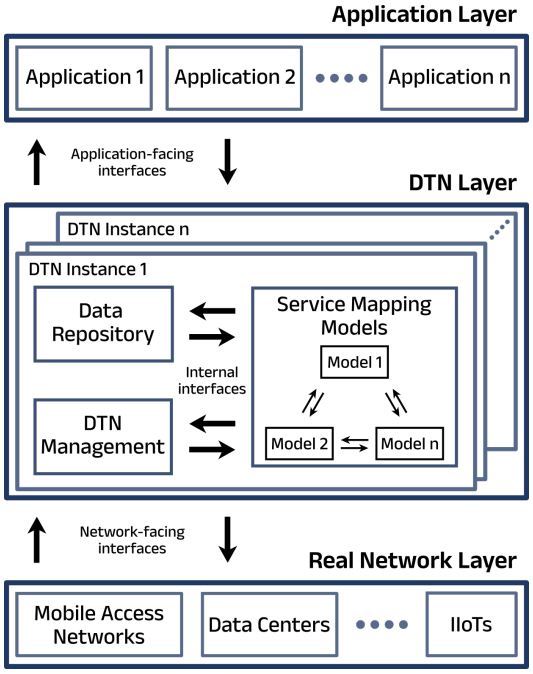

Figure 1. Graphical representation of the reference architecture for a Digital Twin Network

3.2.1. Real Network Layer

This layer includes the actual network components, consisting of physical and virtual entities such as routers, switches, virtual network functions. The real network exchanges data and control messages with the twin through interfaces, allowing for real-time synchronization and control. It can span across various domains like mobile access networks, data centers, or IIoT networks.

3.2.2. Digital Twin Layer

The heart of the DTN is composed of three closely interconnected subsystems. The first is the Data Repository Subsystem, which is tasked with collecting and storing data from the real network. This subsystem ensures that accurate and up-to-date information is readily available to create precise models. The second subsystem, the Service Mapping Models Subsystem, focuses on developing and managing various data models that represent the network's components and functions. These models are essential for activities like analysis, diagnosis, and optimization, as they mirror the state and behavior of the actual network. The third subsystem is the Digital Twin Network Management Subsystem, which oversees the lifecycle, performance, and resource management of the twin. It is responsible for monitoring, visualizing, and controlling various elements of the twin, including the network topology and security, to ensure continuous synchronization with the real network. Designed with flexibility in mind, these subsystems can operate within a single domain or across multiple domains. This capability allows the DTN to deliver efficient and effective services, regardless of the complexity or geographical spread of the underlying network.

3.2.3. Application Layer

The top layer supports various network applications, such as Operations, Administration, and Maintenance (OAM), which leverage the DTN to perform network operations with minimal impact on the real network. Applications interact with the DTN through the northbound interface, making requests that are processed by the appropriate twin instances. This interaction allows for the execution of both conventional and innovative network functions in a cost-effective and non-disruptive manner.

3.3. Challenges

In the networking domain, DTNs face distinct challenges due to the highly digital nature of network elements and their varied configurations, as summarized in Table 2. Effectively addressing these challenges requires a unified architecture for DTNs, as outlined in the previous section, which forms the foundation for developing and integrating key enabling technologies. While these challenges are not exclusive to DTNs and are present in many other fields, they become particularly pronounced when designing and implementing a DTN.

Table 2. Challenges in building and maintaining Digital Twin Networks.

Challenge	Description
Scalability	Efficiently managing the growing complexity and costs of data acquisition, storage, and model implementation. Ensuring the DTN supports large-scale networks without compromising performance, accuracy, or efficiency.
Interoperability	Seamless integration of diverse software, hardware, and communication technologies requires standardized interfaces and unified data models. Varying characteristics of wired, wireless, and mesh networks, such as latency, bandwidth, and reliability, can affect DTN accuracy, performance, and functionality.
Data Modeling	Balancing accuracy, flexibility, and scalability for diverse applications and large networks. While virtualization works for small networks, larger scales require cost-effective solutions like mathematical abstractions or AI. Maintaining this balance adds complexity to developing functional models.
Real-Time	Minimizing latency and ensuring synchronization between the network and its twin. Real-time demands and simulation delays create performance challenges, requiring automated processing and optimized workflows. Matching real-time needs with suitable computing resources ensures efficient execution.
Security	DTNs face heightened risks due to constant synchronization and third-party integrations, expanding the attack surface. Robust measures like firewalls, intrusion detection, encryption, and authentication are essential to protect data and secure communication across these fronts.

For example, scalability is a common issue in large systems, but in the context of DTNs, where real-time operation is required for large networks, this challenge is magnified. Solutions may require multiple instances or a hierarchical structure of twins to maintain performance. Another challenge is accuracy, which depends on both the specific needs of the solution and budget constraints. Finally, often overlooked in research but crucial in practice, is security. Security introduces additional complexities, including latency issues in monitoring, which can exacerbate the challenges of scalability and accuracy.

4. Functional DTN Implementation

4.1. DTN Description

Figure 2 illustrates our implementation of a Digital Twin Network. Conceptually, it integrates a simulator with graph algorithms and a theoretical link model, creating a versatile and multifaceted system. This interconnected system maintains a continuous connection with the physical network, acquiring information through monitoring mechanisms. Below is a brief overview of the key facets; for a more detailed description, please refer to [16].

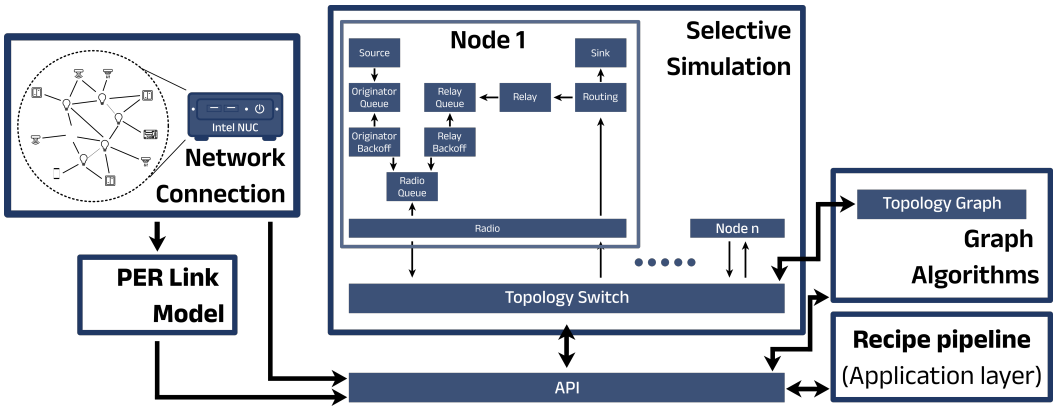

Figure 2. Visual representation of the multi-faceted Digital Twin Network for Bluetooth Mesh.

4.1.1. Selective Simulation Facet

This simulation approach uses predetermined parameter states instead of random values, reducing the need for extensive runs to observe trends efficiently. In our adapted Click Router architecture, selective simulation supports simulated time, extending the native real-time capabilities of Click [30]. An alternative approach could have been to build upon an existing network simulator and adapt it to meet the required DTN capabilities. However, no Bluetooth Mesh simulator was available at the time. Click is a flexible software architecture that allows network devices like routers to be built by combining modular packet-processing components. It lets users customize how packets are handled by defining their own processing flows, making it highly adaptable for different network needs. Thus, with some minor modifications and the addition of real-time capabilities, Click serves as a robust and viable simulator for network systems. Each component, representing a Click element, simulates Bluetooth Mesh functionalities within the DTN. Click packets, or agents, have three headers: properties (Bluetooth Mesh details), monitoring (data monitoring), and instructions (special behaviors). Each network node is represented by a composite node element with interconnected components.

4.1.2. Graph Algorithms Facet

The DTN is improved by integrating graph features and algorithms into the Click architecture. This enhancement uses the open-source C++ graph library LEMON [31] to implement a directed graph based on the topology.

4.1.3. Physical Network Connection Facet

This facet comprises two processes. First, each node in the mesh is set up to generate notifications with a Time-To-Live (TTL) of 1, reaching only neighboring nodes, at specific intervals. Second, at a separate interval, these nodes send the collected data to the gateway connected to the DTN.

4.1.4. Recipe Pipeline

Engagement with a DTN involves using predefined recipes based on Bluetooth Mesh features. These recipes can be linked into a pipeline to suggest reconfigurations, compare performance, and tailor solutions to specific application needs.

4.2. IRTF Mapping

Our interpretation [16] of the Digital Twin Network was developed independently of the IRTF interpretation. However, there is a potential mapping between the two, as they

share many concepts, albeit with different terminology. Therefore, this section describes the mapping between the key elements of both interpretations.

In our implementation, the application layer is a collection of diverse structures. It includes request–response queries, which are used to retrieve specific data points from the DTN, such as the current topology or the link quality between two nodes. Another component is recipes, which are essentially collections of queries designed to provide deeper insights into specific optimizations or configurations. Lastly, we have pipelines that group multiple recipes together; each of these pipelines is tailored for a specific application with the ultimate goal of optimizing outcomes. Hence, while we use the term recipe pipeline for application layer, both refer to the same concept, merely under different names. One of the key applications in our work is optimizing lighting performance, which is a central feature of Bluetooth Mesh. The focus of this pipeline is to minimize network delays to ensure that users do not perceive any noticeable lag, as delays above approximately 150 ms [32] are generally regarded as unpleasant. In addition to minimizing latency, the pipeline emphasizes reliability, ensuring that actions such as pressing a button result in immediate responses, such as the lighting turning on. Thus, the lighting performance pipeline is designed to optimize both responsiveness and reliability.

The real network layer consists of a Bluetooth Mesh nodes, containing a single gateway that interacts with the DTN. This gateway forwards information from the DTN into the network and also forwards aggregated data back to the DTN. We use Message Queuing Telemetry Transport (MQTT) to bridge the gap between the application layer, the DTN and gateway, ensuring efficient and reliable data exchange across all components.

The management of our DTN instance handles external requests and replies, catering to both applications and the network. Our manager serves as a combination of the data repository and management functions, streamlining the handling and storage of data. The data repository is an InfluxDB storage system where data can be extracted and temporarily stored as readable and interpretable files, enabling their use by models and Click simulators.

Models within the DTN communicate with each other using these file structures, ensuring interoperability and efficient data flow. We have two main models at the center of our multi-faceted DTN: the Click simulator and the graph model.

5. Design and Implementation Strategies

In this section, we will explore the practical steps and considerations involved in implementing a DTN. By leveraging examples from the existing literature, we aim to demonstrate why specific technologies and methodologies are chosen and how they contribute to building an effective DTN. The implementation process is anchored in the reference architecture. Through these discussions, we aim to provide a clear and detailed roadmap for implementing a DTN, offering practical insights and actionable steps that can guide professionals in bringing this sophisticated concept to life.

5.1. Data Collection

Data collection forms the bedrock of a DTN, and its effective implementation is crucial for achieving the network's operational goals. This section explores key considerations and methodologies involved in the collection, storage, and management of data within a DTN, drawing insights from both established protocols and innovative solutions tailored to specific network environments. Telemetry (in-band or out-of-band) plays a crucial role in gaining insights into network performance and its operational state. By continuously collecting and analyzing data, it offers real-time visibility into various performance metrics and system health indicators, enabling proactive management and optimization of the network.

Implementing data collection in a DTN requires careful consideration of existing protocols and tools, as displayed in Figure 3, that facilitate unified and efficient data gathering. Protocols like SNMP, NETCONF, and IPFIX are commonly utilized in IP-based networks for their robustness and interoperability [33–35]. However, the applicability of these protocols may vary depending on the underlying network technology and specific goals of the DTN. For instance, in scenarios where standardized tools are lacking, custom protocols inspired by existing frameworks need to be designed to meet specific data collection needs while aligning with network structure requirements.

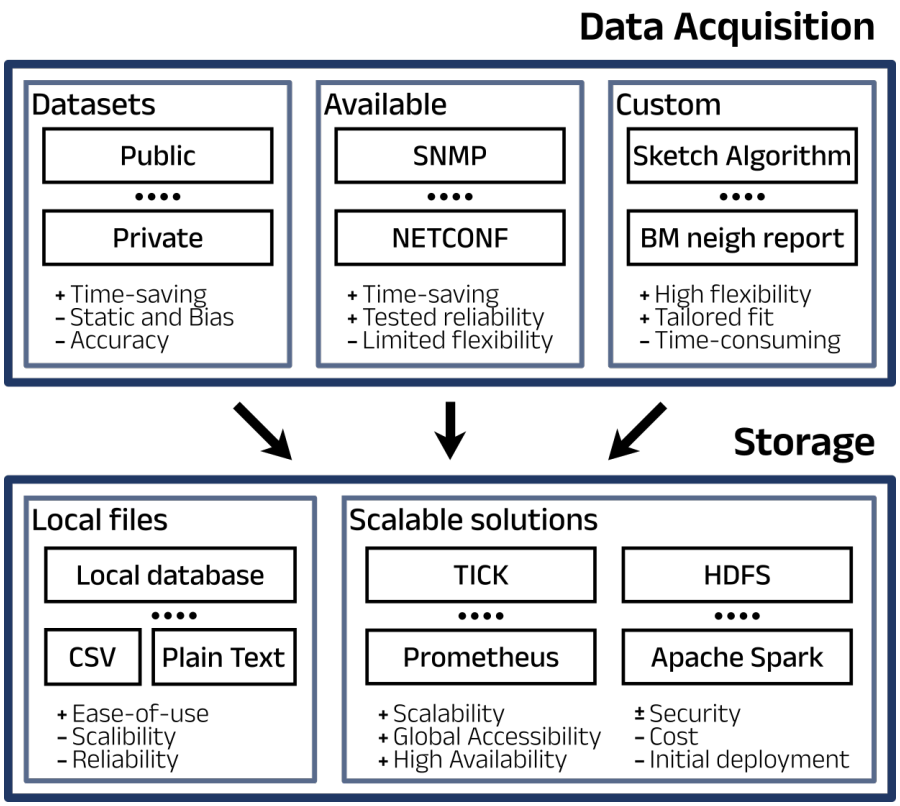

Figure 3. Overview of data acquisition and storage tools for Digital Twin Networks.

Many existing works in the literature, such as [18,19,21,23,25], often simplify data acquisition processes or rely on fixed datasets without elaborating on the methodology. Note that simulator data can often exceed the amount of information retrievable from a real network, due to the increased control and centralization they offers. Therefore, a consideration during the design phase is determining which data can potentially be collected based on the underlying technology and how new data can be effectively integrated. In contrast, refs. [24,36] propose methodologies like dataset updates and a Sketch algorithm, respectively, for real-time data aggregation and processing within DTN. The work [26] employs that sketch algorithm to enhance data collection efficiency, illustrating practical implementations in DTN environments.

Beyond acquisition, the type of data collected is critical and can include end-to-end information, configuration data, link quality metrics, buffer utilization, and more, depending on network management goals. Storing these data in a repository demands thoughtful consideration regarding scalability and data retention strategies. For instance, adopting the TICK stack (Telegraf, InfluxDB, Chronograf, and Kapacitor), as in [37], ensures efficient time-series data management, supporting both real-time analytics and historical data preservation.

In addition to the TICK stack, other scalable data storage solutions, as shown in Figure 3, like the Hadoop Distributed File System, Apache Spark, and Prometheus, are viable options, available in [38–40]. These platforms offer robust capabilities for managing large volumes of data in distributed environments, catering to the diverse needs of DTNs. Certainly, the selection of a repository depends on specific needs and objectives. For instance, in certain scenarios, simple file storage may suffice.

Lastly, effective data transformation and aggregation techniques are essential to enhance the utility of collected data for network modeling and decision-making processes.

Lightweight data collection methods are preferred to minimize resource overhead on network equipment while ensuring data relevance and accuracy for model development.

5.2. Modeling

Another key concept is the modeling, which is the core of the twin. As with the data collection, we also here have multiple possibilities for models depending on the goal of the DTN. What is common to all of these models is that they will be constructed by the same principles and use the data from the repository. It is also valuable to explore multiple interacting models, as this allows for the integration of various insights into a unified and novel perspective.

Those tools can be divided into three groups (see Figure 4), each having their target advantages and disadvantages.

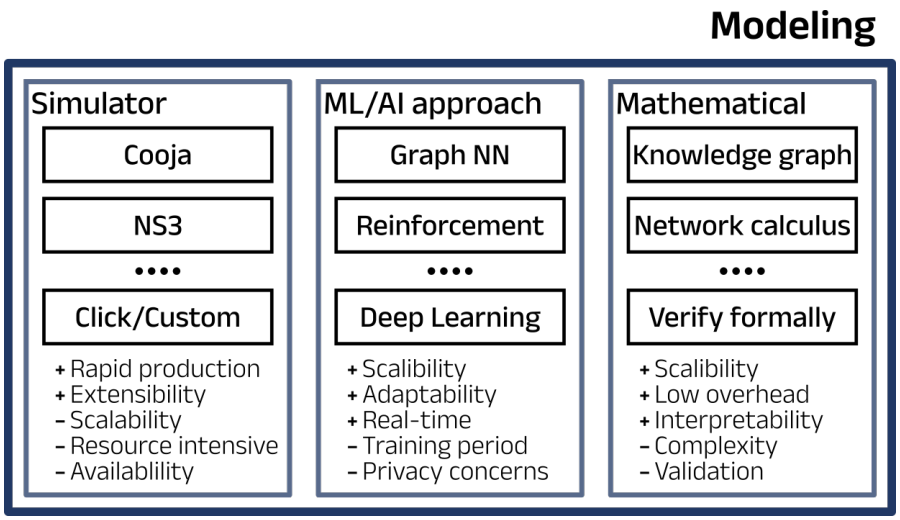

Figure 4. Overview of modeling tools for Digital Twin Networks.

5.2.1. Simulators and Emulators

These are crucial in developing and testing DTNs, especially for smaller network environments. These tools harness the packet processing capabilities of virtual network elements to rapidly verify the functions of both the control and data planes. Studies, such as [26], have utilized Cooja to simulate Internet of Things (IoT)-oriented networks in smaller settings. Lastly, the approach described by [24] utilizes an emulator to model the network. Employing these tools can significantly expedite the development process, particularly when existing simulators are available for the network protocols being developed.

However, the use of simulators and emulators is not without limitations. These tools often demand substantial resources and have limited capabilities for a complete performance analysis in real-time, making them less suitable for extensive, complex network simulations. Additionally, they struggle with scalability, meaning their effectiveness decreases as network size and complexity increase.

5.2.2. ML/AI Approach

These algorithms are key for enhancing DTNs, especially given their ability to manage complex and dynamic network conditions. These data-driven techniques utilize extensive network data to create detailed models that replicate and predict network behaviors, leading to more effective analysis and control.

The strengths of AI/ML models in DTN modeling include their capacity to generalize from known data to new scenarios and their improving interpretability, making their outcomes more reliable. Future developments will likely focus on enhancing the real-time responsiveness and interactivity of these models to ensure they can swiftly adapt to network changes and support dynamic control within the DTN.

GNNs, Deep Learning (DL), and Reinforcement Learning (RL) are among the AI methods frequently referenced. For example, while [18] discuss GNNs and DL, they do not detail their training processes. In contrast, ref. [19] effectively demonstrates the use of GNNs in DTN, showcasing how ML can be integrated to improve network performance and decision-making. Likewise, ref. [41] tackles the approach with RL, highlighting another AI technique's potential in optimizing network management and operations.

This collection highlights the benefits of AI/ML for modeling smaller or specific network segments, where these methods can rapidly adapt and provide insights. As the field advances, AI/ML are expected to play an even greater role in the evolution of DTNs, driving them towards more intelligent and autonomous operations.

5.2.3. Mathematical Models

For large-scale networks where simulators struggle due to resource constraints or complexity, mathematical abstraction methods are often employed to create efficient network models. These techniques include knowledge graphs, network calculus, and formal verification, each offering unique advantages in modeling complex network behaviors and predicting future states.

Mathematical models are currently favored for larger networks due to their scalability and lower computational overhead compared to detailed simulations. However, there is an increasing trend towards combining mathematical methods with AI/ML techniques. This hybrid approach leverages the precision of mathematical models with the adaptability of ML, providing a more robust solution for managing and optimizing large, complex networks. Despite this, it is also feasible to integrate mathematical models with simulators, creating a versatile framework that can address both high-level network dynamics and detailed packet-level interactions.

For instance, ref. [21] utilizes forecasting models to predict network behaviors and performance. Similarly, ref. [25] explores various approaches, including rule-based and policy-based models, as well as Markov models, and suggest that reinforcement learning could be used to extend these models further. This variety of methods highlights the flexibility and broad applicability of mathematical modeling in DTNs.

In their work, ref. [36] implement knowledge graphs to represent network relationships and interactions, demonstrating how this method can capture the complexity of network structures. Additionally, ref. [28] explores multiple mathematical models to address different aspects of network performance and reliability, showcasing the versatility of mathematical approaches in DTN modeling.

In summary, while mathematical models are indispensable for large-scale networks, the choice of the modeling technique depends on the needs and the network, as well as the experience and background of the developers of the DTN. The trend towards integrating these methods with ML and simulation techniques reflects the evolving needs.

This convergence promises more comprehensive and adaptable solutions for network management and optimization.

5.3. Visualization

Visualization is a crucial aspect of DTN systems, fulfilling the need for network visibility technology to accurately and intuitively display data and model outcomes. This technology not only mirrors the real-time interaction between the physical network and its twin but also aids in comprehending the internal network structure. Despite its importance, visualization is often underrepresented in research literature or omitted altogether, as many studies focus more on the technical and analytical aspects of DTNs.

Platforms, such as Kibana [42], Grafana [43], Power BI [44], or Splunk [45], can be employed to enhance development speed by providing robust, ready-made visualization tools. However, if greater control is needed and the budget allows for in-house tool development, this approach becomes the most suitable solution. In-house solutions can leverage technologies like NetworkX [46], Vis.js [47], D3.js [48], and other frameworks, offering flexibility to implement custom visualizations in any programming language. For instance, Figure 5 shows a screenshot of our internal visualization tool, which utilizes Grafana along with custom plugins developed using D3.js. Ultimately, the choice of visualization tools and underlying query tools, as well as their design, should align with the specific goals of the DTN and the needs of its users, whether for detailed network analysis or general monitoring.

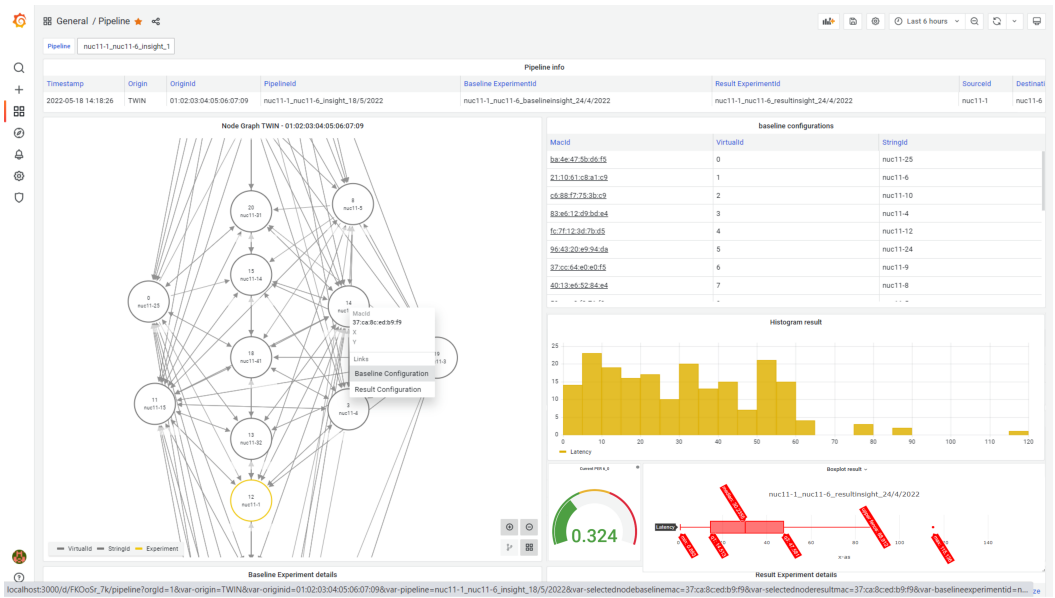

Figure 5. A snapshot of our internal visualization tool, implemented in Grafana, used to gain insights into our results and monitor network state.

Our architecture is fully decoupled, which means that adapting the visualization layer to other tools primarily involves tweaking the interfaces between components. The complexity of this adaptation naturally depends on the learning curve and flexibility of the chosen visualization tool. Although Grafana is used in our implementation and extended, similar integration could be achieved with other platforms, depending on user preference and expertise.

5.4. Interfaces

Interfaces are a key part of DTN architecture and they facilitate communication and data exchange between system components. Figure 6 illustrates the locations of the inter-

faces in our practical implementation. There are three primary types of interfaces, as per the reference architecture.

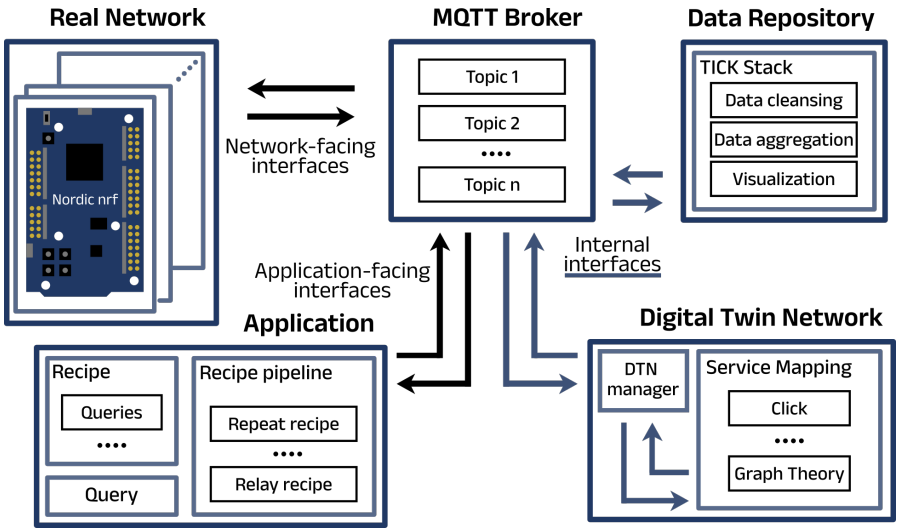

Figure 6. Visualization of Digital Twin Network interfaces alongside our implementation.

5.4.1. Network-Facing Interfaces

These interfaces connect the real network with its twin, enabling the exchange of critical network data. Typically, interfaces as indicated above (e.g., SNMP, NETCONF, and IPFIX), are utilized for real-time data collection. Due to the absence of internet protocol and the limitations of both in-band telemetry and out-of-band methods, it might be required to develop a custom model to gather monitoring data.

For the network-facing interface from the DTN to the network for configuration/reconfiguration, it is important to note that when the frequency of updates (up to real-time granularity) becomes too high, a central (single) DTN may not be feasible. In such cases, shifting to a different system is necessary. This system would involve an intermediate managing separate entity handling the southbound Application Programming Interface (API) and traffic, enabling the operation of multiple DTNs for a single network. These DTNs can then focus on algorithms, models, and AI/ML. By creating an overarching entity to manage this process, the concept of a DTN is maintained, but with enhanced scalability and robustness.

5.4.2. Application-Facing Interfaces

These interfaces link the network twin with various applications that interact with the DTN. A lightweight and adaptable interface like Representational State Transfer (REST) is commonly used as the northbound interface.

Note that user feedback can be collected through the application-facing interfaces. This would enable iterative refinement of the visualization layer based on real-world interaction and evolving user needs.

5.4.3. Internal Interfaces

These interfaces operate within the DTN, facilitating communication between its subsystems: Data Repository, Service Mapping Models, and Digital Twin Network Management. These interfaces are optimized for high speed, efficiency, and concurrency.

For example, when a new data point is collected from the real network, the Data Repository stores it and notifies the Service Mapping Models subsystem via an internal interface. This triggers the update of relevant models, which in turn communicate with the Management subsystem to assess whether any reconfiguration or alerting is needed.

This seamless internal communication ensures that the DTN remains synchronized and responsive to real-time changes.

Despite their important roles, these interfaces are often overlooked or not thoroughly discussed in the existing literature. It is essential for these interfaces to be open and standardized to avoid vendor lock-in and promote interoperability. Additionally, new interfaces or protocols may need to be developed to better suit the unique requirements of a DTN system.

5.5. Management

Management is key for the effective deployment and value extraction of DTNs in production environments. It integrates network data into a model to optimize, predict, and guide operations. This process involves continuous feedback and analysis to ensure outcomes meet expectations, creating a closed-loop system. Key aspects of management include orchestrating multiple DTN instances, ensuring their efficient deployment and resource utilization. Collaboration management is also important, coordinating roles among network administrators, data scientists, and security teams to maintain the twin's accuracy and performance. Additionally, Conflict Detection involves managing interactions within the DTN, using policies and algorithms to address issues like user intents and access controls. Finally, energy-efficient management focuses on reducing energy consumption in both network operations incited by the DTN (e.g., additional monitoring schemes) and the DTN system, contributing to greener network practices.

As with interfaces and visualization, the management of DTNs is often omitted in research, as it is more practical and less theoretical.

6. Tackling Challenges in DTN Implementation

As discussed in Section 3.3, many of the challenges faced by DTNs are not unique to them but are interconnected and often amplify one another. Several of these challenges, as outlined in Table 2, have already been addressed to some extent within the proposed architecture. For instance, the incorporation of interfaces and mapping concepts significantly enhances Interoperability, facilitating communication between different DTNs.

However, while improvements have been made, challenges such as Interoperability and others remain unresolved in certain areas and require targeted strategies to address them effectively. The following subsections will detail specific strategic approaches to mitigate these challenges and enhance the functionality and reliability of DTNs.

6.1. Scalability of DTN

Scalability is a fundamental concern in architecture and infrastructure design, but it is especially critical in DTN implementations due to the diverse challenges across multiple subsystems, as shown in Figure 7. These subsystems, such as data volume and storage, user interaction, network complexity, twin throughput, and algorithmic efficiency, often face scalability bottlenecks that can impact performance and impose budgetary constraints.

While these challenges may seem overwhelming, many have been individually addressed through existing research and tools. For smaller-scale settings, solutions such as partitioning instances into modular units and utilizing communication protocols like MQTT, Apache Kafka [49], or REST APIs can effectively manage interactions between DTN components. This modularity aligns with core DTN principles. However, as the system scales in complexity and size, these approaches alone become insufficient, particularly in networks spanning multiple domains or incorporating diverse technologies.

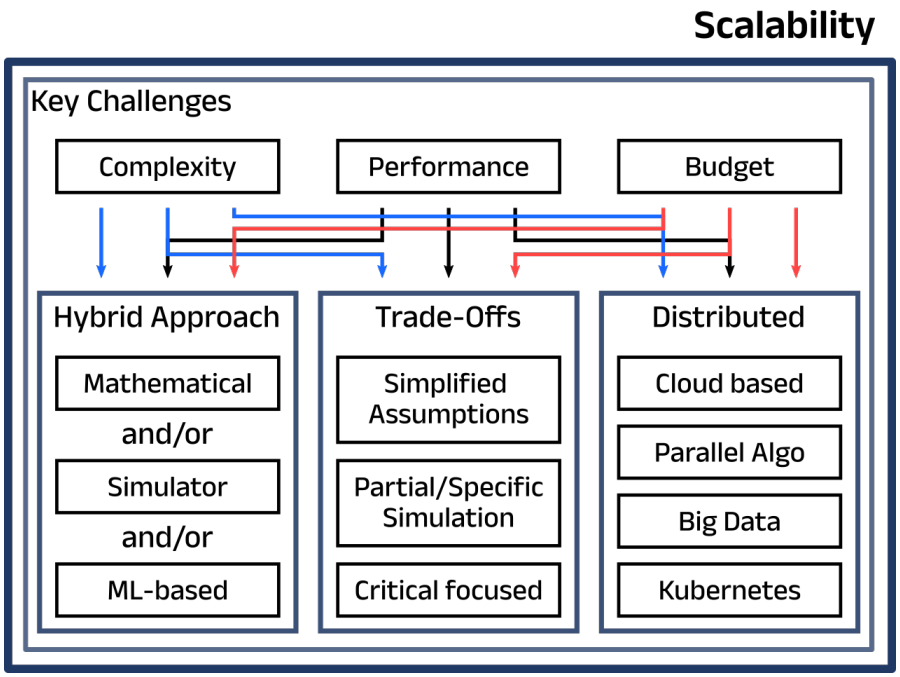

Figure 7. Key challenges in scalability and their interconnections, along with solutions, for Digital Twin Networks.

To address these limitations, a combination of modularity and hierarchical structuring is essential. Drawing inspiration from solutions like Google's routing algorithms, which leverage hierarchical subdivisions of geographic data, complex networks can similarly be divided into manageable sections. Assigning dedicated twins to oversee different parts of the network improves scalability while balancing workloads. In larger deployments, load balancers, orchestration tools, and distributed computing techniques become critical for ensuring seamless operations.

Existing technologies like Apache Hadoop [50], a cornerstone of Big Data processing, and Kubernetes [51], for orchestration and containerization, can be leveraged to handle distributed computing requirements. Kubernetes has been prototyped for use in DTN management by [52]. These tools not only manage large-scale data but also enhance uptime and operational continuity. Additionally, the widespread availability of cloud computing infrastructure provides an attractive solution for scaling DTNs, offering flexibility, reliability, and cost-effectiveness when the use case permits.

By combining modularity, hierarchical design, and distributed computing paradigms, DTNs can achieve the scalability needed to handle complex, large-scale networks while maintaining performance and reliability.

As algorithmic complexity increases with the growth of network size and components, it becomes a significant constraint in achieving scalability. Simulators inherently slow down as they attempt to model larger systems with more granular detail, observed for our DTN. To address this, hybrid solutions are often required, where certain parts of the system are actively simulated while others are approximated using mathematical models or ML-based techniques, as proposed in [53] to facilitate simulators for large-scale networks. These alternatives, though not free from their own complexities, are generally less affected by scalability limitations compared to full-scale simulation. Moreover, these algorithms can benefit from distributed computing solutions, leveraging parallel execution across multiple systems to improve efficiency and throughput.

However, scalability challenges are not merely technical. They are also influenced by budgetary constraints. While a sufficient budget can resolve many of these issues, limited resources often force developers to prioritize and make trade-offs based on the specific

goals of the DTN. For instance, developers might choose to simulate only a particular type of network traffic or introduce simplifying assumptions to reduce computational demands. In some cases, less critical elements may be ignored altogether to focus resources on the most impactful aspects of the system. Balancing these trade-offs requires careful consideration of the use case, system goals, and performance requirements.

6.2. Interoperability

Interoperability remains a significant challenge, particularly among systems developed by different companies. This is largely due to the substantial investment organizations make in creating proprietary solutions tailored to their specific needs. Understandably, companies are hesitant to compromise these investments by adopting universal standards. However, achieving interoperability necessitates such standards to ensure seamless communication and integration.

This issue is especially prominent in the IoT environment, where various groups develop independent standards that often lack compatibility. As a result, systems typically do not function together out of the box, requiring additional effort to bridge these gaps. These fragmented approaches increase complexity, costs, and inefficiencies, hindering scalability and innovation.

The draft proposal tackles [15] this for the DTN by suggesting a modular approach, dividing systems into smaller, independent components and enabling interfaces to interact in all directions. This flexibility allows developers to incorporate proprietary or open-source solutions with minimal adjustments, significantly mitigating interoperability issues. Although full interoperability may never be fully achieved, this approach fosters greater compatibility and ease of integration.

However, for this vision to succeed, consensus on the design and implementation of interfaces is essential. This will likely require leadership from either a standardizing body or a pioneering company that introduces a broadly accepted solution. In the future, network management standards, such as YANG models specifying configurable elements and collectable monitoring data, or advancements in network standards like the 3GPP Network Exposure Function (NEF), could help address these challenges. In the meantime, the best course of action is to remain open-minded and accepting of alternative solutions while actively leveraging open-source tools and frameworks wherever possible. This approach encourages collaboration, reduces redundancy, and accelerates progress toward a more interconnected ecosystem.

6.3. Data Modeling

Data modeling is naturally tied to the challenge of scalability and essentially comprises the hybrid approaches displayed in Figure 7. As highlighted in the scalability discussion, combinations of methods must be employed when the network becomes too complex for simulation, mathematical representation, or even ML-based solutions alone.

One particularly important and interesting aspect of data definition for the DTN is its ability to track data historically. While this historical tracking is not mandatory, it is both convenient and highly beneficial, especially in this case. It allows for the creation of additional environments where new data modeling techniques can be tested and validated against the same datasets as existing models. This capability significantly accelerates the development of increasingly complex data models while ensuring they are reliably validated. Furthermore, it enhances the testing and research phases, particularly in developing improved hybrid approaches.

6.4. Real-Time

Real-time functionality in DTN operates on two key frontiers. The first is how quickly a DTN can update its internal structure. The second is how efficiently it can generate new configurations when a need for adaptation is detected.

The process of internal updates largely depends on the monitoring approach used, which is inherently constrained by the type of network being employed. These constraints necessitate trade-offs and careful decision-making. In our case, we utilized a Bluetooth Mesh network, which is heavily constrained by limited data throughput. This raised a critical question: How frequently should the DTN be updated with monitoring data to ensure it remains reliable and trustworthy? To answer this, we researched, in [16], the optimal update frequency to strike a balance between maintaining accurate network representation and minimizing strain on the network. This approach allowed us to define an acceptable frequency that met our needs without overloading the system. However, DTNs designed for different networks, with distinct goals and limitations, would need to undergo a similar evaluation process tailored to their specific context. In networks without size limitations, the primary challenge shifts to scalability, specifically, the ability to process data within a responsive timeframe, even before applying models to it.

Ethical and privacy considerations are also critical when implementing real-time monitoring, particularly in end-user environments. DTN deployments must ensure compliance with data protection regulations such as the General Data Protection Regulation (GDPR), and adopt appropriate safeguards including anonymization, access control, and user consent mechanisms.

The second aspect of real-time functionality pertains to the DTN's internal operations—specifically, how quickly it can process data and execute tasks. As previously discussed, a simulator can be slow when attempting highly granular simulations. Such limitations were already observed in a DTN developed for Bluetooth Mesh [16]. This performance is directly influenced by the modeling technique employed. In our case, we initially relied on a simulator due to our team's prior expertise. However, as the DTN evolved through updates and additional research in the context of Bluetooth Mesh networks, the computational speed became a bottleneck for achieving proper real-time support.

To address this, we explored and tested variations where GNNs took over specific aspects of the simulation. Unfortunately, these trials did not yield the improvements we sought for more advanced solutions. Nevertheless, the current DTN is perfectly suited for its intended purpose, managing a typical Bluetooth Mesh network. This highlights an essential point: When designing for real-time performance, it is crucial to clearly define the operational boundaries upfront. These boundaries should guide the choice of modeling techniques and ensure the implementation aligns with the system's actual requirements.

Once again, hybrid modeling or multi-modeling is the most compelling solution, allowing the DTN to provide different levels of responsiveness. For instance, a DTN could employ two models: one designed for quick, immediate responses and another for subsequent upgrades or refinements where necessary. Ultimately, focusing on real-time capabilities requires a tailored approach that considers the specific constraints and goals of the network. This ensures the DTN operates effectively within its defined parameters while avoiding unnecessary complexities.

6.5. Security

Security in DTNs is a multifaceted challenge that demands a proactive and adaptive approach. With their interconnected nature, DTNs require comprehensive protection across all components, including the network, cloud infrastructure, interfaces, data, and third-

party communications. Below, we outline strategies to address these challenges effectively, culminating with the adoption of frameworks as a cornerstone for DTN security.

Security in DTN is not a one-size-fits-all solution. Each component of the system requires tailored protection, including the listed components in Table 3. Given the breadth of these challenges, security measures must address each layer while considering the interactions between them. Additionally, security is not static; it must evolve alongside emerging threats and advancements in technology.

Table 3. Security focus areas for multiple components within DTNs.

Component	Security Focus Areas
Network	Safeguarding against unauthorized access and attacks.
Cloud Infrastructure	Ensuring data confidentiality, integrity, and availability in cloud environments.
Models and Interfaces	Preventing tampering or misuse of the twin's algorithms and public/private APIs
Third-Party Dependencies	Protecting communications and ensuring the trustworthiness of third-party programs.

Encryption and authentication both serve as the backbone of security. Encryption, for data protection, is a non-negotiable requirement for the DTN, ensuring data confidentiality during storage and transmission. However, encryption can slow down data processing, especially in for example bandwidth-constrained networks like Bluetooth Mesh. This can be addressed by employing encryption algorithms that are both performance-efficient and secure, along with implementing end-to-end encryption to safeguard sensitive communications.

Robust authentication mechanisms are necessary to limit access to authorized users, both in the network and in connecting with the DTN. Best practices in the current day include multi-factor authentication and/or role-based access control to restrict access based on user roles.

Security frameworks play a critical role in guiding organizations towards establishing, maintaining, and continuously improving their security posture. Frameworks like the NIST Cybersecurity Framework (CSF) 2.0 [54] and ISO/IEC 27001 [55] offer structured guidance, providing multi-tiered approaches that allow organizations to implement foundational security measures and gradually advance to higher levels of resilience.

These frameworks are designed to address the evolving landscape of cybersecurity, helping organizations protect their networks, data, and critical infrastructure from various threats. They provide a scalable model that enables organizations to adapt to new challenges while maintaining effective security practices.

A consortium, leading organization, or research group should define a security framework for the DTNs, conceptually visualized in Figure 8. This would allow the smallest twins to establish a baseline level of security while progressively implementing more advanced protections, while considering the variable landscape of twins.

The adoption of established security frameworks for DTN, as suggested in this paper, can serve as a foundation for DTN security. These frameworks offer proven methodologies that align well with the unique requirements of DTNs. By incorporating these best practices, DTN implementations can improve their resilience against evolving cyber threats, while establishing a scalable and adaptable foundation for future advancements.

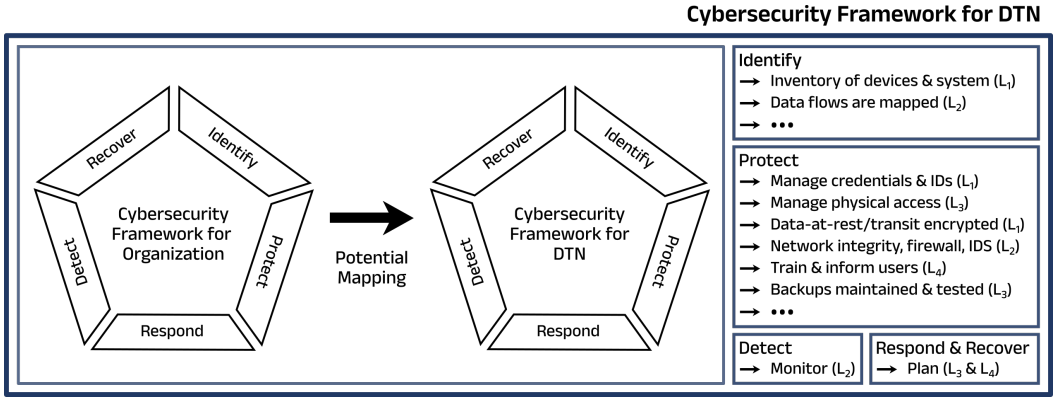

Figure 8. Conceptual mapping illustrating the similarities between an organizational cybersecurity framework and a potential framework for DTNs, inspired by the Cyberfundamental Framework [56] from the Centre for Cyber Security Belgium (CCB). Levels L_1 to L_4 represent the different tiers of security maturity.

7. Conclusions

Digital Twin Networks represent an interesting approach to network management, offering real-time insights and predictive capabilities. Throughout this paper, we have underscored the complexities and ambiguities surrounding DTN implementation, emphasizing the lack of standardized definitions and clear implementation guidelines in the existing literature.

It is crucial for organizations that venture into DTN projects to define their specific needs and goals upfront. This proactive approach ensures that the chosen tools and techniques align effectively with the desired outcomes, optimizing both implementation efficiency and operational effectiveness.

Looking forward, the establishment of an official definition for DTNs holds promise for enhancing interoperability between systems. A standardized framework would streamline communication protocols and data exchange formats, facilitating seamless integration across diverse network environments. This standardization, coupled with the availability of existing tools and methodologies discussed in this paper, would accelerate the design and implementation phases of DTN projects significantly. In addition to this standardized framework and tools, the DTN would greatly benefit from a unified cybersecurity framework tailored specifically for Digital Twin Networks.

As DTNs continue to evolve and gain traction in various industries, future research efforts should focus on refining these definitions and expanding the repository of practical tools. By doing so, the network community can collectively advance the capabilities and applicability of DTNs, paving the way for more resilient and efficient network infrastructures.

Author Contributions: J.W.: conceptualization, methodology, software, validation, formal analysis, investigation, resources, data curation, writing—original draft preparation, visualization; M.B.: writing—review; J.H.: writing—review and editing, supervision, project administration. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: No new data were created or analyzed in this study. Data sharing is not applicable to this article.

Acknowledgments: During the preparation of this manuscript/study, the author(s) used chatpgt-4o for the purposes of spellcheck and sentence restructuring. The authors have reviewed and edited the output and take full responsibility for the content of this publication.

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

Abbreviations

The following abbreviations are used in this manuscript:

AI	Artificial Intelligence
API	Application Programming Interface
BLE	Bluetooth Low Energy
CDS	Connected Dominating Set
DL	Deep Learning
DT	Digital Twin
DTN	Digital Twin Network
GDPR	General Data Protection Regulation
GNN	Graph Neural Networks
IDT	Intelligent Digital Twin
IIoT	Industrial Internet of Things
IoT	Internet of Things
IRTF	Internet Research Task Force
KPI	Key Performance Indicator
ML	Machine Learning
MQTT	Message Queuing Telemetry Transport
OAM	Operations, Administration, and Maintenance
PER	Packet Error Rate
QoS	Quality-of-Service
REST	Representational State Transfer
RL	Reinforcement Learning
TTL	Time-To-Live

References

1. Al-Sarawi, S.; Anbar, M.; Alieyan, K.; Alzubaidi, M. Internet of Things (IoT) communication protocols: Review. In Proceedings of the 2017 8th International Conference on Information Technology (ICIT), Amman, Jordan, 17–18 May 2017; pp. 685–690. [CrossRef]
2. Baldini, G.; Karanasios, S.; Allen, D.; Vergari, F. Survey of Wireless Communication Technologies for Public Safety. *IEEE Commun. Surv. Tutor.* **2014**, *16*, 619–641. [CrossRef]
3. Parikh, P.P.; Kanabar, M.G.; Sidhu, T.S. Opportunities and challenges of wireless communication technologies for smart grid applications. In Proceedings of the IEEE PES General Meeting, Minneapolis, MN, USA, 25–29 July 2010; pp. 1–7. [CrossRef]
4. Eisen, M.; Ribeiro, A. Optimal Wireless Resource Allocation With Random Edge Graph Neural Networks. *IEEE Trans. Signal Process.* **2020**, *68*, 2977–2991. [CrossRef]
5. Sun, H.; Chen, X.; Shi, Q.; Hong, M.; Fu, X.; Sidiropoulos, N.D. Learning to optimize: Training deep neural networks for wireless resource management. In Proceedings of the 2017 IEEE 18th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC), Sapporo, Japan, 3–6 July 2017; pp. 1–6. [CrossRef]
6. Liu, L.; Yin, B.; Zhang, S.; Cao, X.; Cheng, Y. Deep Learning Meets Wireless Network Optimization: Identify Critical Links. *IEEE Trans. Netw. Sci. Eng.* **2020**, *7*, 167–180. [CrossRef]
7. He, Y.; Zhang, Z.; Yu, F.R.; Zhao, N.; Yin, H.; Leung, V.C.M.; Zhang, Y. Deep-Reinforcement-Learning-Based Optimization for Cache-Enabled Opportunistic Interference Alignment Wireless Networks. *IEEE Trans. Veh. Technol.* **2017**, *66*, 10433–10445. [CrossRef]
8. Wu, Y.; Zhang, K.; Zhang, Y. Digital Twin Networks: A Survey. *IEEE Internet Things J.* **2021**, *8*, 13789–13804. [CrossRef]
9. Almasan, P.; Ferriol-Galmés, M.; Paillisse, J.; Suárez-Varela, J.; Perino, D.; López, D.; Perales, A.A.P.; Harvey, P.; Ciavaglia, L.; Wong, L.; et al. Digital Twin Network: Opportunities and Challenges. *arXiv* **2022**, arXiv:2201.01144. [CrossRef]
10. Nguyen, H.X.; Trestian, R.; To, D.; Tatipamula, M. Digital Twin for 5G and Beyond. *IEEE Commun. Mag.* **2021**, *59*, 10–15. [CrossRef]
11. Vallée, A. Digital twin for healthcare systems. *Front. Digit. Health* **2023**, *5*, 1253050. [CrossRef]
12. Zhou, G.; Zhang, C.; Li, Z.; Ding, K.; Wang, C. Knowledge-driven digital twin manufacturing cell towards intelligent manufacturing. *Int. J. Prod. Res.* **2020**, *58*, 1034–1051. [CrossRef]

13. Mashaly, M. Connecting the Twins: A Review on Digital Twin Technology & its Networking Requirements. *Procedia Comput. Sci.* **2021**, *184*, 299–305. [CrossRef]
14. Reiche, L.T.; Gundlach, C.S.; Mewes, G.F.; Fay, A. The Digital Twin of a System: A Structure for Networks of Digital Twins. In Proceedings of the 2021 26th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), Vasteras, Sweden, 7–10 September 2021; pp. 1–8. [CrossRef]
15. Zhou, C.; Yang, H.; Duan, X.; Lopez, D.; Pastor, A.; Wu, Q.; Boucadair, M.; Jacquenet, C. Digital Twin Network: Concepts and Reference Architecture. 2024. Available online: https://datatracker.ietf.org/doc/draft-irtf-nmrg-network-digital-twin-arch/ (accessed on 10 May 2024).
16. Wieme, J.; Baert, M.; Hoebeke, J. Managing a QoS-enabled Bluetooth Mesh network using a Digital Twin Network: An experimental evaluation. *Internet Things* **2024**, *25*, 101023. [CrossRef]
17. Mozo, A.; Karamchandani, A.; Gómez-Canaval, S.; Sanz, M.; Moreno, J.I.; Pastor, A. B5GEMINI: AI-Driven Network Digital Twin. *Sensors* **2022**, *22*, 4106. [CrossRef]
18. Mozo, A.; Karamchandani, A.; Sanz, M.; Moreno, J.I.; Pastor, A. B5GEMINI: Digital Twin Network for 5G and Beyond. In Proceedings of the NOMS 2022-2022 IEEE/IFIP Network Operations and Management Symposium, Budapest, Hungary, 25–29 April 2022; pp. 1–6. [CrossRef]
19. Wang, H.; Wu, Y.; Min, G.; Miao, W. A Graph Neural Network-Based Digital Twin for Network Slicing Management. *IEEE Trans. Ind. Inform.* **2022**, *18*, 1367–1376. [CrossRef]
20. Hui, L.; Wang, M.; Zhang, L.; Lu, L.; Cui, Y. Digital Twin for Networking: A Data-Driven Performance Modeling Perspective. *IEEE Netw.* **2023**, *37*, 202–209. [CrossRef]
21. Zhao, J.; Xiong, X.; Chen, Y. Design and Application of a Network Planning System Based on Digital Twin Network. *IEEE J. Radio Freq. Identif.* **2022**, *6*, 900–904. [CrossRef]
22. Becattini, M.; Borsatti, D.; Bujari, A.; Carnevali, L.; Garbugli, A.; Khachatrian, H.; Raptis, T.P.; Tarchi, D. Empirical Application Insights on Industrial Data and Service Aspects of Digital Twin Networks. In Proceedings of the 2024 IEEE International Mediterranean Conference on Communications and Networking (MeditCom), Madrid, Spain, 8–11 July 2024.
23. Bellavista, P.; Giannelli, C.; Mamei, M.; Mendula, M.; Picone, M. Application-Driven Network-Aware Digital Twin Management in Industrial Edge Environments. *IEEE Trans. Ind. Inform.* **2021**, *17*, 7791–7801. [CrossRef]
24. Yang, H.; Li, Y.; Yao, K.; Sun, T.; Zhou, C. A Systematic Network Traffic Emulation Framework for Digital Twin Network. In Proceedings of the 2021 IEEE 1st International Conference on Digital Twins and Parallel Intelligence (DTPI), Beijing, China, 15 July–15 August 2021; pp. 94–97. [CrossRef]
25. Zhao, L.; Han, G.; Li, Z.; Shu, L. Intelligent Digital Twin-Based Software-Defined Vehicular Networks. *IEEE Netw.* **2020**, *34*, 178–184. [CrossRef]
26. Kherbache, M.; Maimour, M.; Rondeau, E. Network Digital Twin for the Industrial Internet of Things. In Proceedings of the 2022 IEEE 23rd International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM), Belfast, UK, 14–17 June 2022; pp. 573–578. [CrossRef]
27. Eriksson, J.; Österlind, F.; Finne, N.; Tsiftes, N.; Dunkels, A.; Voigt, T.; Sauter, R.; Marrón, P.J. COOJA/MSPSim: Interoperability testing for wireless sensor networks. In Proceedings of the 2nd International Conference on Simulation Tools and Techniques, Rome, Italy, 2–6 March 2009; Simutools'09. [CrossRef]
28. Lu, Y.; Maharjan, S.; Zhang, Y. Adaptive Edge Association for Wireless Digital Twin Networks in 6G. *IEEE Internet Things J.* **2021**, *8*, 16219–16230. [CrossRef]
29. Zaki-Hindi, A.; Sottet, J.S.; Turcanu, I.; Faye, S. Network Digital Twins for 6G: Defining Data Taxonomy and Data Models. In Proceedings of the 2024 IEEE Conference on Standards for Communications and Networking (CSCN), Belgrade, Serbia, 25–27 November 2024. [CrossRef]
30. Kohler, E.; Morris, R.; Chen, B.; Jannotti, J.; Kaashoek, M.F. The Click Modular Router. *ACM Trans. Comput. Syst.* **2000**, *18*, 263–297. [CrossRef]
31. Dezső, B.; Jüttner, A.; Kovács, P. LEMON—an Open Source C++ Graph Template Library. *Electron. Notes Theor. Comput. Sci.* **2011**, *264*, 23–45. [CrossRef]
32. Welford, A.; Brebner, J. *Reaction Times*; Academic Press: Cambridge, MA, USA, 1980.
33. Internet Engineering Task Force. A Simple Network Management Protocol (SNMP). 2025. Available online: https://www.ietf.org/rfc/rfc1157.txt?number=1157 (accessed on 8 January 2025).
34. Internet Engineering Task Force. Network Configuration Protocol (NETCONF). 2025. Available online: https://datatracker.ietf.org/doc/html/rfc6241 (accessed on 8 January 2025).
35. Internet Engineering Task Force. IP Flow Information Export (IPFIX) Implementation Guidelines. 2025. Available online: https://datatracker.ietf.org/doc/html/rfc5153 (accessed on 8 January 2025).

36. Zhu, Y.; Chen, D.; Zhou, C.; Lu, L.; Duan, X. A knowledge graph based construction method for Digital Twin Network. In Proceedings of the 2021 IEEE 1st International Conference on Digital Twins and Parallel Intelligence (DTPI), Beijing, China, 15 July–15 August 2021; pp. 362–365. [CrossRef]
37. InfluxData. TICK Stack. 2025. Available online: https://www.influxdata.com/time-series-platform/ (accessed on 10 January 2025).
38. Shvachko, K.; Kuang, H.; Radia, S.; Chansler, R. The Hadoop Distributed File System. In Proceedings of the 2010 IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST), Incline Village, NV, USA, 3–7 May 2010; pp. 1–10. [CrossRef]
39. Zaharia, M.; Xin, R.S.; Wendell, P.; Das, T.; Armbrust, M.; Dave, A.; Meng, X.; Rosen, J.; Venkataraman, S.; Franklin, M.J.; et al. Apache Spark: A unified engine for big data processing. *Commun. ACM* **2016**, *59*, 56–65. [CrossRef]
40. Prometheus. Prometheus Monitoring System and Time. 2025. Available online: https://prometheus.io (accessed on 10 January 2025).
41. Dai, Y.; Zhang, K.; Maharjan, S.; Zhang, Y. Deep Reinforcement Learning for Stochastic Computation Offloading in Digital Twin Networks. *IEEE Trans. Ind. Inform.* **2021**, *17*, 4968–4977. [CrossRef]
42. Elastic. Kibana: Explore, Visualize, Discover Data—Elastic. 2025. Available online: https://www.elastic.co/kibana (accessed on 10 January 2025).
43. Grafana Labs. Grafana: The Open Observability Platform. 2025. Available online: https://grafana.com (accessed on 11 January 2025).
44. Microsoft. Power BI. 2025. Available online: www.microsoft.com/power-platform/products/power-bi (accessed on 30 January 2025).
45. Cisco Company. Splunk | The Key to Enterprise Resilience. 2025. Available online: https://www.splunk.com (accessed on 21 January 2025).
46. NetworkX Developers. NetworkX. 2025. Available online: https://networkx.org (accessed on 21 January 2025).
47. Vis.js Community. Vis.js Community Edition. 2025. Available online: https://visjs.org (accessed on 11 January 2025).
48. Observable. D3 js. 2025. Available online: https://d3js.org (accessed on 10 January 2025).
49. The Apache Software Foundation. Apache Kafka. 2025. Available online: https://kafka.apache.org (accessed on 12 January 2025).
50. The Apache Software Foundation. Apache Hadoop. 2025. Available online: https://hadoop.apache.org (accessed on 12 January 2025).
51. Kubernetes. Kubernetes. 2025. Available online: https://kubernetes.io (accessed on 14 January 2025).
52. Lombardo, A.; Morabito, G.; Quattropani, S.; Ricci, C. Design, implementation, and testing of a microservices-based Digital Twins framework for network management and control. In Proceedings of the 2022 IEEE 23rd International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM), Belfast, UK, 14–17 June 2022; pp. 590–595. [CrossRef]
53. Kazer, C.W.; Sedoc, J.a.; Ng, K.K.; Liu, V.; Ungar, L.H. Fast Network Simulation Through Approximation or: How Blind Men Can Describe Elephants. In Proceedings of the 17th ACM Workshop on Hot Topics in Networks, New York, NY, USA, 15–16 November 2018; HotNets'18, pp. 141–147. [CrossRef]
54. NIST. Cybersecurity Framework. 2025. Available online: https://www.nist.gov/cyberframework (accessed on 16 February 2025).
55. *ISO/IEC 27001:2022*; Information Security, Cybersecurity and Privacy Protection—Information Security Management Systems—Requirements. International Organization for Standardization and International Electrotechnical Commission: Geneva, Switzerland, 2022. Available online: https://www.iso.org/standard/27001 (accessed on 17 February 2025).
56. Centre for Cybersecurity Belgium. CyberFundamentals Framework. 2025. Available online: https://atwork.safeonweb.be/tools-resources/cyberfundamentals-framework (accessed on 17 February 2025).