JOURNAL OF
Optical Communications
and Networking

End-to-end transport network digital twins with cloud-native SDN controllers and generative AI [Invited]

Allen Abishek, Daniel Adanza, Pol Alemany, Lluis Gifre, Ramon Casellas, Ricardo Martínez, Raul Muñoz, and Ricard Vilalta*

Centre Tecnològic de Telecomunicacions de Catalunya (CTTC-CERCA), Castelldefels, Barcelona 08860, Spain
**ricard.vilalta@cttc.es*

Received 3 December 2024; revised 25 February 2025; accepted 2 April 2025; published 22 April 2025

This paper explores the potential of network digital twins (NDTs) in networking (both IP Ethernet networks and optical transport networks), highlighting their integration with cloud-native software-defined networking (SDN) controllers and intent-based networking enabled by generative artificial intelligence (GenAI). The proposed framework represents an approach that combines advanced virtualization, real-time analytics, and GenAI. The use of NDTs enables a comprehensive and dynamic digital representation of the physical network, capturing critical aspects, such as topology, traffic patterns, and performance metrics, which permits data-driven decision-making to lead to more efficient networking operations. The incorporation of cloud-native SDN controllers along with an NDT ensures that the system remains scalable, flexible, and responsive to dynamic network conditions. Intent-based networking, powered by GenAI, allows the network to interpret high-level objectives from operators and autonomously translate them into actionable configurations that are enforced by orchestrators and SDN controllers. This eliminates manual intervention, minimizes errors, accelerates the deployment of network services, and provides a means for easier network management. The presented framework significantly enhances automation, enabling predictive maintenance by identifying potential issues before they impact network performance. It optimizes network design by simulating various configurations and testing their feasibility in a risk-free environment. These capabilities collectively improve operational efficiency, reduce downtime, and ensure optimal resource utilization.

https://doi.org/10.1364/JOCN.550864

1. INTRODUCTION

With the recent development in telecommunications technology, digital processes are increasingly replacing manual processes, significantly enhancing efficiency and capability. A key part of this digital transformation is the concept of the digital twin [1]. A digital twin is an integrated, data-driven virtual representation of real-world entities and processes, with continuously synchronized interactions. By integrating data from sensors, telemetry, and other monitoring sources, digital twins enable precise simulation, analysis, and optimization of complex systems. These virtual models are widely applied across diverse domains, such as manufacturing, health care, and telecommunications to predict system behavior, identify potential failures, and implement corrective actions in a risk-free environment. Adopting digital twins enhances operational performance, minimizes downtime, and drives innovation, offering a novel, proactive, and seamless approach to managing and optimizing critical systems in real time [2].

A digital twin is more than a one-time simulation model; it is a continuously updated, data-driven virtual replica of a physical system. In contrast to traditional network simulators, which work with static or historical data, a digital twin ingests real-time telemetry, enabling it to mirror the ongoing behavior and state of the actual network. This persistent synchronization ensures that operational insights and predictive models remain accurate and timely, even as conditions evolve. Such a dynamic representation goes beyond "what-if" evaluations, as it supports iterative experimentation, proactive maintenance, and closed-loop decision-making. As a result, digital twins provide a powerful basis for optimizing network design, orchestrating resources, and integrating AI-driven processes, all while minimizing risk to live operations.

In this paper, the system of interest being replicated virtually is the "network"; thus, it shall be called the "network digital twin" (NDT).

ETSI zero-touch network and service management (ZSM) defines the NDT as the digital twin whose physical counterpart

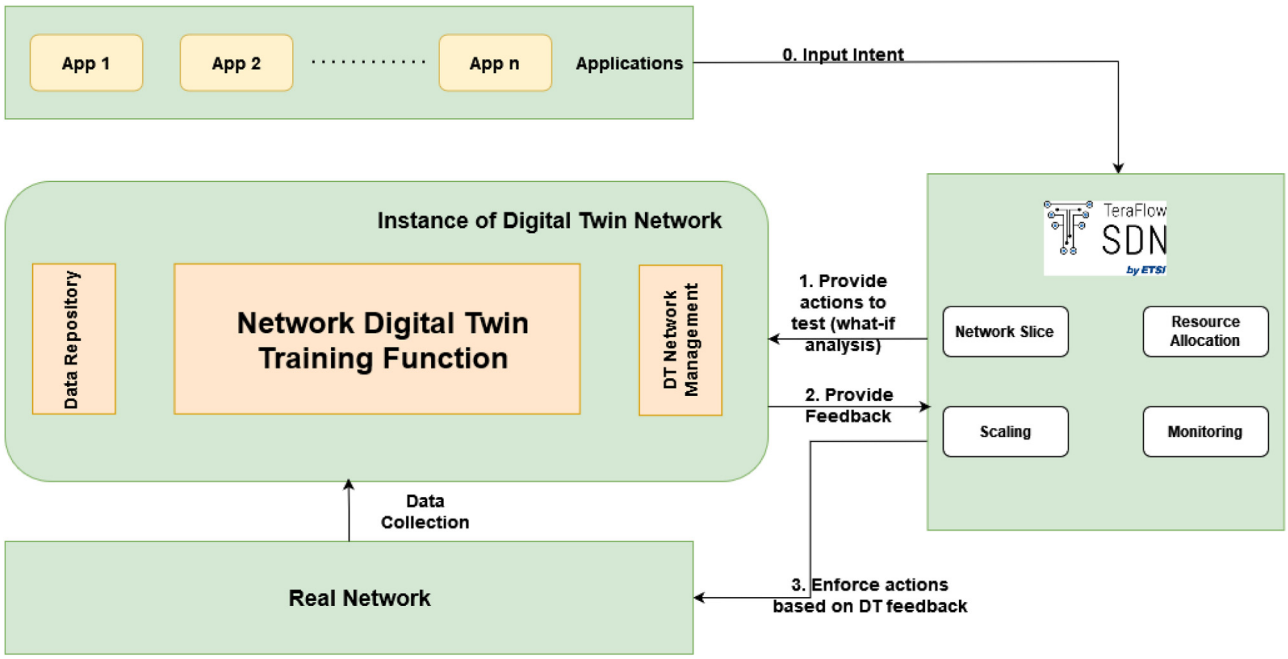

Fig. 1. NDT for network operations.

is a communication network or some part of one. The communication network can include physical network elements and components, virtualized network functions (VNFs—i.e., network functional elements instantiated as software-based entities), and physical hosts for such VNFs, services, and traffic [3]. Figure 1 illustrates the concept and functionality of an NDT system, emphasizing its integration with physical networks and its utilization of artificial intelligence (AI). It can be seen that NDTs have multiple functionalities. The architecture of Fig. 1 shows a high-level diagram of how an NDT facilitates easier network operations and multiple applications running on a network infrastructure that is managed by a software-defined networking (SDN) controller that is connected to an NDT as well. The SDN controller manages operations done on the NDT as well as the network operations and configurations of the real physical network. Some of the capabilities of the SDN controller are not limited to network slicing, resource allocation, and monitoring (i.e., telemetry). The NDT training function aids in predictive maintenance and what-if scenario analysis. We shall now look in detail at the capabilities, functionalities, expected performance, and behavior of an NDT. The key feature of an NDT is the intrinsic nature of performing iterative operations; multiple different scenarios can be simulated in the NDT as a "what-if" scenario, and the optimization component, which is coupled with the NDT, can learn from the simulated network traffic, network configuration, cybersecurity attack, topology changes, etc. [1]. When the NDT makes any changes to the configuration, network protocols, or traffic management in itself, many iterations of different changes and their effect on the NDT can be studied by the optimization component to discover the most optimal configuration or topology, protocol parameters, security policies, and more. This capability reduces the risk of potential issues arising from changes to the network. AI-driven approaches, such as reinforcement learning or deep learning-based surrogate modeling, permit the NDT to optimize and forecast scenarios in a data-driven manner. That reduces extensive manual parameter tuning or more static rule-based systems. This is particularly useful when dealing with highly dynamic, large-scale network environments, where traditional optimization methods might struggle with scalability or adaptability.

The NDT with the parallel AI agent/application continuously optimizes network operations, enhancing performance and efficiency while reducing the need for manual human intervention. AI-driven analysis helps predict issues, propose solutions, and streamline network management processes. The NDT can be used primarily for anomaly detection, system state assessment, and resource allocation as shown in [4]. Thus, it can be said that the NDT is now an essential architectural component that is used as a means to achieve AI-enabled automated network operations in a zero-touch fashion.

There are primarily two types of network infrastructures: IP networks, which are primarily used in data centers (on-premise enterprises), and optical networks. An IP Ethernet network consists of network elements such as, but is not limited to, switches, routers, hubs, and firewalls. These network elements are layered in the network topology. Optical networks are networks that connect two locations that are physically separated by large distances using optical technology. An optical network primarily consists of, but is not limited to, components such as reconfigurable optical add/drop multiplexers (ROADMs), optical terminal devices, optical modulators, optical amplifiers, infrared lasers, and PIN photo-diodes. The entire end-to-end transport network is shown in Fig. 2, including a combination of the optical transport network (OTN) and the IP network domains. NDTs when introduced into the system enable transformative solutions offering a range of capabilities that

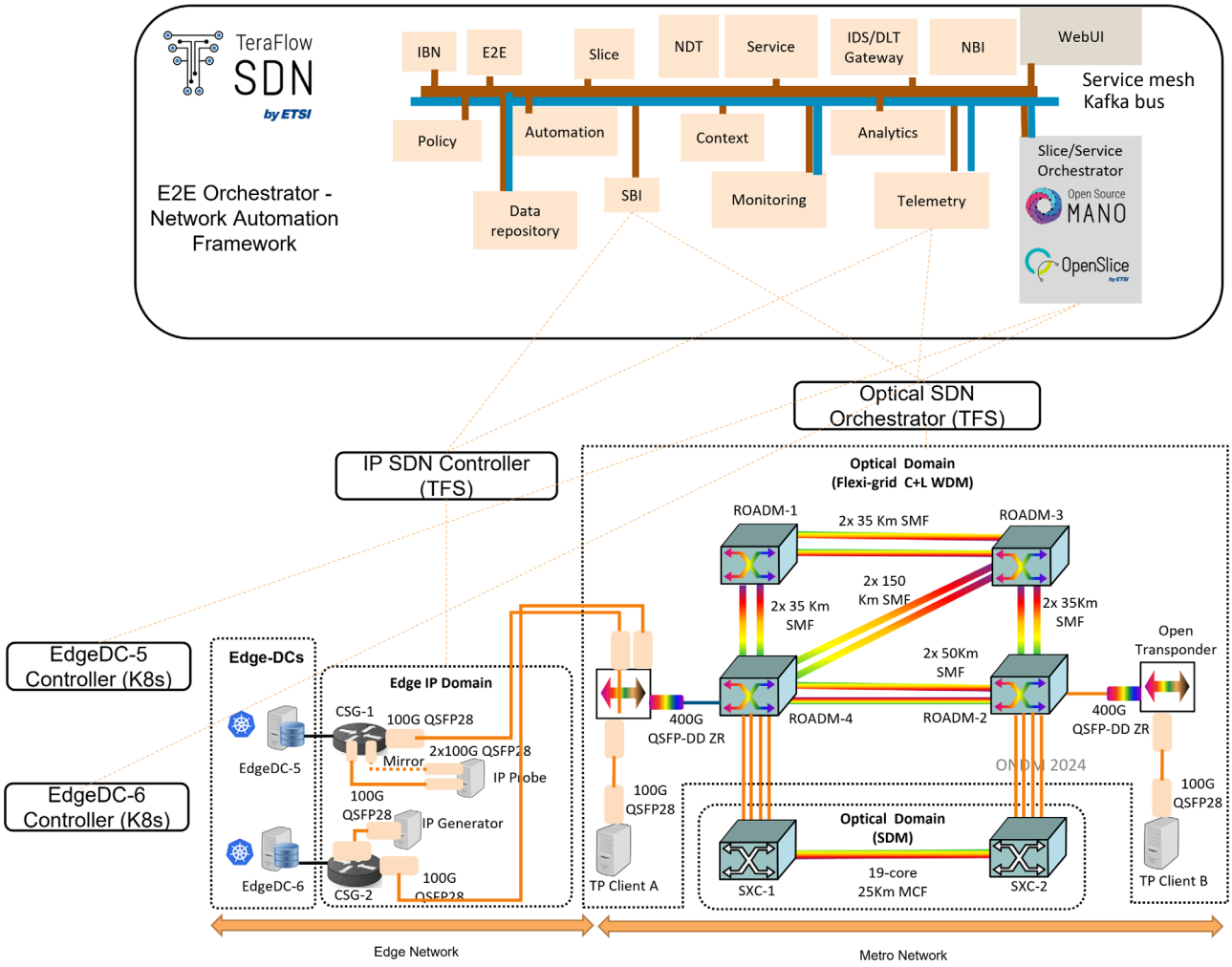

Fig. 2. Architecture for deployment of an end-to-end network digital twin.

enhance network design, automation, traffic management, maintenance, and security. We shall now discuss the different ways an NDT can add value to network infrastructure and network operations; we will go into the technical details in Section 3 of this paper.

• Enhanced Design and Performance: The NDT allows network administrators to test topology changes, configuration changes, and security policy changes on the NDT before implementing these actions on the actual network in the real world. This proactive, informed decision-making approach minimizes risks and makes the actual network more resilient and less prone to damage.

• Automation of Continuous Monitoring: The NDT's important feature is the ability to dynamically replicate the physical state of the actual physical network in real time without any human intervention; this is made possible by continually monitoring the actual network and replicating its state in the NDT [5]. This functionality of the NDT allows any optimization component running in parallel with the NDT to perform optimization changes iteratively, thus enabling zero-touch provisioning (ZTP). This in turn leads to enhanced operational efficiency and reliability.

• Traffic Management: The NDT provides real-time insights into traffic patterns, enabling proactive planning for future demands. By forwarding a replica of real user plane data traffic to the NDT, the behavior of the NDT in response to the network traffic can be used to enforce policies in network configuration to manage network traffic. This ensures smooth operation without congestion, even during peak traffic conditions. A good example of this can be the use of the NDT of the OTN segment of the transport network, which is defined as ONDT, to identify bottlenecks in data traffic flows in the OTN and implement traffic steering policies in the ONDT and verify whether the changes implemented in the ONDT reduce traffic congestion in the actual network; this is demonstrated in the paper by Zalat *et al*. [6].

• Predictive Maintenance: With the help of continuous network monitoring consistently replicating the network state in real time, this helps in improving the efficiency and reliability of network analysis; network analysis makes it possible to proactively foresee potential issues that are soon to occur and create a policy or action to be executed before the issue arises. This minimizes the risk of errors and ensures a better quality of seamless automated network operations.

• Security: NDTs enhance overall network security by simulating cyberattacks to test and refine the defense strategies of a network against threats and attacks from malicious entities. The NDT continually monitors network security for anomalies, malicious packets, suspicious logins, user profiles, user activity, etc. These data are used to develop robust mechanisms to counter evolving dynamic threats. With the incorporation of AI in NDTs, the paper by Muhammad *et al.* involves using adversarial networks to train the optimization component in the ONDT in real time to be better prepared to identify and mitigate attacks that may come to the actual network from malicious untrustworthy entities [7].

• Equipment Management: NDTs enable the precise modeling of a network's current operational state while forecasting future requirements. This functionality facilitates the identification of equipment nearing the end of its operational life cycle, ensuring timely replacement. Additionally, NDTs simulate the deployment and integration of new hardware, evaluating its compatibility and potential impact on the overall network. This proactive approach minimizes performance disruptions and supports the continuous enhancement of network infrastructure, contributing to robust and adaptive system performance.

Now that it has been established that, to perform an accurate what-if analysis or seamless automated network configuration changes for optimization of the actual network, the NDT and its operations must be an accurate, identical replica of the actual network, there is a need for both an independent IP packet and optical NDT to provide a complete virtual replication of the complete network infrastructure as shown in Fig. 2. In this paper, the NDT of optical and IP Ethernet networks is instantiated separately but is integrated to function as a single entity as the actual transport network is an IP network run on top of an optical DWDM network. This is one of the main novelties introduced in this paper. It is important to note that the integration of AI with the NDT enhances the capabilities of the NDT by enabling sophisticated analytical processes, intelligent decision-making, and automated optimization. A detailed description of the architecture and working is explained in Section 3, and the validation of the setup is shown in Section 5 of this paper.

This paper extends the paper by Vilalta *et al*. [8] with end-to-end orchestration of network services and generative AI (GenAI) included in the architecture. The previous paper only focused on an optical NDT, and this paper extends it with a solution for handling the NDT life cycle for end-to-end transport networks. Moreover, this paper also extends previous results, with the proof-of-concept of natural language processing using GenAI of a high-level intent in nontechnical vocabulary, which can be fed as input to the IBN component that translates the intent into technical details of how to achieve the intent.

The organization of this paper is as follows. Section 2 reviews related work, providing an overview of existing research and technologies relevant to this study, highlighting similarities and technological gaps addressed by the work in this paper. Section 3 discusses the NDT architectural setup that was conceived by the authors of this paper; it also discusses the details of automation and replication of processes, network operations, and the topology of the actual network into the NDT. The methods to quantify and validate the performance of the NDT are also described in this section. Section 4 goes into the concept of intent-based networking (IBN), detailing its integration with NDTs and GenAI. This section explores how these technologies enable seamless, autonomous network operations with minimal human intervention, emphasizing their role in optimizing network management and performance. Next, Section 5 presents the experimental setup and results, providing a detailed analysis of the outcomes. This section also includes an in-depth discussion of the underlying factors influencing the results. Finally, Section 6 concludes the paper by summarizing the key findings, their contributions to the field, and the broader implications for network management. Additionally, this section outlines directions for future research, emphasizing potential advancements and extensions of the work presented.

2. RELATED WORK

NDTs are digital representations of physical networks used for simulation, analysis, and optimization, as discussed by Fuller *et al*. [9]. They enable proactive maintenance and fault prediction by detecting potential issues before they affect live operations, as shown by Wang *et al*. [10]. NDTs help anticipate faults and reduce disruptions in real-world operations. The rapid advancement of network automation technologies has replaced many manual configuration processes with digital solutions. This shift makes it easier to integrate NDTs into network infrastructures as a core component of network architecture [11]. Automation also allows NDTs to dynamically reflect the real-time state of physical networks, enabling continuous monitoring and optimization in complex information and communications technology (ICT) systems.

It is important to note that the accuracy of the replication of the network state of the actual network within an NDT directly impacts its performance. Most modern networks, when abstracted, consist of multiple interconnected layers, including OTNs for long-haul connections, IP/MPLS networks, and Ethernet networks within data centers [1]. Diverse tools already exist to emulate this layered structure.

The authors of [12] present a comprehensive framework for full life-cycle management of optical networks via real-time monitoring, mirror modeling, and automatic control. They highlight how emerging DT solutions—powered by both physics-driven and data-driven methods—enable more accurate representations of real-world fiber-optic infrastructures and pave the way for autonomous network configuration, performance prediction, resource optimization, and fault resolution. Crucially, the authors explore how softwarized measurement approaches and hybrid modeling techniques (combining physical laws with machine learning) can reduce complexity and improve reliability, especially as networks evolve toward multi-band, large-scale deployments.

Curri [13] leverages the open-source GNPy library to demonstrate how accurate physical-layer models can be seamlessly integrated into a digital twin framework for optical networks. By capturing key transmission parameters—such

as amplifier gain, noise profile, and nonlinear interference—GNPy enables an end-to-end view of link performance and capacity limits. The authors show that this approach not only facilitates realistic simulations of multi-vendor environments but also supports proactive network planning and automated fault recovery. Their results underscore the importance of robust software-based models in digital twin design, where the combination of real-time network telemetry and physics-informed modules can significantly enhance the agility, reliability, and cost-effectiveness of optical infrastructure management.

For example, Mininet-Optical [14] is used to simulate OTNs, following methods proposed in [15], while Rieger *et al*. [16] employ containerlab to replicate IP/Ethernet network topologies. The paper by Vilalta *et al*. highlights the use of NDTs to integrate cognitive intelligence into networks, aiming to create resilient, zero-touch systems with minimal human intervention [15].

NDTs can also be used to enhance the working of applications running on an ICT system by optimizing the network design of network topology, network configuration, network security policies enforcement, network resource allocation, and much more; this can be achieved by employing machine learning algorithms to enact the actions stated above as shown in the paper by Qin *et al*. [4].

NDTs can also strengthen cybersecurity by simulating network attacks. This permits the AI algorithms to have some training and to trigger countermeasures. This results in better protected live networks, as shown in the paper by Kumar *et al*. [17]; blockchain technology is used for authentication of devices into the network, and the NDT stores all the operating states and different behavior models of entities attached to the network. The NDT is integrated with the SDN controller of the actual network to ensure seamless communication between the real world and the digital world. In the context of autonomous networks, the NDT integrates itself with closed-loop automation (CLA) systems, which use real-time network data and AI-driven analytics to adjust configurations and policies dynamically, as shown in the paper by Kumar *et al*. [18]. The important role that NDTs play in autonomous networking has been demonstrated in a paper by Ferriol-Galmés *et al*. [19]. The paper showcases how graphical neural networks complement the functionalities of the NDT to accurately model different network parameters: topology, routing, queuing policies, and traffic to predict service-level agreement (SLA) metrics (e.g., delay and jitter). The novelty of the paper relies on the usage of a GNN and the generalization of the training network parameters instead of being network specific.

Thus, it can be said that NDTs serve as controlled environments for evaluating policy changes, enabling the safe application of validated modifications to live networks, as highlighted by Mirzaei *et al*. [20]. This proactive methodology not only mitigates potential failures but also enhances overall network performance. The integration of NDTs into ZSM frameworks significantly bolsters their capabilities, paving the way for more autonomous and resilient network infrastructures. Moreover, this approach presents substantial research opportunities for advancing network performance and reliability within the telecommunications ecosystem.

Limitations, developing scalable and accurate NDTs, require significant computational resources and advanced modeling techniques [1]. Real-time synchronization with live network components poses technical challenges due to high processing demands and data handling requirements, often straining the computing and network resources of existing infrastructures. The adoption of NDTs in network management remains in its early stages, with integration into real-time operations hindered by complexities, such as interoperability across diverse network architectures and maintaining synchronization under dynamic conditions. These limitations highlight the need for further research to address scalability and efficiency, as well as to develop more proactive mechanisms for failure prediction and mitigation in practical deployments.

This paper takes insights, ideas, and approaches from the papers discussed in this section and adds novelty and innovation by introducing the integration of optical and IP NDTs together with IBN embedded with GenAI running on top of the NDT for easier user interaction with the NDT and seamless network operations by forming a CLA platform for network operations by instantiating network services dynamically in real time as per the needs of the SLA and network infrastructure. This approach significantly enhances the network availability, reduces the downtime, and supports the development of resilient, fault-tolerant network architectures. Traditional methods often lack proactive mechanisms for failure prediction and mitigation.

3. ARCHITECTURE OF THE NDT

The E2E TeraFlowSDN (TFS) orchestrator is the parent SDN controller, which has the IP SDN controller and the optical SDN controller under it. This E2E orchestrator gives visibility and control to both IP segments and optical segments of the transport network. The parent SDN controller also instantiates the NDT (which contains both optical and IP segments) and interfaces with the NDT and plays a key role in fidelity in the NDT's replication of the actual network. Moreover, the E2E orchestrator acts as a network automation framework, comprising several core modules and handling different aspects of network management. The E2E orchestrator is able to handle IP over DWDM network scenarios to ensure faster communication with minimized latency from source to destination. IP routers connect to the underlying transport network—often a mix of optical and/or Ethernet-based transmission paths. IP-capable edge routers or Layer 3 switches form the boundary between the customer side (IP-based devices) and the provider's core transport layer. The transport network (which may include ROADMs, optical amplifiers, and Ethernet switches) ensures that data are moved efficiently over long-distance or high-bandwidth links, while IP devices handle packet-level decisions at the edges.

The IP SDN controller manages the IP-based network segments, connecting with Edge Data Center Controllers (EdgeDC-5 and EdgeDC-6). This can be seen in Fig. 2. These edge data centers utilize Kubernetes (K8s) for container orchestration. Figure 2 shows the different modules present in the parent SDN controller, which is also a TFS instance. The modules of TFS that are of interest in this paper are the NDT,

telemetry, SBI, automation, and IBN. The NDT module is what instantiates the NDT, real-time network traffic is passed into the NDT with the help of the telemetry module, and SBI helps the parent SDN controller connect to the child SDN controller for the IP network segment and the child SDN controller to connect to the optical segment of the actual network; automation is responsible for the automation of policies, configurations in the network, and the IBN module for intent translation to technical implementations.

We adopt TeraFlowSDN due to its openness, cloud-native microservices' architecture, robust multi-domain and multi-technology support, alignment with ETSI standards, built-in intent framework, and active developer community. Nonetheless, the proposed framework can readily integrate alternative SDN controllers with comparable capabilities.

Another important component is IBN, which permits users to define their network requirements by focusing on "what" needs to be done instead of "how" it should be implemented. Its main role consists of translating high-level intents into concrete network policy configurations. For this purpose, IBN integrates large language models (LLMs) and natural language processing techniques to carry out tasks like intent translation, feasibility checks, conflict resolution, SLA management, and closed-loop control. This will be explained in detail in Section 4.

Alternatively, the optical SDN controller oversees the optical network segments, interacting with physical optical components, such as ROADMs, optical amplifiers, and optical transceivers. To ensure comprehensive network management, the E2E TFS orchestrator incorporates several critical modules, with the NDT module being one of them. The role of the NDT module in the TFS controller is to create a virtual replica of the actual network topology within its module; this provides a digital environment for thorough testing and validation of network changes in the NDT before they are implemented on the actual network, thus ensuring that any modifications do not adversely affect the network's performance.

The life cycle of an NDT is initiated with the deployment phase, wherein the NDT integrates with a Kafka-based streaming system to receive real-time topology data from the SDN controller of the physical network. This interaction is facilitated through the Transport Application Programming Interface (TAPI) of the Open Networking Foundation (ONF). The optical line system (OLS) subsequently publishes the network topology, enabling the configuration and deployment of virtual SDN controllers and optical nodes, and once the topology network elements (switches, routers, firewalls, optical links, Ethernet links, optical amplifiers, ROADMs, etc.) have been instantiated to form the NDT, the next step is to update the NDT to make it an accurate replica of the actual network. During the update phase, the SDN controller utilizes ONF TAPI streaming to publish changes or updates in network topology. This prompts the NDT to dynamically reconfigure its virtual nodes and update telemetry data to ensure accuracy; this in turn will optimize the performance of the NDT and help with more efficient network operations. The optical network segment of the NDT is emulated by an emulator tool called Mininet-Optical, and the IP network segment of the NDT is emulated by the emulator tool containerlab. Accurate replication can be ensured by the real-time streaming of the telemetry data of the configurations and states of the network elements in the actual network and implementing them in the NDT; this can ensure the fidelity of the PHY layer of both the IP and optical domains accurately in the NDT.

The operation phase of the NDT allows for the testing of connectivity services, leveraging tools such as quality of transmission (QoT) estimators to assess feasibility. Only validated services are applied to the physical network, ensuring robustness and performance optimization and at the same time the NDT training function, which has in-built AI or machine learning models, studies the data from network telemetry and builds models to optimize network configurations, policies, topologies, etc. This can be seen in Fig. 2, where the Data Repository and Context store information on network state information, and the NDT training function trains its models using the information stored in the Data Repository. Finally, when there is no more need for the existence of the NDT, the removal phase involves the controlled decoupling and termination of the NDT, ensuring a clean disconnection from the network.

The sequence diagram of the TFS controller alongside the NDT life cycle is shown in Fig. 3. There, the entire process pipeline of working is described as follows. A client, usually the network operator, sends a request to the NDT Manager component that sets up the NDT. The process of instantiating the NDT is called NDT Initialization. The NDT(s) are instantiated through a network function virtualization-orchestrator (NFV-O) client. The NFV-O is responsible for deploying the necessary virtualization tools to deploy the NDT. In our implementation, an example of NFV-O is ETSI Open Source MANO (OSM). TFS later configures the instance of the NDT. Once the setup of the NDT into the network infrastructure is complete, it is now ready for runtime operations.

In the next phase called runtime operations where the state of the actual network is replicated in the NDT by using means of telemetry and state replication. The NDT continually listens to the monitoring OSM module in TFS, by which any new change in the actual is immediately mimicked in the NDT; this is a continual loop that exists from the moment of instantiation of the NDT and its deployment. The final phase in this process is the performance of the "what-if" scenario analysis, where different what-if scenarios are tested and validated on the NDT.

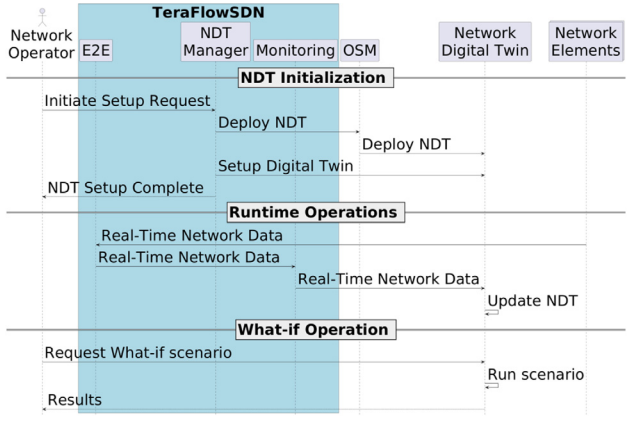

Fig. 3. NDT sequence diagram.

This structured life cycle ensures that the NDT serves as an accurate, dynamic replica of the physical network, supporting advanced validation, testing, and optimization of network services, thus getting closer to achieving autonomous networking with ZSM.

4. INTENT-BASED NETWORKING AND GENAI

IBN [21] allows users to input their needs and requirements with an abstract request that contains only the "what," but not the "how" (i.e., specific policies or configurations on network devices) the system should orchestrate and manage the resources in the network to fulfill the requests from the users.

Hence, network resource identification, specific policies, and configurations of network devices are delegated to the IBN system. This component must take care of identifying multiple entities composing an intent, such as expectations, targets, and contexts [22]. Although this functionality could be provided by more traditional natural language processing (NLP) techniques, the use of LLMs permits greater automatization and flexibility in carrying out these tasks. A key feature of an IBN system is its flexibility to automatically ensure user intent by continuously adjusting policies and validating real-time network conditions, which results in the generation of a report describing the fulfillment of the intent and any conflict between intents and the feasibility of an intent.

As described in [23], an IBN-based solution should be capable of managing an intent in its whole life cycle [22] through the use of multiple functionalities illustrated in the architecture of Fig. 4:

- Intent Interface: This component serves as the primary gateway for users to engage with the intent management solution and represents a core element of its architecture. It manages intent data objects and oversees the various actions required to control the life cycle of intents. Its key functionalities include (a) intent interpretation, which is powered by an LLM and has the role of processing user requests and enabling the system to understand the intended actions; (b) intent capability exposure, which provides users with an overview of the system's capabilities, detailing what can be managed and achieved through intents; (c) intent CRUD operations, which facilitate essential create, read, update, and delete operations, triggering processes necessary for managing intents effectively; (d) intent report management, which is also powered by an LLM for attribute identification and the overwriting of the data objects composing the reports, whose functionality is the inclusion of methods to generate reports related to intents, aligning with the intent reporting functional block; and finally, (e) intent ADSR, which enables actions, such as activate, deactivate, suspend, and resume, allowing intents to remain in a standby state until needed again.
- Intent Fulfillment: This functional component comprises essential capabilities that remain inaccessible to users but are critical for enabling the features they interact with. Key functionalities within this module include (a) intent translation, which converts intents and their associated key performance indicators (KPIs) and key value indicators (KVIs) into executable service management tasks and policies derived from the agreed SLA and is performed using LLMs and retrieval augmented generation (RAG) techniques, which provide the LLM with a base knowledge to enhance the accuracy of its responses; (b) intent feasibility checking, which verifies that incoming intent requests can be effectively executed based on the system's current state and resources; (c) intent closed-loop (CL) governance and coordination, which has two primary objectives to initiate the required instances of "intent-driven CL control for fulfillment evaluation" to ensure proper deployment and compliance with the specified requirements and to manage the coordinated operation of multiple closed loops within the IBN framework, as detailed in [3]; and finally, (d) intent conflict detection and resolution, which detects potential conflicts when a new intent is introduced or an action is applied to an existing intent, ensuring that these do not negatively impact other intents, but if conflicts are identified, this service resolves them efficiently, and this functionality is critical to maintaining the system's integrity, ensuring smooth execution and achieving the desired outcomes.
- Intent Reporting: The capabilities within this functional component manage various types of information related to intents. The "intent feasibility check information" indicates whether an intent can be deployed successfully, considering factors such as the availability of current resources. The "intent

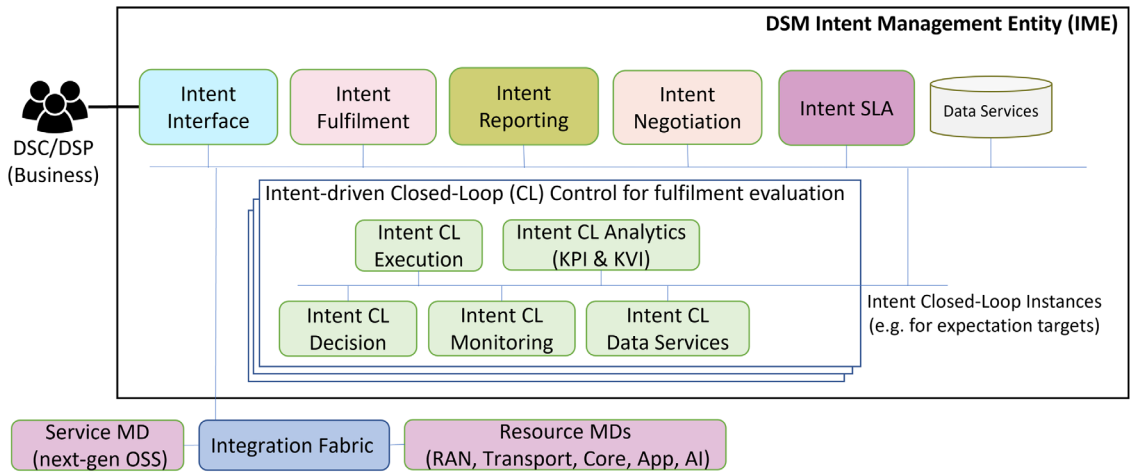

Fig. 4. Intent-based architecture: functionalities and components.

fulfillment information" consolidates data on the status of intent execution and the achievement of associated targets. Last, the "intent conflict information" manages details about detected conflicts and their resolution processes.

- Intent Negotiation: This functional block enables IBN users to access detailed information about intents prior to their provisioning. Its internal functionalities include (a) probing an intent to assess whether all expectations can be met, providing tenants with clarity on whether their requirements are achievable; (b) allowing tenants to identify the most demanding intent requirements that the system can successfully handle, closely tied to the feasibility assessment; and finally, (c) offering the capability to simulate hypothetical outcomes for alternative actions (i.e., preferences), serving as a foundation for decision-making by evaluating potential results from different solution strategies and actions before implementation. This final functionality relies on predictive capabilities to estimate how the proposed actions would impact the system's state.
- Intent SLA: This block enhances SLA management and empowers users to make more informed decisions when formulating intents to identify the requirements related to the desired quality of service (QoS) (i.e., policies and KPIs). To this end, two functionalities are identified: (a) 3P profiling, which enables a comprehensive characterization of tenants (i.e., third parties) through a detailed profile and captures tenant-specific information, including security features (such as supported credentials and access control mechanisms), trustworthiness metrics (especially relevant in federation scenarios), and user profiling data linked to contracted services and SLAs, and (b) the service portfolio, which provides tenants with an abstracted overview of available 6G services and their associated SLAs, effectively functioning as a product catalog. This information allows tenants to craft more accurate and appropriate intent requests. The available service details include aspects such as service status (e.g., defined, designed, built, tested, and released), ownership, variations within the same service (e.g., different SLA options), costs, and dependencies on other services.
- Intent-Driven Closed-Loop Control for Fulfillment Evaluation: This functional block provides the necessary capabilities to manage the life cycle of intent CL instances, ensuring that their requirements are consistently met. Once an intent CL instance is created, it encompasses the following functionalities: (a) intent CL governance service, which facilitates orchestration by managing all actions related to the assigned intent instance; (b) intent CL execution, which carries out tasks as directed by the governance service; (c) intent CL analytics (KPI & KVI), which processes monitored metrics to extract insights and valuable information; (d) intent CL decision, which utilizes the insights generated by the analytics functionality to determine the most appropriate course of action for the intent CL instance; (e) intent CL monitoring, which collects data and metrics associated with predefined KPIs for subsequent analysis by the analytics capability; and finally, (f) intent CL data services, which provides storage for all information related to each specific intent CL instance.
- Data Services: This functional block is responsible for maintaining intent data objects along with additional information, such as SLAs and policies.

The design of all these functional blocks is done to have a complete and stable use of intents that should allow, to any user, an agile and easy way to manage the service and resources [24,25] available in all the management domains (MD) below through the cross-domain integration fabric.

In the context of an NDT, IBN can significantly enhance the efficiency of deploying network services and resources. By simulating several rounds of adjustments and validations on the digital twin platform, the impact on actual networks is minimized. Therefore, an NDT serves as a crucial platform for implementing IBN systems and facilitating their deployment.

IBN requires integration of an LLM in order to process natural language from human-based requests and identify the different items that compose an intent (i.e., expectations, targets, and contexts) into network actions to properly deliver the final vertical service expected by the users. In this scenario, the authors have presented a chatbot within the TeraFlowSDN framework that marks a significant advancement in intent-based networking [26]. This chatbot, known as IntentLLM, is integrated with TeraFlowSDN, and it interprets user prompts and converts them into transport slice requests, demonstrating the creation and management of new network slices in real time. This development builds on previous work in intent-based networking by incorporating more sophisticated natural language processing capabilities, allowing the chatbot to understand the context and synonyms to provide detailed explanations of network intents.

5. EXPERIMENTAL VALIDATION

This section implements a selected subset of features presented in this paper in relation to the NDT and IBN. To this end, Section 5.A focuses on the experimental validation of an E2E NDT, and Section 5.B presents a validation of the IBN architecture.

A. Evaluation of the Life-Cycle Management of an End-to-End NDT

The presented solution for an E2E NDT uses two different software to virtually replicate the optical and the IP domains. First, the optical network is represented by Mininet-Optical. Second, the IP network is represented by containerlab. Likewise in transport networks, when a new optical link is established in the optical network, the corresponding virtual link is included in the IP layer of the respective digital twins. All the experimental validations are done over 50 iterations, and cumulative distribution function graphs are plotted for each experimental validation.

Containerlab is an open-source framework designed to simplify the deployment and management of virtual network topologies using containerized network functions. It leverages container technologies such as Docker and Kubernetes. Containerlab provides a lightweight and scalable environment for emulating complex network scenarios. Every network element in a topology is run as a container in the containerlab environment. It supports a wide range of network operating systems and integrates seamlessly with network automation

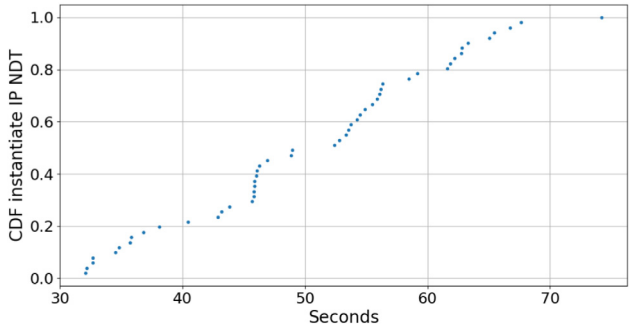

Fig. 5. Time taken to instantiate an IP network digital twin on containerlab.

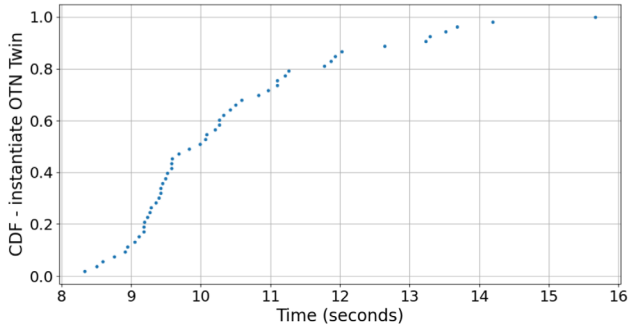

Fig. 6. Time taken to instantiate a transport network twin on Mininet-Optical.

tools, enabling accurate testing, validation, and optimization of network configurations efficiently [16].

The topology built-in containerlab consists of a topology with two Nokia SR Linux switches connected to two Nokia SR Linux switches and each switch connected to a host end-device running on Linux. This is the NDT of the IP layer presented in Fig. 3. Figure 5 shows the cumulative distribution function (CDF) of the time taken in seconds to instantiate the IP-based NDT topology on containerlab, with the median time taken of 50.371 s and the mean time taken to be around 52.420 s. It can be seen that around 80% of the IP network topology instantiations are completed in 60 s, while the other 20% of the network topology instantiations are completed in 60–78 s. The significance of the 80% threshold is that, with 80% probability, the time taken to perform any action (like add a node, add a link, and instantiate the NDT) with 80% accuracy will be less than the time value that corresponds to the value 0.8 in the CDF distribution.

Mininet-Optical is an advanced extension of the Mininet emulator, tailored specifically for the accurate emulation of optical transport networks. It enhances traditional network emulation by incorporating optical-layer components, such as wavelength division multiplexing (WDM), optical cross-connects, transponders, and optical modulators. These features enable the detailed modeling and testing of optical transport networks, supporting dynamic wavelength allocation, traffic engineering, and fault management in a virtualized environment [14]. Being an open-source tool, it provides a cost-effective and flexible solution for studying the complexities of optical transport networks and accurately replicating the state of the actual optical layer as an NDT.

The optical network topology consists of four ROADM optical switches, where each is connected to an optical network terminal, each optical line terminal is connected to an Ethernet link from the IP network, and the optical line terminal converts fiber optical signals to digital signals and vice versa. The four ROADMs are in turn connected to each other. In this scenario, the time taken to instantiate the mentioned topology as an NDT in Mininet-Optical is analyzed and validated. From Fig. 6, it can be seen that the mean time required to instantiate the NDT is around 9.983 s, and the median time is around 9.642 s. Figure 6 shows that 80% of the topology instantiations occur between 8 and 11 s.

In order to consider the scalability of the proposed solution, we have measured the time taken to add a node to both the

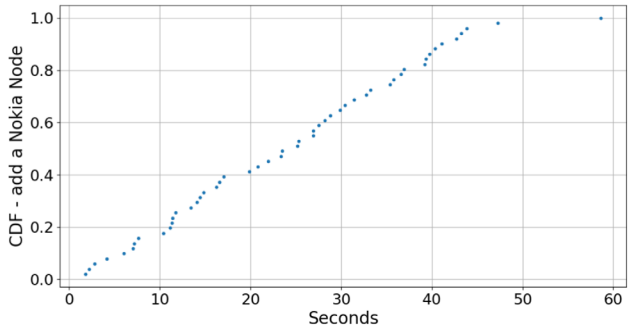

Fig. 7. Time taken to add a Nokia SR Linux switch to an IP NDT.

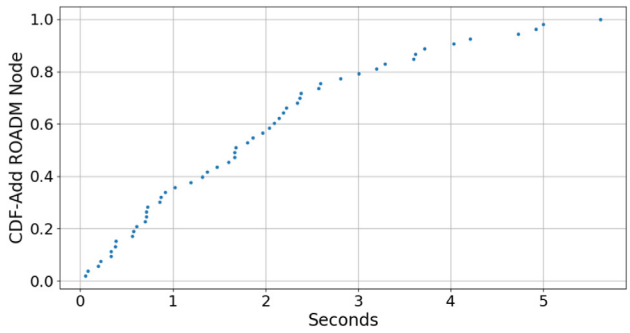

Fig. 8. Time taken to add a ROADM to the optical network twin.

IP NDT on containerlab and the optical NDT on Mininet-Optical. A fifth node is added to the IP network's leaf–spine topology in the leaf layer. The results of the validation can be seen in Fig. 7, which shows that the mean time to add a Nokia SR Linux switch is 24.256 s, and the median time is 25.160 s.

A ROADM node is also added to the optical NDT. In Fig. 8, the mean time to add a ROADM node to the NDT of the optical network is calculated to be 1.935 s, and the median time is 1.683 s.

Upon comparison of the time required to add a node to the IP NDT and the optical NDT, it can be seen that the time taken to add a Nokia SR Linux switch to the containerlab topology is considerably longer than the time taken to add a ROADM node to the OTN. This can be explained by the fact that the container/Nokia SR Linux switch being added to the topology is much heavier in size as it is an emulated networking operating system in a container. The Mininet-Optical NDT

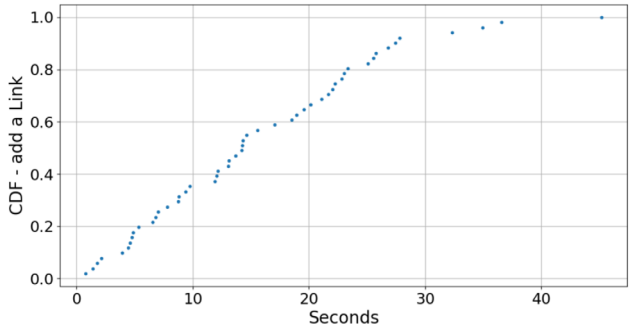

Fig. 9. Time taken to add an Ethernet link to the IP NDT.

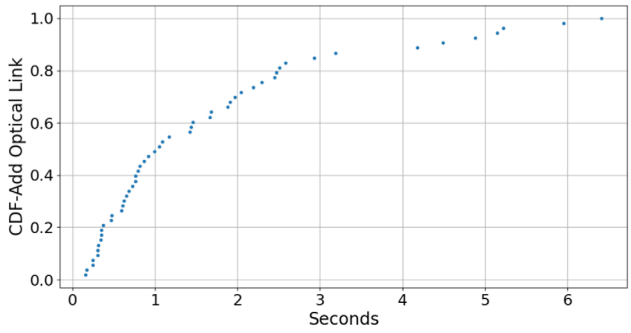

Fig. 10. Time taken to add an optical link to the optical transport NDT.

focuses on emulating the effects of physical operations of the nodes, but it is not an emulated operating system, which is why the instantiation process with an added ROADM node is faster than when the node is a Nokia SR Linux switch. Another validation and analysis is performed to study the time taken to add a link between two nodes in the NDT topology of both the IP network and the optical network. In Fig. 9, it can be seen that the mean time to add a link between the two Nokia SR Linux switches in the spine layer is 15.798 s, and the median time taken is 14.250 s.

It can also be seen from Fig. 10 that the mean time taken to add an optical link between two ROADM nodes in the network topology running on Mininet-Optical is found to be 1.689 s, and the median time is 1.052 s. From the CDF distribution, it can be seen that 80% of the iterations take around 2 s to add an optical link between two ROADM nodes.

The time taken to add an optical link to the optical NDT is much less than the time taken to add an IP link to the IP NDT. This is consistent with the fact that the nodes to be instantiated in containerlab are much heavier and consume more computing resources than those in Mininet-Optical.

B. Validation of the IBN Architecture

The architecture of the IBN system has been presented in Section 4, and it has the goal of transforming user requests into an intent deployed on the network structure. In this section, we propose to validate the presented architecture with a proof-of-concept that includes the proposed components providing different functionalities. Additionally, the IBN architecture has been tested 20 times, and the accuracy results and the execution time have been gathered.

1. Walkthrough Example

The components comprising the IBN architecture are illustrated in Fig. 11. The proposed architecture works as follows:

A **user request** performs the petition through the WebUI interface. This petition is received through the **northbound interface (NBI)**. The NBI has the main role of filtering the received requests and applying some simple logic to call the corresponding components. Thereafter, the **interpreter** carries on the work. Its main role can be summarized in two processes. The first process is to extract the relevant information from the received user request and identify the specified KPIs and their corresponding values. The second process is to generate the intent JSON object according to the extracted KPI values. This data structure follows the standards specified in the 3GPP data models.

To have a more reliable validation, a practical example is given, and the provided results by the IBN architecture are shown. For the received intent request, "provide a connectivity service between node A port X and node B port Y of QoS gold and BW 1T." First of all, the interpreter identifies the main KPIs. The KPIs identify relevant attributes, such as location attributes, including port, source, and destination nodes; time restrictions, indicating specific dates or durability restrictions; and other bandwidth restrictions. For the given intent request, the following data structure has been received by the IBN architecture:

{ source: A, sourcePort: X, destination: B, destinationPort: B, QoT: gold, bandwidth: 1T, }

As a second step, the interpreter would use this extracted information to construct the intent object. For clarification, the inputs and outputs provided by the IBN architecture are gathered in Fig. 12.

After the intent object has been generated by the interpreter, the information is forwarded to the **manager**, which is in

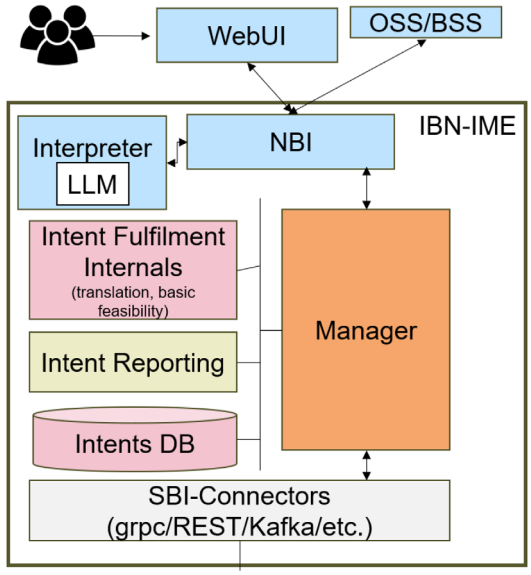

Fig. 11. Main components of the IBN implementation.

Fig. 12. Sample case of a constructed intent.

charge of communicating with the rest of the components and ensuring that the tasks are indeed being carried out.

Figure 12 provides an example of an intent in the validated implementation of the IBN architecture; to perform these tasks, the interpreter has been powered with the LLM model Llama 3 8B to handle both tasks—the KPI extraction and the JSON object generation. After the intent object is generated, the manager communicates with the **intent fulfillment internals** to check the feasibility of the intent and to ensure that the hierarchical structure of the generated JSON intent indeed complies with the 3GPP data model. Second, the manager also communicates with the **intent reporting** to generate corresponding reports with the updated information for the network. Third, the hierarchical information composing the intent and the reports is stored in the **intents database**. Finally, the manager communicates with the **southbound interface (SBI) connectors** to proceed with the intent deployment of the network.

C. Time and Accuracy Results

To have a deeper view of the results presented by the IBN architecture, the solution was tested with a sample of 20 different intent requests.

The total execution time for each intent request has been gathered, and the CDF was illustrated in Fig. 13, where it is possible to appreciate that the total execution time occurred

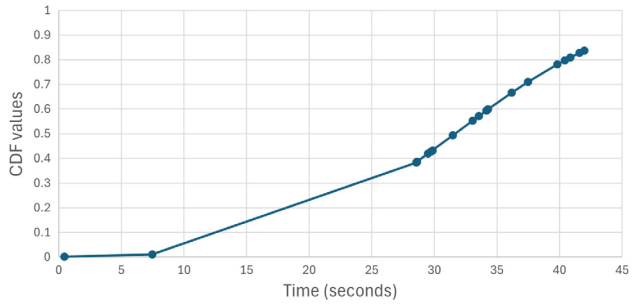

Fig. 13. CDF values for the amount of time taken to construct and trigger the intent request in the network.

in a range from 0.45 to 42 s to deploy the whole execution, and 40% of the samples took less than 28 s to be executed. However, there is also a bigger concentration of values in the range from 28 to 37 s, which represents 30% of the samples registered.

Regarding accuracy, it is worth taking into account that the 20 generated intent objects were compared to the expected generated objects, reaching 85% accuracy, which means the IBN system generated the intent object correctly 17 times and was wrong the remaining 3.

The reasons why the LLM was mistaken were mainly due to the usage of synonyms, such as the usage of the term "origin" instead of "source," which sometimes misleads the conception of the LLM.

It is not feasible to establish an unbiased comparison with other approaches due to the technical specifications and the size of the samples. It can be inferred that the main benefits of the suggested IBN architecture rely on automatization and flexibility. The user does not need to take into account how to handle the network configuration. The user only needs to enumerate what he/she wants, which results in a simplification of the task.

6. CONCLUSION AND FUTURE WORK

This paper has explored the application of an NDT integrated with cloud-native SDN controllers and IBN with GenAI for a network topology comprising an IP and optical network. The presented framework demonstrates significant potential for optimizing network design, automation, and maintenance, offering enhanced efficiency and performance of a network with an NDT. The integration of LLMs into this framework further enhances its capabilities, allowing for intuitive, natural language interactions and automated network management.

The presented approach not only simplifies user interactions. It also ensures that the network can adapt dynamically in an ever-changing environment automatically, implying minimal human intervention and also facilitating better network operations since it makes it easier for human interaction with the network due to the IBN interface. The paper demonstrates why NDT and IBN platforms are an integral and essential part of the network architecture. Through comprehensive simulations, real-time monitoring, and proactive maintenance, the NDT framework can significantly reduce operational costs and improve overall network reliability.

Although the primary focus of this work has been to propose a flexible architecture for multi-domain NDTs, there remains significant scope for further exploration and enhancement. For example, accuracy metrics could address not only prediction error and model convergence but also the extent to which the NDT captures system-level complexities in different operational scenarios.

Funding. HORIZON EUROPE Civil security for society [101119602 (COBALT)]; Ministerio de Asuntos Económicos y Transformación Digital, Gobierno de España [TSI-063000-2021-19 (6GMICROSDN-SMARTNICS), TSI-063000-2021-20 (6GMICROSDN-NOS), TSI-063000-2021-21 (6GMICROSDN-CLOUD)]; Ministerio de Ciencia, Innovación y Universidades [PID2021-127916OB-I00 (RELAMPAGO)].

REFERENCES

1. P. Almasan, M. Ferriol-Galmés, J. Paillisse, *et al.*, "Digital twin network: opportunities and challenges," arXiv (2022).
2. E. Glaessgen and D. Stargel, "The digital twin paradigm for future NASA and U.S. air force vehicles," in *53rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference* (2012).
3. ETSI, "Zero-touch network and service management (ZSM); closed-loop automation. v1.1.1," Tech. Rep. ETSI GR ZSM 009-1 (2021).
4. B. Qin, H. Pan, Y. Dai, *et al.*, "Machine and deep learning for digital twin networks: a survey," IEEE Internet Things J. **11**, 34694–34716 (2024).
5. Z. Wang, D. Jiang, and S. Mumtaz, "Network-wide data collection based on in-band network telemetry for digital twin networks," IEEE Trans. Mob. Comput. **24**, 86–101 (2024).
6. M. Zalat, C. Barber, D. Krauss, *et al.*, "Network routing optimization using digital twins," in *PoEM Companion* (2023).
7. K. Muhammad, T. David, G. Nassisid, *et al.*, "Integrating generative AI with network digital twins for enhanced network operations," arXiv (2024).
8. R. Vilalta, A. Abishel, L. Gifre, *et al.*, "Applying digital twins to optical networks with cloud-native SDN controllers and generative AI," in *50th European Conference on Optical Communications (ECOC)* (IET, 2024).
9. A. Fuller, Z. Fan, C. Day, *et al.*, "Digital twin: enabling technologies, challenges and open research," IEEE Access **8**, 108952–108971 (2020).
10. D. Wang, Z. Zhang, M. Zhang, *et al.*, "The role of digital twin in optical communication: fault management, hardware configuration, and transmission simulation," IEEE Commun. Mag. **59**(1), 133–139 (2021).
11. Y. Liu, W. Zhang, L. Li, *et al.*, "Toward autonomous trusted networks-from digital twin perspective," IEEE Netw. **38**, 84–91 (2024).
12. D. Wang, Y. Song, Y. Zhang, *et al.*, "Digital twin of optical networks: a review of recent advances and future trends," J. Lightwave Technol. **42**, 4233–4259 (2024).
13. V. Curri, "GNPy model of the physical layer for open and disaggregated optical networking [Invited]," J. Opt. Commun. Netw. **14**, C92–C104 (2022).
14. B. Lantz, A. A. Daz-Montiel, J. Yu, *et al.*, "Demonstration of software-defined packet-optical network emulation with Mininet-Optical and ONOS," in *Optical Fiber Communication Conference* (Optica Publishing Group, 2020), paper M3Z.9.
15. R. Vilalta, L. Gifre, R. Casellas, *et al.*, "Applying digital twins to optical networks with cloud-native SDN controllers," IEEE Commun. Mag. **61**(12), 128–134 (2023).
16. S. Rieger, L.-N. Lux, J. Schmitt, *et al.*, "DigSiNet: using multiple digital twins to provide rhythmic network consistency," in *IEEE Network Operations and Management Symposium (NOMS)* (IEEE, 2024).
17. P. Kumar, R. Kumar, A. Aljuhani, *et al.*, "Digital twin-driven SDN for smart grid: a deep learning integrated blockchain for cybersecurity," Sol. Energy **263**, 111921 (2023).
18. R. Kumar, A. Aljuhani, D. Javeed, *et al.*, "Digital twins-enabled zero touch network: a smart contract and explainable AI integrated cybersecurity framework," Future Gener. Comput. Syst. **156**, 191–205 (2024).
19. M. Ferriol-Galmés, J. Suárez-Varela, J. Paillissé, *et al.*, "Building a digital twin for network optimization using graph neural networks," Comput. Netw. **217**, 109329 (2022).
20. J. Mirzaei, I. Abualhaol, and G. Poitau, "Network digital twin for Open RAN: the key enablers, standardization, and use cases," arXiv (2023).
21. S. Kerboeuf, P. Porambage, A. Jain, *et al.*, "Design methodology for 6G end-to-end system: Hexa-X-II perspective," IEEE Open J. Commun. Soc. **5**, 3368–3394 (2024).
22. 3GPP, "Management and orchestration; intent driven management service for mobile networks. v18.2.1," Tech. Rep. (2023).
23. P. Alemany, R. Muñoz, J. A. Ordoñez-Lucena, *et al.*, "Multi-stakeholder intent-based service management automation for 6G networks," in *Joint European Conference on Networks and Communications & 6G Summit (EuCNC/6G Summit)* (2024), pp. 866–871.
24. P. Alemany, D. Adanza, L. Gifre, *et al.*, "Grouping intent-based packet-optical connectivity services," in *49th European Conference on Optical Communications (ECOC)* (2023), pp. 744–747.
25. R. Vilalta, D. Adanza, C. Manso, *et al.*, "Demonstration of intent-based networking for end-to-end packet optical cloud-native SDN orchestration," IET Conf. Proc. **2023**, 1738–1741 (2023).
26. D. Adanza, C. Natalino, L. Gifre, *et al.*, "IntentLLM: an AI chatbot to create, find, and explain slice intents in TeraFlowSDN," in *IEEE 10th International Conference on Network Softwarization (NetSoft)* (2024), pp. 307–309.