Policy-Driven SD-WAN for Hybrid Terrestrial and Non-Terrestrial 6G Connectivity

Luca Borgianni[1], Eleonora Bellini[1], Alessio Giorgetti[1], Davide Adami [2], Stefano Giordano[1]

[1]*University of Pisa, Dept. of Information Engineering, Via G. Caruso 16, 56122 Pisa, Italy*
[2]*CNIT - University of Pisa, Dept. of Information Engineering, Via G. Caruso 16, 56122 Pisa, Italy*

***Abstract*—Software Defined Wide Area Network (SD-WAN) is gaining attention in view of the integration of Non-Terrestrial Network (NTN) to guarantee a resilient connection, also in remote areas. In this paper, we present a real test-bed implementation that poses the basis for an innovative SD-WAN architecture designed for the 6G era that integrates terrestrial links and Low-Earth-Orbit (LEO) satellite connectivity. In particular, we describe a control-plane centric design that leverages ONOS as the SDN controller, Open vSwitch as the programmable data plane, and OpenFlow for southbound control, implemented and validated on a small-scale, real testbed with a Starlink connection. Moreover, we introduce a policy-driven module for link selection and failover. The ultimate goal is to provide a reproducible tutorial for deploying the SD-WAN testbed, thereby offering the scientific community a realistic experimental environment to evaluate advanced failover mechanisms and to investigate the optimal integration of LEO satellite networks.**

***Index Terms*—Software Defined Wide Area Network (SD-WAN), Non-Terrestrial Network (NTN), Starlink, real test-bed**

I. INTRODUCTION

The continuous expansion of digital services, combined with the increasing reliance on real-time applications, has led to an unprecedented demand for ubiquitous and resilient connectivity. While terrestrial cellular networks have succeeded in delivering enhanced capacity and low-latency services in densely populated and urbanized areas, they remain limited in their ability to ensure seamless coverage across rural, remote, and maritime regions, where the deployment of terrestrial infrastructure is economically and logistically challenging. This limitation raises the need for complementary solutions capable of extending network reach while maintaining reliability and quality of service.

Non-Terrestrial Networks (NTNs), and in particular Low-Earth-Orbit (LEO) satellite constellations, are emerging as a key enabler for bridging this digital divide. LEO satellites offer global coverage, reduced propagation delay compared to traditional geostationary satellites, and the potential to provide broadband services in areas traditionally underserved by terrestrial operators. Their integration with terrestrial infrastructures is not only relevant for expanding consumer broadband services but also for supporting mission-critical scenarios, such as disaster recovery operations, public safety, and military communications.

At the same time, the increasing heterogeneity of connectivity options requires a more sophisticated management layer capable of coordinating multiple underlay technologies. Software-Defined Wide Area Network (SD-WAN) [1] technologies are gaining popularity as they provide a programmable and flexible approach to orchestrate multi-path communication. By abstracting the underlying physical infrastructure, SD-WAN enables traffic steering, policy-based routing, and fast failover across diverse network links. The convergence of SD-WAN with NTNs, and particularly with LEO satellite connectivity, therefore represents a promising pathway toward resilient and adaptive networking in the 6G era [2], [3].

The next generation of mobile systems (6G) will require ubiquitous connectivity spanning terrestrial and non-terrestrial domains. LEO constellations (e.g., Starlink) offer high-capacity, low-latency links that complement terrestrial infrastructures, but their characteristics (variable latency, intermittent availability, asymmetric bandwidth) call for adaptive network architectures.

Despite this potential, the majority of existing research efforts focus on the simulation, emulation, or architectural studies of areas such as path computation, resource scheduling [4], with limited availability of reproducible experimental platforms. This lack of real-world validation hinders the ability of the research community to assess the practical challenges of SD-WAN NTN integration, such as handling variable satellite link quality, optimizing multi-path selection policies, and guaranteeing seamless failover in dynamic environments. A research accessible testbed is thus essential to enable experimentation, benchmarking, and the development of innovative mechanisms tailored to the unique characteristics of LEO satellite networks.

In this paper, we aim to address this gap by presenting a real-world implementation of an SD-WAN testbed that integrates terrestrial broadband with LEO satellite connectivity. The proposed architecture adopts a control-plane centric approach built on ONOS as the SDN controller, Open vSwitch as the programmable dataplane, and OpenFlow for southbound communication. The system has been integrated with a commercial Starlink terminal, demonstrating the feasibility of multi-path orchestration across terrestrial and non-terrestrial links. Thus, we extend the testbed with a policy-driven routing module, enabling adaptive link selection and advanced failover mechanisms.

The contribution of this work is twofold. First, we provide a reproducible tutorial and open methodology to allow researchers and practitioners to replicate and extend the proposed testbed in different contexts. Second, we establish a realistic experimental environment that can serve as a foundation for future investigations on resilient multi-path communication and the optimized integration of LEO satellite networks into the 6G ecosystem.

II. RELATED WORK

Previous research has investigated how SD-WAN solutions can be leveraged to enable multipath communication across heterogeneous underlay networks [5]. For example, the study in [4] introduces an SD-WAN framework that integrates terrestrial and non-terrestrial networks (TN/NTN) and employs AI-driven tunnel selection to enhance both reliability and performance. Nonetheless, most existing works remain limited to simulation-based evaluations or high-level architectural proposals, with little emphasis on validating deployments over real LEO satellite infrastructures.

SD-WAN frameworks and SDN controllers have been extensively studied; however, the explicit integration of LEO satellite links within SD-WAN testbeds is less explored. Prior surveys on SD-WAN evolution motivate the need for policy-centric orchestration and edge intelligence. Practical works on Open vSwitch and ONOS demonstrate the feasibility of controller-based flow programming. This paper builds on those foundations and contributes an applied integration with a commercial LEO uplink (Starlink) and an edge-level, script-based failover mechanism validated in live experiments.

Authors in [6] propose a novel approach that embeds satellite constellation awareness into the SD-WAN Customer Premises Equipment (CPE). By integrating a predictive simulation engine, their system anticipates satellite movements to enable proactive routing decisions, thereby optimizing the use of LEO connectivity without requiring modifications to the satellite infrastructure and remaining transparent to end-users. The main contribution lies in introducing an intelligent edge-based management strategy for SD-WAN deployments that incorporates dynamic LEO links. Nonetheless, the evaluation is conducted in a simulated environment rather than on a real-world testbed, which limits the immediate validation of its practical effectiveness.

While research attention remains limited, commercial activity is much more pronounced. A number of SD-WAN vendors and satellite providers already promote offerings that integrate LEO connectivity [7]–[9]. These are often marketed for their extended coverage, high throughput, and multipath resilience. However, public documentation rarely discloses the underlying technical methods, leaving uncertainty about how such systems cope with unique LEO dynamics, beyond conventional link aggregation or failover techniques.

III. ARCHITECTURE

This section describes the implementation of an SDN architecture for the integrated management of terrestrial and non-terrestrial networks. The goal is to demonstrate how the combination of tools such as Open vSwitch, the ONOS controller, and the use of the OpenFlow protocol enables centralized, dynamic, and scalable network management. The architecture was tested in a hybrid context, where a node acts as a gateway to the Starlink satellite network, simulating practical integration between terrestrial and space domains.

The architecture focuses on the following two SD-WAN layers:

1) Control Layer (ONOS): An ONOS instance maintains topology, computes paths across heterogeneous domains, and issues OpenFlow rules. ONOS exposes northbound APIs for intent management and enables reactive packet forwarding.
2) Data Layer (OVS + Edge Gateway): OVS instances perform flow forwarding. An edge gateway (MiniPC3 in the testbed) acts as an L3/NAT gateway and local failover agent between Starlink and terrestrial uplinks.

To validate this architecture, we deployed a real testbed implementation on three Mini PCs, each equipped with an Intel Core i5 CPU, 16 GB DDR5 RAM, and a 512 GB SSD. Devices provide dual Gigabit Ethernet interfaces, integrated Wi-Fi, and run Ubuntu 22.04 LTS. The testbed implements a single SDN-based domain to represent, for example, a remote site of an SD-WAN. The SDN infrastructure was deployed using Open vSwitch (OvS) as the data plane component, emulating OpenFlow-enabled switches/hosts, and ONOS as the centralized SDN controller, communicating via the OpenFlow 1.3 protocol. In this setup (Fig. 1 and Fig. 2), two of the Mini PCs operated as OvS switches/hosts, while one acted as a gateway with two external interfaces (Starlink and a terrestrial uplink). The OvS bridges were created and configured to support OpenFlow 1.3, with ONOS reachable by the bridges. Internal connectivity used the 10.10.10.x/24 IP addressing scheme. The choice of OvS and ONOS stems from their proven reliability and wide adoption in the SDN community: OvS provides efficient virtualization of advanced switch functionalities, while ONOS offers scalability, modularity, and centralized network control.

The configuration involved in each MiniPc:

1) OvS installation,
2) creation and configuration of a virtual bridge,
3) addition of virtual interfaces: two interfaces were dedicated to point-to-point connections with other nodes, while a third was configured as a loopback interface with a private IP address, allowing the MiniPc to act both as switch and as host (Fig. 4), and
4) deployment of the ONOS controller on a local server, and connection of the OvS to ONOS.

The overall configuration was validated using the ONOS graphical interface, which automatically discovered the OpenFlow switches and hosts, confirming the triangular network topology and the correct state of links and IP assignments (Fig. 5).

Finally, MiniPC3 was configured as an edge gateway towards both terrestrial and satellite domains. By enabling IPv4 forwarding and configuring NAT functionality, it operated as an L3/NAT gateway and local failover agent, ensuring seam-

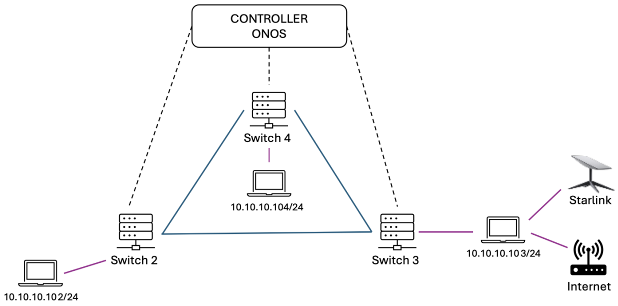


Fig. 1: Physical topology of the SDN testbed: schematic view

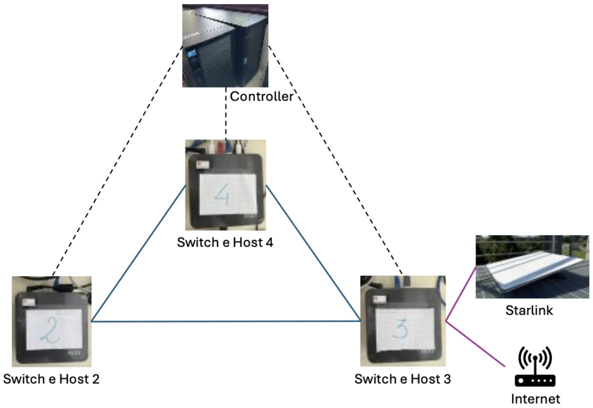


Fig. 2: Physical topology of the SDN testbed: real implementation

less traffic forwarding between the Starlink satellite uplink and terrestrial connections.

Fig.4 and Fig.2 illustrate the logical and physical architecture deployed for our experiments with a Starlink gateway located in Pisa, Italy. Fig.3 shows a real photo of the testbed with three MiniPCs (MiniPc2, MiniPc3, MiniPc4) interconnected.

A. *Logical Topology*

As shown in Fig. 4, three mini-PCs (MiniPc2, MiniPc3, and MiniPc4) are interconnected to form a triangular topology. Host2 and Host4 represent generic hosts of the SD-WAN site, while Host3 represents the gateway of the SDN site. Each Mini-PC is equipped with multiple network interfaces: one dedicated to host connectivity and others used for inter-switch communication. The Starlink gateway provides upstream Internet connectivity, which is shared among the three hosts.

Fig. 3: MiniPC nodes employed in the SDN testbed implementation

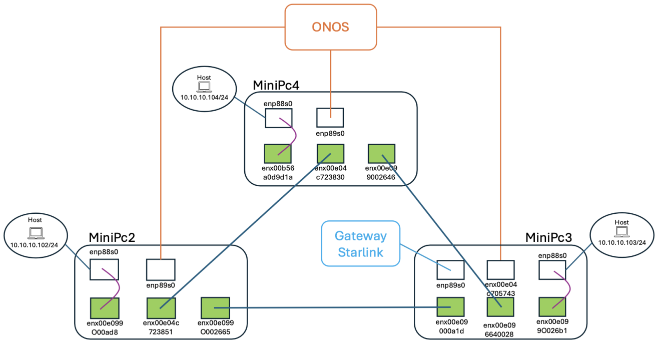
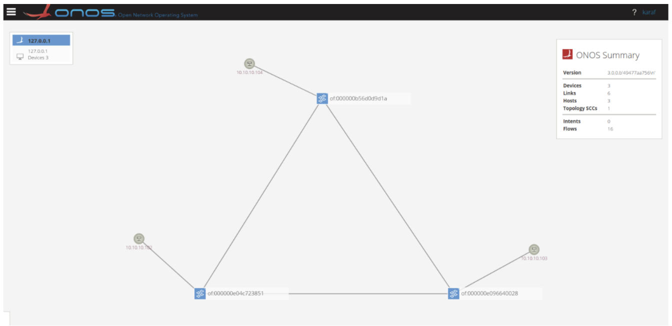


Fig. 4: Logical schema of the interface connections in the SDN testbed, including ONOS controller and Starlink gateway



ONOS Summary

Version 3.0.0.b'49477aa756\n'

Devices 3
Links 6
Hosts 3
Topology SCCs 1

Intents 0
Flows 16

Fig. 5: SDN topology detected by ONOS

The ONOS controller maintains global knowledge of the topology and communicates with the mini-PCs via dedicated management interfaces. Data-plane forwarding is realized by means of Open vSwitch (OvS) instances running on each mini-PC, allowing ONOS to enforce routing and failover policies. Host devices are attached to the mini-PCs to generate traffic flows and validate end-to-end connectivity.

Switch 3 is connected to the Starlink user terminal installed in Pisa (Italy), providing real-world LEO satellite backhaul access to the testbed. Switch/Host 4 acts as the central aggregation node, bridging traffic between the other switches and the Starlink gateway. The ONOS controller is hosted on a separate server, connected through management links to all mini-PCs.

The use of Starlink as the backhaul introduces satellite-specific impairments such as variable latency, jitter, and periodic handovers between satellites, occurring every 15 seconds. These dynamics are directly observable in the data plane and allow us to test SDN-based routing and failover strategies under realistic satellite network conditions. By integrating ONOS with the programmable mini-PCs, the system can react to performance degradation by rerouting traffic or activating backup interfaces, as detailed in Section IV.

IV. FAILOVER MECHANISMS AND EVALUATION

In this section, we illustrate a practical application of the implemented testbed described previously. Specifically, we demonstrate a simple yet effective failover mechanism. While the design and evaluation of a fully-featured, advanced failover strategy lie beyond the primary scope of this work, we implement a basic tunnel selection mechanism to provide a concrete example of how the testbed can support failover functionality in a real-world scenario

A. Failover algorithm implemented

Algorithm 1 is executed in Host3 to select the best performing tunnel.

I_p, G_p : primary interface and gateway
I_s, G_s : secondary interface and gateway
Δt : probe interval
L_{th} : packet-loss threshold
T_{th} : latency threshold
τ : hold time before revert (10 s)
$s(t) \in \{p, s\}$: active interface at time t
$\ell(t),\ \rho(t)$: measured loss (%) and latency (ms) at time t

Algorithm 1 Failover Backup Algorithm

```
Require: $I_p, G_p, I_s, G_s, \Delta t, L_{\text{th}}, T_{\text{th}}, \tau$
$s \leftarrow p$;   $t_{sw} \leftarrow -\infty$ {start on primary}
set_default($G_p, I_p$)
loop
    wait($\Delta t$)
    $(\ell, \rho) \leftarrow$ probe(ip.address, iface=$s$) {single probe}
    if $s = p$ then
        if $\ell \neq \text{N/A} \wedge \ell \geq L_{\text{th}} \vee \rho \neq \text{N/A} \wedge \rho > T_{\text{th}}$ then
            set_default($G_s, I_s$);   $s \leftarrow s$;   $t_{sw} \leftarrow \text{now}()$ {switch to secondary}
        end if
    else{$s$ = secondary}
        if $\text{now}() - t_{sw} \geq \tau$ then
            set_default($G_p, I_p$);   $s \leftarrow p$ {return to primary after hold $\tau$}
        end if
    end if
end loop=0
```

The proposed failover algorithm manages the selection between a *primary* and a *secondary* network interface based on real-time performance measurements. Let (I_p, G_p) denote the primary interface and gateway, and (I_s, G_s) the corresponding secondary. The algorithm periodically probes a remote host (e.g., Google DNS at 8.8.8.8) every Δt seconds to collect packet loss $\ell(t)$ and latency $\rho(t)$.

When operating on the primary interface, a switch to the secondary is triggered if either the packet loss exceeds a threshold L_{th} or the latency surpasses T_{th}. Upon switching, the system records the switch time t_{sw} and routes traffic via the secondary interface.

While operating on the secondary, the algorithm periodically checks whether the hold time τ has elapsed since the last switch. If so, it reverts to the primary interface to restore the preferred routing configuration.

This mechanism ensures fast reaction to degraded performance while avoiding oscillations through the introduction of a minimum hold time. The pseudocode in Algorithm 1 summarizes the procedure in a compact and implementation-oriented form.

B. Experiment Design and Results

Two classes of experiments were executed:

- Short-threshold failover (25 ms): Gateway pings 8.8.8.8 and switches to terrestrial link when Starlink RTT spikes above 25 ms.
- Relaxed-threshold transatlantic test (115 ms): Gateway pings a remote server in NYC and uses a higher threshold to test behaviour across large RTT variance.

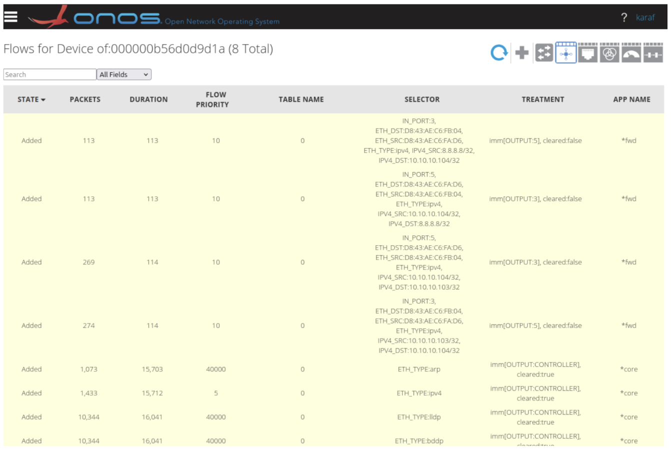
In particular, the metrics we collected were: per-probe RTT, packet loss %, timestamps of route switches, and ONOS flow tables readouts.

The local failover agent produces fast route switching in response to transient degradations; the testbed logs show consistent switching events correlated with RTT peaks. Specifically, Algorithm 1, running on Host3 (gateway), continuously probes a remote host (Google DNS 8.8.8.8) and monitors packet loss and RTT. When latency exceeds a predefined threshold or packet loss is detected, the algorithm updates the system's default route, switching traffic from the primary Starlink interface to the terrestrial backup.

The ONOS controller reactively installs flow rules when traffic is detected from one of the hosts. The flow rules match on input port, MAC, and IP addresses. Fig. **??** illustrates the flow rules installed in Switch4. Besides management rules directing new traffic flows to the controller, there are four flow rules. The two rules at the top match ping traffic from Host4 to the Google DNS server; the other two flow rules match the traffic between Host4 and Host3, which acts as the gateway of the LAN.

The combined approach avoids prolonged outages and keeps end-to-end reachability for test workloads, at the cost of occasional increased latency during switch events (visible as spikes). The telemetry series illustrate the latency variations and the thresholds used.

Fig. 7 highlights this behaviour for the 25 ms test, where Starlink is used as the primary link. The RTT time series displays sharp peaks above the threshold (orange line), which trigger an immediate switchover to the terrestrial path. After each peak crossing the orange line, the gateway stays on the terrestrial link for about 10 seconds before automatically reverting to the Starlink link, showing the gateway's rapid yet stable reaction to transient degradations. This confirms that low thresholds yield very responsive failover at the expense of more frequent switching events.



Flows for Device of:000000b56d0d9d1a (8 Total)

STATE	PACKETS	DURATION	FLOW PRIORITY	TABLE NAME	SELECTOR	TREATMENT	APP NAME
Added	113	113	10	0	IN_PORT:3, ETH_DST:D8:43:AE:C6:FB:04, ETH_SRC:D8:43:AE:C6:FA:D6, ETH_TYPE:ipv4, IPV4_SRC:8.8.8.8/32, IPV4_DST:10.10.10.104/32	imm[OUTPUT:5], cleared:false	*fwd
Added	113	113	10	0	IN_PORT:5, ETH_DST:D8:43:AE:C6:FA:D6, ETH_SRC:D8:43:AE:C6:FB:04, ETH_TYPE:ipv4, IPV4_SRC:10.10.10.104/32, IPV4_DST:8.8.8.8/32	imm[OUTPUT:3], cleared:false	*fwd
Added	269	114	10	0	IN_PORT:5, ETH_DST:D8:43:AE:C6:FA:D6, ETH_SRC:D8:43:AE:C6:FB:04, ETH_TYPE:ipv4, IPV4_SRC:10.10.10.104/32, IPV4_DST:10.10.10.103/32	imm[OUTPUT:3], cleared:false	*fwd
Added	274	114	10	0	IN_PORT:3, ETH_DST:D8:43:AE:C6:FB:04, ETH_SRC:D8:43:AE:C6:FA:D6, ETH_TYPE:ipv4, IPV4_SRC:10.10.10.103/32, IPV4_DST:10.10.10.104/32	imm[OUTPUT:5], cleared:false	*fwd
Added	1,073	15,703	40000	0	ETH_TYPE:arp	imm[OUTPUT:CONTROLLER], cleared:true	*core
Added	1,433	15,712	5	0	ETH_TYPE:ipv4	imm[OUTPUT:CONTROLLER], cleared:true	*core
Added	10,344	16,041	40000	0	ETH_TYPE:lldp	imm[OUTPUT:CONTROLLER], cleared:true	*core
Added	10,344	16,041	40000	0	ETH_TYPE:bddp	imm[OUTPUT:CONTROLLER],	*core

Fig. 6: Flow Entries on device Switch 4.

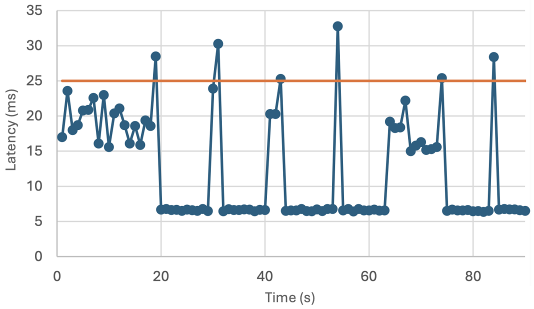


Fig. 7: Latency trend for short-threshold failover (25 ms) where Host 3, acting as the gateway, pings 8.8.8.8.

Fig. 8 shows the transatlantic case with a 115 ms threshold: the higher limit allows the Starlink link to absorb greater natural RTT variation before triggering failover. Similarly, when the threshold is crossed, the traffic is moved to the terrestrial path for roughly 10 seconds and then switched back to Starlink once the RTT stabilises below the threshold, reducing oscillations and illustrating the trade-off between stability and responsiveness.

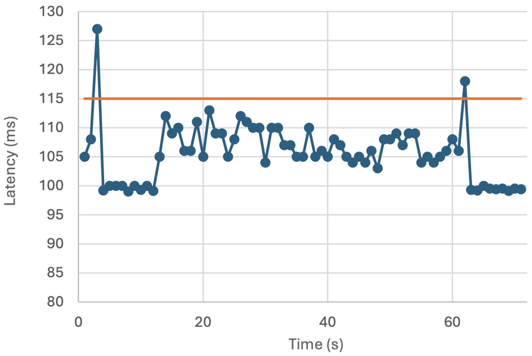


Fig. 8: Latency evolution for relaxed-threshold transatlantic test (115 ms) with gateway pinging NYC server

V. CONCLUSION

We have presented a practical SD-WAN architecture and a validated, small-scale implementation integrating terrestrial and LEO satellite links. The combination of ONOS, Open vSwitch, and a pragmatic edge failover agent demonstrates that programmable control can effectively coordinate heterogeneous uplinks for resilience and QoS in 6G-oriented deployments. While the testbed is intentionally modest, the lessons and artifacts (scripts, OVS/ONOS configuration sequences, telemetry traces) provide a reproducible foundation for future scaling and research into predictive orchestration and service slicing, also in the view of applying Machine Learning approaches [10].

ACKNOWLEDGMENT

This work was partially supported by the European Union - Next Generation EU under the Italian National Recovery and Resilience Plan (NRRP), Mission 4, Component 2, Investment 1.3, CUPE83C22004640001 partnership on "Telecommunications of the Future" (PE00000001 - program "RESTART") Project WATCHEDGE and by the Italian Ministry of Research (MUR) in the framework of the CrossLab and Forelab Projects (Departments of Excellence).

REFERENCES

[1] S. Troia, L. Borgianni, G. Sguotti, S. Giordano, and G. Maier, "A comprehensive survey on software-defined wide area network," *IEEE Communications Surveys Tutorials*, pp. 1–1, 2025.

[2] M. M. Azari, S. Solanki, S. Chatzinotas, O. Kodheli, H. Sallouha, A. Colpaert, J. F. M. Montoya, S. Pollin, A. Haqiqatnejad, A. Mostaani *et al.*, "Evolution of non-terrestrial networks from 5g to 6g: A survey," *IEEE communications surveys & tutorials*, vol. 24, no. 4, pp. 2633–2672, 2022.

[3] W. Jiang, B. Han, M. A. Habibi, and H. D. Schotten, "The road towards 6g: A comprehensive survey," *IEEE Open Journal of the Communications Society*, vol. 2, pp. 334–366, 2021.

[4] L. Borgianni, D. Adami, and S. Giordano, "Optimizing network performance and reliability with an integrated sd-wan and satellite 6g architecture," in *2023 2nd International Conference on 6G Networking (6GNet)*, 2023, pp. 1–4.

[5] S. Troia, G. Maier, and S. Bregni, "Experimental evaluation of sd-wan performance in a municipal network test bed," in *2023 IEEE Latin-American Conference on Communications (LATINCOM)*. IEEE, 2023, pp. 1–5.

[6] G. Sguotti, S. Troia, and G. Maier, "Mitigating satellite link variability in sd-wan using constellation-aware edge routing," in *2025 25th Anniversary International Conference on Transparent Optical Networks (ICTON)*, 2025, pp. 1–5.

[7] J. Chung. (2024) Unlock the power of low earth orbit satellites with vmware velocloud sd-wan. Last access: 08/09/2025. [Online]. Available: https://news.broadcom.com/sase/unlock-the-power-of-low-earthorbit-satellites-with-vmware-velocloud-sd-wan

[8] SES. (2020) Multi-access sd-wan anywhere. Last access: 08/0492025. [Online]. Available: https://www.ses.com/sites/default/files/2020-02/SD-WAN%20InsightPaper%20Fixed-data.pdf

[9] G. Melesio, J. Schoch, and J. Carranza. (2022) Versa sd-wan solution use case for satellite isps. Last access: 08/09/2025. [Online]. Available: https://versa-networks.com/documents/whitepapers/versa-sd-wan-solution-use-case-for-satellite-isps.pdf

[10] M. Khalid, J. Ali, and B.-h. Roh, "Artificial intelligence and machine learning technologies for integration of terrestrial in non-terrestrial networks," *IEEE Internet of Things Magazine*, vol. 7, no. 1, pp. 28–33, 2024.