Chat-IBN-RASA: Building an Intent Translator for Packet-Optical Networks based on RASA

Celso H. Cesila[1], Rossano P. Pinto[1], Kayol S. Mayer[2], Andrés F. Escallón-Portilla[3],
Darli A. A. Mello[2], Dalton S. Arantes[2], Christian E. Rothenberg[1]
School of Electrical and Computer Engineering
University of Campinas (UNICAMP), Campinas, Brazil
[1] {ccesila, rossano, chesteve}@dca.fee.unicamp.br
[2] {kayol, darli, dalton}@unicamp.br
[3] a218682@dac.unicamp.br

2023 IEEE 9th International Conference on Network Softwarization (NetSoft) | 979-8-3503-9980-6/23/$31.00 ©2023 IEEE | DOI: 10.1109/NetSoft57336.2023.10175491

***Abstract*—This work presents Chat-IBN-RASA, a conversational AI chatbot based on the open-source RASA framework, acting as an Intent Translator component for Intent-Based Networking (IBN) architectures. The IBN-driven RASA chatbot allows users to interact with the network management system using high-level language communication without the need for in-depth technical knowledge of the network. The chatbot is trained using Natural Language Understanding (NLU) models, a defined domain, stories, and custom actions for API communication and database queries. We present the prototype implementation in a use case of survivability intents in packet-optical networks. The custom actions featured include communication with the network database, path computation, and the recommended path for intent deployment, considering availability, protection type, data rate, and link distances.**

***Index Terms*—Intent-based Networking, IBN, chatbot, NLU**

I. Introduction

The management of computer networks is becoming increasingly complex with the emergence of new technologies such as Industry 4.0, IoT, edge/fog computing, and 5G networks. One promising solution for addressing this issue is the adoption of Intent-Based Networking (IBN) approaches that rely on an abstraction layer over the network infrastructure. This approach allows users to use a high-level language to express their network intents, which an autonomic framework can then translate into network configuration [1]. Additionally, this framework can also take closed-loop remediation actions in the event that these intents are not met. One key piece to deliver the envisioned degrees of abstraction in IBN architectures is a suitable translation component to convert natural language into lower-level configurations and commands.

As we look for more natural ways to incorporate automation into our daily routines, conversational systems are becoming increasingly common for human-computer interaction. Recent popular examples of conversational AI include specific ones like Siri from Apple and Alexa from Amazon, as well as more general AI language models such as ChatGPT from OpenAI, which have gained considerable attention in recent times.

In this work, we present Chat-IBN-RASA as a platform for building conversational AI chatbots to develop intent translation components for IBN architectures. With the open-source RASA framework, it is possible to create chatbots utilizing AI and enable self-training with minimal data, which streamlines the system. RASA gives users the flexibility to tailor the chatbot to their specific needs [2]. The proposed intent translation component allows users without an in-depth technical knowledge of the network to efficiently manage the network through high-level language communication.

The paper is organized as follows: (II) presents background and related work, including the main features of IBN and the RASA framework, along with related research efforts; (III) describes the proposed Intent Translator architecture; (IV) addresses the prototype implementation; (V) shows the proof-of-concept (PoC) use case; (VI) presents the points that still need to be addressed and explored; and (VII) presents the conclusion of this work.

II. Background and Related Work

A. Intent-based Networking (IBN)

Definitions for IBN can be found in several works, whereas [1] describes IBN as a novel networking paradigm that automates the process of network configuration. Users can describe their functional goals using a high-level API or language, and the intent system adapts the network configuration to deliver these goals.

The IBN approach specifies management or design changes, describing the new intended end state of the network (*what the network should be like*) rather than prescribing the sequence of modifications to bring the network, via the definition of specific commands and protocols, to that final state (*how to achieve network changes*). The IBN tends to be robust in scale since the intent is perennial, i.e., stable over time, even when the low-level state of the network elements changes rapidly. One of the principles of an IBN is to offer autonomous behavior between monitoring processes (network measurements), analysis, and control of the network, ensuring that the operator's intentions are satisfied.

Towards the realization of IBN, it is also observed, as an enabler element, the application of techniques of artificial intelligence (AI) and machine learning (ML). Advances in ML/AI are at the forefront of designing and operating communication networks of any nature [3]. It is possible to envision the evolution of SDN networks with the adoption of

AI/ML to generate knowledge, especially to: (i) assist in the decisions of human operators or (ii) automate decision and control processes according to the intent that describes the expected behavior of the network.

Finishing the contextualization of IBN, multiple standardization groups discuss key concepts and principles of IBN, and the consequent impact on technical specifications is expected. Recently, the IETF NMRG (Internet Engineering Task Force Network Management Research Group) published an RFC 9315 [4] that presents the concept of Intent and Intent-based Management, the distinction between intent and policy, and the functionalities of IBN. The TM Forum Autonomous Network Project was also designed as an intent-focused group, presenting the definition and properties of intent and a framework focused on the scope of intent.

B. RASA Framework

The RASA framework is an open-source platform for building conversational AI chatbots. It utilizes artificial intelligence and machine learning to enable self-training with minimal data, making it a simple and flexible option for creating chatbots. The framework's architecture allows for integration with multiple data sources, allowing users to tailor the chatbot to their specific needs [5].

The design of RASA is modular, allowing easy integration with other systems. The RASA stack has two main components: Core and NLU. RASA Core can function as a dialogue manager in partnership with NLU services aside from RASA NLU. Although the code is written in Python, both services offer HTTP APIs, making them accessible to projects utilizing other programming languages. The dialogue state is kept in a tracker object, with one tracker per conversation session. The tracker is the only component that holds the state. It saves slots and the history of events in a conversation. The state of a conversation can be reconstructed by replaying all events [6].

The RASA Core system predicts the next step in conversation at each iteration by choosing from a pre-defined list of actions. These actions can be simple messages sent to the user or custom functions executed to respond to the user's intent. Custom actions can be defined in the actions.py file in the actions folder, allowing for personalized responses to user intent. The response can come from the domain.yml file where an action is defined, or it can be created through custom actions by returning an action name from a function. The returned action name is then treated as the action to take. The function that returns the action name is called whenever the action is referenced in the system [7].

The NLU component of RASA contains intents that are classified by name and associated with specific sentence patterns. During the training process, these intents are represented numerically, and the sentence patterns are converted into vectors using the fastText3 tool. RASA considers the last five previous states (as determined by the `max_history` parameter). It converts input data into binary vectors of combined intents, entities, slots, and actions. The following action is represented as a one-hot vector based on the total number of actions. This information is then passed to the LSTM model for training to predict future actions [8].

RASA is one of the main available platforms for building machine learning-based chatbots, such as Microsoft Bot Framework, Google DialogFlow, among others. We chose RASA for several reasons: it is an open-source NLP tool, self-hosted with local run capability, and offers adaptability and data control. RASA's custom actions with exposed APIs enable human and automation interactions with the chatbot and IBN system, simplifying the process for both.

C. Related Work

Several works provide a holistic view of the IBN approach, including [1], [6], [9], [10], [16]. Next, we focus on relevant related work with a focus on intent translation.

In [11], the authors present an intent system with new intent primitives designed to integrate human processes into future networks and reduce the complexity of network intent expression. The system has a conversational intent expression interface using AI and NLP tools. The input is processed using Dialogflow, an NLP service from Google that can interact with various online services. The system automatically translates user intents into network configurations. The authors argue that the new intent primitives and a reduced cognitive load for non-technical users will increase efficiency and reduce ambiguity in user input.

The work in [12] introduces an intent-refinement process for translating network configurations from natural language utterances using machine learning and operator feedback. The process consists of three stages: using a DialogFlow chatbot to extract main actions and targets from the user's intent and a neural sequence-to-sequence model to translate the extracted entities into a structured network definition program in Nile. Their key contributions include a novel intent-refinement process, Nile as a comprehensive intent definition language that acts as an abstraction layer, and improved translation accuracy through operator feedback.

In [13], the authors propose an IBN System that uses voice assistance for network management and visual representation for real-time network topology and intents. The system is based on Software-Defined Networking (SDN) concepts and was developed using the Python programming language, Django framework, Nginx, and MariaDB. Amazon Echo Dot was used to input voice commands which were then processed by the Speech Recognition system and Intent Engine to convert intents into network configurations and pushed to the infrastructure using the OpenFlow protocol.

Unlike works [11] and [12], we adopt RASA as the main tool of our solution, thus benefiting from a self-hosted open-source code base with adaptability and data control. Furthermore, unlike [13], our approach offers easy chatbot management, tool integration, and a significant advantage in custom actions that expose an API for both human interactions through the chatbot and interactions from other components of the IBN system, simplifying the process for both human and automation components of the network.

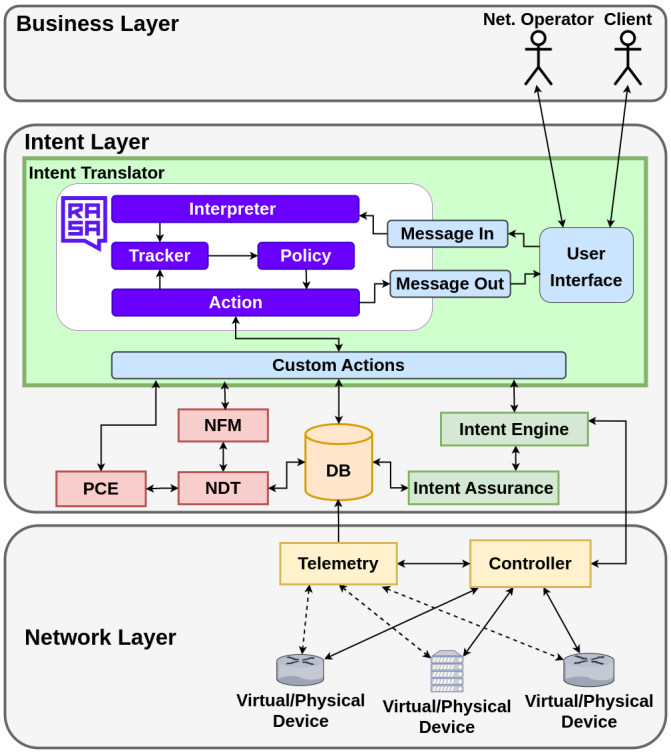

Fig. 1. IBN architecture with a focus on the Intent Translate component.

III. Chat-IBN-RASA Intent Translator Architecture

The IBN architecture is illustrated in Fig. 1 and comprises three layers: Network, Intent, and Business Layers. The Intent Layer oversees all processes related to the installation, management, and monitoring of the network to meet user intent.

The Intent Translator service, located at the top of the solution, interacts with the user to gather desired intent characteristics via natural language communication facilitated by a chatbot utilizing conversational AI technology from RASA.

Once all features have been collected, the Path Computation Element (PCE) interacts with a Network Digital Twin (NDT) [14] to selects all feasible lightpaths between the source and destination. The candidate connections are passed on to the Intent Translator, which then directs them to the Network Failure Management (NFM) module. This module records all network failures and adjusts network elements' failure and repair rates (e.g., fibers, reconfigurable add-drop multiplexers, transponders) based on previous history. The Intent Translator determines the best solution based on the NFM information and then sends a response to the client for approval. Upon confirmation, the Intent Engine communicates the necessary settings to the Controller to execute the corresponding commands on the physical devices. The client/operator is notified of the intent installation result. If successful, the Intent Assurance Engine (IAE) is configured to continuously monitor intent validity and fulfillment using the streaming telemetry data from network elements.

IV. Prototype Implementation

The Intent Translator acts as an interface between the client/operator and the IBN system, being responsible for abstracting the communication from a high to a low level and structuring the information so the Intent Engine can handle the requested system intents.

As argued before, our implementation choice for the Intent Translator is to leverage the open-source RASA framework that supplies the building blocks for creating virtual assistants. More than the intent translation aspects, this component focuses on the communication aspects with the user through a virtual assistant, delivering an experience in which the customer can feel that the IBN system not only reproduces the intended intention but also suggests improvements and interacts with the user.

To this end, we developed a chatbot around a service form in which the user chooses which intention he/she would like to execute on the network. The chatbot asks for the user to inform the needed details to provide the intention and interacts with the user, and the IBN system throws APIs to access database and network information to give intent configuration options. Next, the user provides all the necessary information, RASA structures all the information, and sends a JSON-formatted descriptor to the Intent Engine with everything needed for resource provisioning and intent execution.

The following main implementation guides were adopted:

Natural Language Understanding. The NLU model transforms user messages into structured data. To this end, we provided training examples that show how RASA should understand user messages and then trained a model showing several examples and building a natural language model for the context of IBN requirements.

Domain definition. This aspect is related to the Domain specification in RASA to create the chatbot response texts. The Domain defines the universe in which the assistant operates, specifying the intents, entities, slots, responses, forms, and actions the bot should know. It also defines a configuration for chat sessions through the domain.yml file in the RASA structure. These configurations allow the chatbot to react to the user input with actions, responses, or both.

Story Training. With all the chatbot elements defined, it is necessary to create Stories – data type used to train the assistant's dialog management model. Stories can be used to train models capable of generalizing conversational paths. In this implementation, we create more than 30 stories variations of the same workflow in the stories.yml to initialize the learning model. When the RASA learning model is active, it is able to complement the stories.yml file according to the development of the training.

Custom Actions. The last main implementation guide refers to the steps a chatbot can take, such as making an external API call, querying a database, and other actions via Python code. To create custom actions, it is necessary to edit the actions.py file that is present in the RASA code base. We implemented

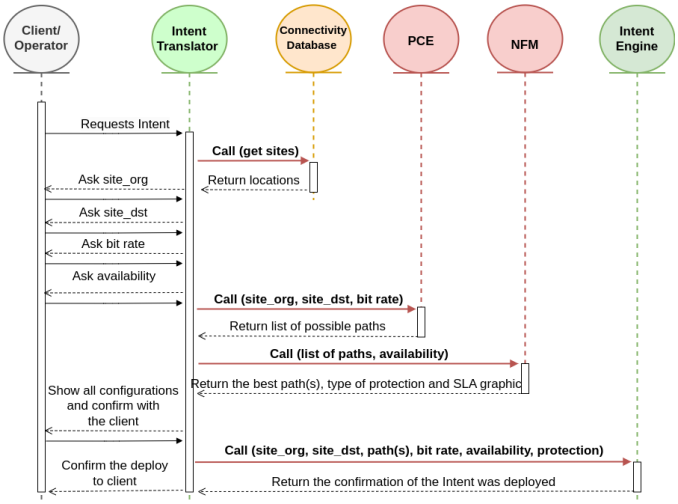

Fig. 2. Use case information workflow.

a set of relevant custom actions in our solution delivering a valuable contribution to the IBN field and RASA community.[1]

The initial custom actions that were developed were `site_org` and `site_dst`, which initiate an API communication with the network database to retrieve the names of cities where network links can be established. Another custom action, `get_paths`, collects the user-specified information, communicates with the Path Computation Element (PCE) to obtain a list of potential paths between two locations, and then forwards the list of paths to the Network Failure Management (NFM) component for assessment of path availability. Based on the availability information, the action generates a list of the best paths, prioritizing paths with the highest availability and lowest computational resources. The output of the action is the recommendation of the optimal path (considering both availability and computational resources) to the user. If the user desires to view all path options, the `others_paths` custom action can be activated to display all paths that meet the specified availability criteria. Once all necessary information has been gathered for the intent deployment, the `send_intent_engine` custom action is triggered. This action assembles the required data for the intent deployment in the IBN system and communicates with the Intent Engine component to carry out the intent installation.

V. Use Case Scenario

We now focus on a Proof of Concept (PoC) use case of our IBN architecture to illustrate candidate usage workflows. The selected scenario presents a situation in which the client intends to connect two sites through a link in which he/she specifies the general parameters of the connection. Figure 2 presents the information workflow of the use case. Additional details are presented in [15].

The Intent Translator service communicates with users via natural language to collect the required intent characteristics through a UI interface chatbot utilizing conversational AI based on the RASA framework. Figure 3 presents a conversation example between the user and the chatbot.

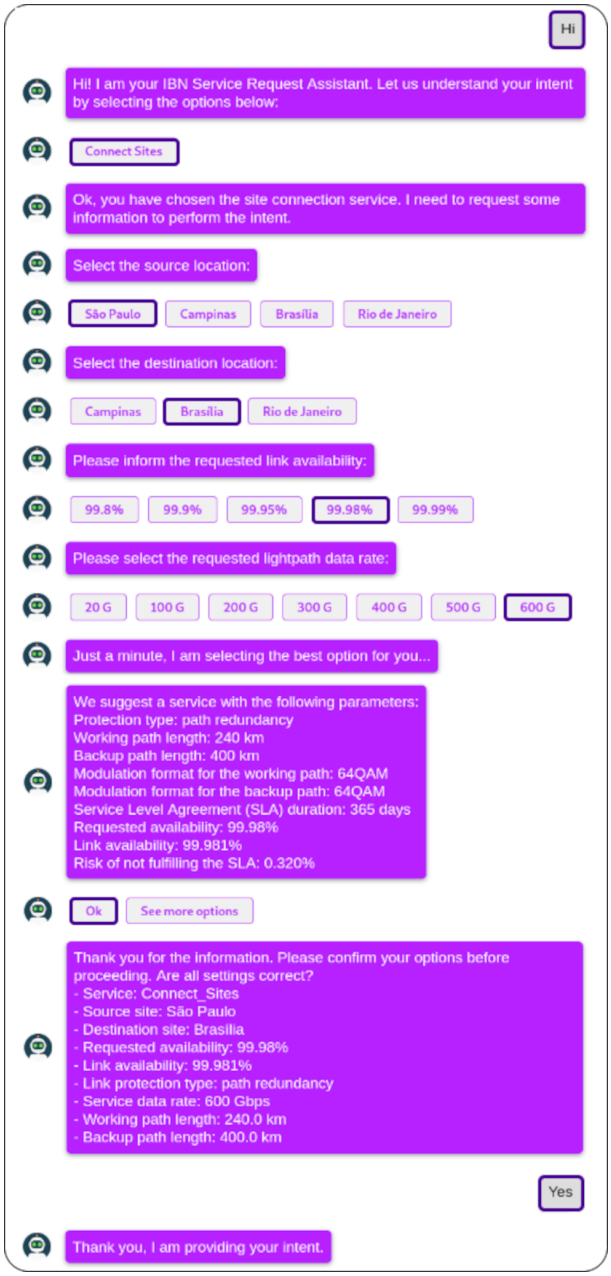

Fig. 3. Conversation example between the user and the chatbot.

The process of establishing a connection between two sites in the system involves several steps. Firstly, the user initiates the conversation and selects the intent **Connect Sites**. The Intent Translator queries the connectivity database to identify the locations of the available sites that can be connected. The user then selects the desired **source** and **destination locations** for the connection. Subsequently, additional parameters such as the **data rate** and link **availability** are requested from the user.

[1]The open-source release of our code contributions is pending approval by Padtec S/A, our industrial funding partner

The Intent Translator then communicates with the Path Computation Element (PCE), passing on the information about the source site, destination site, and the desired data rate of the connection. The PCE performs a calculation and returns a list of feasible paths that meet the specified parameters to the Intent Translator. This information, along with the user-selected availability, is then passed on to the Network Failure Management (NFM) module. The NFM module evaluates the available paths, selecting those that fulfill the requested availability while minimizing the risk of failure.

The Intent Translator classifies the available paths based on various parameters, such as **availability, protection type, data rate, and distance** of the link, and presents the information to the user in a list format. The user can either accept the suggested path or request to see additional options. Upon confirmation from the user, the Intent Translator forwards the necessary information to the Intent Engine, which configures the controller to execute the required commands on the physical devices. The Intent Engine notifies the Intent Translator once the configuration is complete and informs the user of the successful completion of the intent installation.

VI. Future Work

The Intent Translator component has the potential for further improvements by exploring advanced NLP techniques such as sentiment analysis and entity recognition, which can enhance the natural language processing capabilities of the system. This will result in a better understanding of user requests and more relevant suggestions.

The integration of additional network performance data and other relevant metrics can improve the decision-making for suggesting paths between two sites.

Incorporating additional channels for user interaction, such as voice-based interaction, and using maps to visualize the network paths are UI/UX improvements for the chatbot that would make it more user-friendly.

At present we are working out software architecture design options to integrate the Chat-IBN-RASA component with a commercial network management system for packet-optical networks.

VII. Conclusions

This work focuses on developing the Intent Translator component for Intent-Based Networking (IBN) architectures using the open-source RASA platform. The Intent Translator allows users to interact with the IBN system in a more natural, high-level language with the help of a virtual assistant created using RASA. The system is based on four main implementation guides: NLU model, Domain, Stories, and Actions. The custom actions developed during this work enable the chatbot to communicate with the network database, assess path availability, and recommend the optimal path for the user. This work demonstrates the benefits of using conversational AI chatbots in network management and provides a flexible, open-source solution for users who want to manage their networks through high-level language communication.

Acknowledgment

This study was partially funded by CAPES, Brazil - Finance Code 001 and by Padtec S/A.

References

[1] Zeydan, E., & Turk, Y. (2020). Recent Advances in Intent-Based Networking: A Survey. In 2020 IEEE 91st Vehicular Technology Conference (VTC2020-Spring) (pp. 1-5). doi: 10.1109/VTC2020-Spring48590.2020.9128422.

[2] Meshram, S., Naik, N., VR, M., More, T., & Kharche, S. (2021). College Enquiry Chatbot using Rasa Framework. In 2021 Asian Conference on Innovation in Technology (ASIANCON) (pp. 1-8). doi: 10.1109/ASIANCON51346.2021.9544650.

[3] Wang, S., Qureshi, M. A., Miralles-Pechuaán, L., Huynh-The, T., Gadekallu, T. R., & Liyanage, M. (2021). Explainable AI for B5G/6G: Technical Aspects, Use Cases, and Research Challenges. arXiv preprint arXiv:2112.04698.

[4] Clemm, A., Ciavaglia, L., Zambenedetti Granville, L., & Tantsura, J. (2022, October). Intent-Based Networking - Concepts and Definitions (No. RFC 9315). RFC Editor. https://doi.org/10.17487/RFC9315.

[5] Joshi, M., & Sharma, R. K. (2020). An Analytical Study and Review of Open Source Chatbot Framework, RASA. International Journal of Engineering Research & Technology (IJERT), 9(06), June 2020.

[6] Gomes, P. H., Buhrgard, M., Harmatos, J., Mohalik, S. K., Roe-land, D., & Niemoller, J. (2021). Intent-driven closed loops for autonomous networks. Journal of ICT Standardization, 2(2), 257-290.

[7] Rakhra, M., Gopinadh, G., Addepalli, N., Singh, G., Aliraja, S., Reddy, V., & Reddy, M. (2021). E-Commerce Assistance with a Smart Chatbot using Artificial Intelligence. In Proceedings of the International Conference on Industrial Engineering and Management (IC-IEM), 144-148. DOI: 10.1109/ICIEM51511.2021.9445316.

[8] Lam, K. N., Le, N. N., & Kalita, J. (2021). Building a Chatbot on a Closed Domain using RASA. In Proceedings of the 4th International Conference on Natural Language Processing and Information Retrieval (NLPIR 2020), 144-148. ACM. DOI: 10.1145/3443279.3443308.

[9] Abbas, K.; Afaq, M.; Ahmed Khan, T.; Rafiq, A.; Song, W.-C. Slicing the Core Network and Radio Access Network Domains through Intent-Based Networking for 5G Networks. Electronics 2020, 9, 1710. https://doi.org/10.3390/electronics9101710

[10] Gharbaoui, M., Martini, B., & Castoldi, P. (2021). Implementation of an Intent Layer for SDN-enabled and QoS-Aware Network Slicing. In 2021 IEEE 7th International Conference on Network Softwarization (NetSoft) (pp. 17-23). IEEE. doi: 10.1109/NetSoft51509.2021.9492643.

[11] Bezahaf, M., Davies, E., Rotsos, C., & Race, N. (2021). To All Intents and Purposes: Towards Flexible Intent Expression. In 2021 IEEE 7th International Conference on Network Softwarization (NetSoft) (pp. 31-37). IEEE. doi: 10.1109/NetSoft51509.2021.9492554.

[12] Jacobs, A. S., Pfitscher, R. J., Ferreira, R. A., & Granville, L. Z. (2018, August). Refining network intents for self-driving networks. In Proceedings of the Afternoon Workshop on Self-Driving Networks (pp. 15-21).

[13] Chaudhari, A., Asthana, A., Kaluskar, A., Gedia, D., Karani, L., Perigo, L., Gandotra, R., & Gangwar, S. (2019). VIVoNet: Visually-Represented, Intent-Based, Voice-Assisted Networking. International Journal of Computer Networks and Communications, 11, 1-13. DOI: 10.5121/ijcnc.2019.11201.

[14] K. S. Mayer et al., "Demonstration of ML-Assisted Soft-Failure Localization Based on Network Digital Twins," in Journal of Lightwave Technology, vol. 40, no. 14, pp. 4514-4520, 15 July15, 2022, doi: 10.1109/JLT.2022.3170278.

[15] Pinto, R., Cesila, C., Mayer, K., Escallón-Portilla, A., Arantes, D., Mello, D., & Rothenberg, C. (2023). Demonstration of Packet-Optical Intent-Based Survivability Using Mininet-Optical. In Optical Networking and Communication Conference & Exhibition (OFC'23 Demo Zone), March 2023.

[16] Esposito, F., Wang, J., Contoli, C., Davoli, G., Cerroni, W., & Callegati, F. (2018). A Behavior-Driven Approach to Intent Specification for Software-Defined Infrastructure Management. In 2018 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN) (pp. 1-6). doi: 10.1109/NFV-SDN.2018.8725754.