IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 5 March 2025, accepted 6 April 2025, date of publication 11 April 2025, date of current version 24 April 2025.

Digital Object Identifier 10.1109/ACCESS.2025.3560210

RESEARCH ARTICLE

SDN Controller for Optical Network Control

SYLWESTER KACZMAREK[1], MAGDALENA MŁYNARCZUK[1], MACIEJ SAC[1], AND PATRYK MIKLASZEWSKI[2]

[1]Department of Teleinformation Networks, Faculty of Electronics, Telecommunications and Informatics, Gdańsk University of Technology, 80-233 Gdańsk, Poland
[2]Adtran Networks Ltd., 81-537 Gdynia, Poland

Corresponding author: Sylwester Kaczmarek (kasyl@eti.pg.edu.pl)

This work was supported by Gdańsk University of Technology through the Argentum Triggering Research Grants–Excellence Initiative–Research University Program under Grant DEC-25/1/2023/IDUP/I3b/Ag.

ABSTRACT The ability of centralized programming flow tables in switches and routers to perform programmable resource control and management from the central controller under the Software Defined Networking (SDN) concept has introduced a new approach to controlling and managing network resources. The main principles of the SDN concept, as defined by the Open Networking Foundation (ONF), apply primarily to packet networks, making the usage of SDN for optical networks an area of ongoing work. However, ITU-T standardization recommendations have described the reference SDN controller architecture applicable to optical transport networks in terms of abstract components provided by the Automatically Switched Optical Network (ASON) approach. The paper regards the problem of an SDN controller for optical network control according to the G.7702 ITU-T recommendation approved in 2022. The proposition of the SDN controller presented in the paper is compliant with the guidelines outlined in ITU-T recommendations, while ensuring compatibility with the standards established for OpenFlow. The integration of the SDN concept with optical network architecture is performed based on an ASON/GMPLS optical network. In the presented solution, the SDN controller is built with ASON controllers, and dedicated OpenFlow messages are proposed based on ITU-T and ONF guidelines. The proposed solution is evaluated with the simulation model in the OMNeT++ environment. The performance of the proposed solution under specific traffic conditions and different SDN controller placements is presented.

INDEX TERMS ASON, GMPLS, OpenFlow, optical networks, performance, quality of service, SDN controller.

I. INTRODUCTION

The development of information technology and telecommunication services (multimedia, Internet of Things (IoT)) has led to increasing demands for the capacity of telecommunications networks. It has become particularly significant due to the COVID-19 pandemic, when we were dealing with a huge amount of new and dynamically changing teletraffic. Currently, providers are compelled to allocate resources more rapidly, ensuring the appropriate Quality of Service standards while optimizing the utilization of network resources.

Dedicated network architectures are provided to create a unified control plane for the transport network in the core network. One of the propositions is the ASON/GMPLS architecture, which means Automatically Switched Optical Network (ASON) architecture [1], which is built in the control plane on the concept of Generalized Multi-Protocol Label Switching (GMPLS) protocols [2].

Meanwhile, in order to meet the expectations of telecommunications operators and enable the automation of resource management and traffic control, regardless of the solutions used in the transport layer, the concept of Software Defined Networking (SDN) in a unified control plane has been proposed [3]. Over the past decade, providers and researchers have made remarkable advances in SDN packet networks solutions. The SDN approach has been successfully implemented in both wired and wireless networks [4]. However, the application of SDN in optical networks is still the subject of research and requires diligent study.

The associate editor coordinating the review of this manuscript and approving it for publication was Dominik Strzalka.

The general rules of cooperation between the ASON/GMPLS and the SDN have already been standardized in [3] and [5]. The recommendation [5] outlines the concepts and management control components that are common to the use of both the SDN and the ASON in order to control the resources of the transport network. Actually, the references and ITU-T recommendations are subject to revision, so the authors have made an effort to investigate the possibility of applying the recent editions of [3] and to examine SDN based on the ASON/GMPLS control plane approach.

The integration of ASON/GMPLS and the SDN concept results in quality of the SDN network, which is affected by delays introduced by the controller. While such delays may not have a significant impact on handling low-priority traffic, they become crucial for guaranteeing quality for high-priority traffic.

Summarizing, the SDN approach proposed in this work offers the following contributions:

- a new proposition of an SDN controller, according to the ITU-T specifications outlined in [3] and [5],
- extensions of messages in order to control optical ASON/GMPLS networks with adherence to the OpenFlow protocol guidelines [6],
- comprehensive evaluation of the network performance with the proposed SDN controller under different teletraffic conditions and service classes (low- and high-priority) in the OMNeT++ environment [7].

The results of our work on integrating the SDN and ASON/GMPLS concepts are described in this paper, which is organized as follows. Section II contains the presentation of the SDN architecture and a review of related works concerning SDN for optical networks. Section III describes the concept of the SDN controller built with the ASON/GMPLS network controllers according to the ITU-T [3] recommendation. Section IV provides details of the structure and implementation of the simulation model of the SDN/ASON/GMPLS network in the OMNeT++ environment. Section V contains the results of the simulations, including a discussion of issues related to quality of service. The paper is summarized in Section VI, in which planned further work is also presented.

II. SDN ARCHITECTURE AND RELATED WORKS

The general principles of the SDN concept are defined by the Open Networking Foundation (ONF) [8], an organization dedicated to the promotion and adoption of SDN. The general concept of the SDN network based on ONF standards, including packet and optical networks, is presented in Fig. 1.

In the architecture of this network, we distinguish: data plane, controller plane, management plane, and application plane.

The data plane consists of a set of transport network packet devices and optical devices, which directly handle customer traffic, i.e. transport and processing data under the control of the controller plane, and communicating with the controller

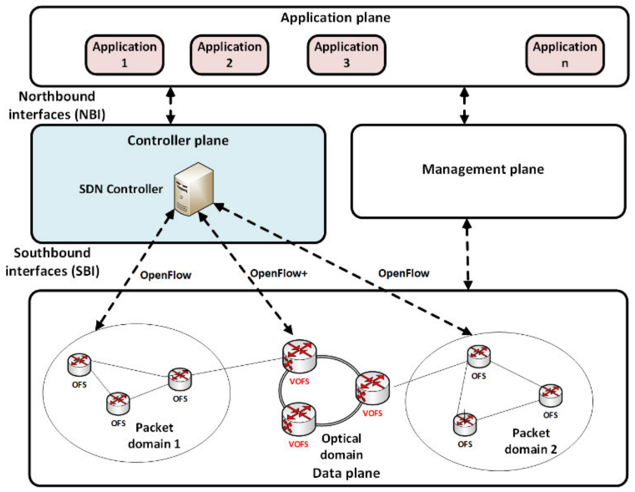

FIGURE 1. SDN network architecture.

plane using southbound interfaces (SBI). This means that the devices themselves only execute commands from the SDN controller, because the control, i.e. decision-making regarding packet handling, remains separate and is moved to the controller plane. The data plane is also responsible for the implementing the maintenance tasks required by the management plane.

The controller plane provides tools for controlling data plane resources. The most important component of the controller plane is the SDN controller, which is separated from the data plane to ensure control of data plane resources and communication with the application plane. It is also important to note that there may be several SDN controllers in multidomain networks.

The application plane contains SDN applications. These applications communicate with the controller plane through standardized interfaces, utilizing an abstract view of network resources and data models. This enables the applications to customize and automate operations on network resources.

Communication between the data plane and the controller plane is provided via SBI, while communication between the application plane and the controller plane is performed with the use of northbound interfaces (NBI). The interface between the data and controller plane is mainly provided by the OpenFlow protocol, so in the typical packet network, the OpenFlow Switch (OFS) is equipped with the flow tables and software dedicated to communication with the SDN controller. On the other hand, for transport network solutions using OXC, OADM, ROADM, the nodes of this network have their own control planes with installed software for communication with the Virtual OpenFlow Switch (VOFS) SDN controller. For optical networks, the OpenFlow extensions (depicted in Fig. 1 as OpenFlow+) are necessary to configure optical resources. This approach entails considerable architectural complexity.

The first works concerned with SDN for optical networks were focused on Packet and Circuit Convergence (PAC.C) extensions to OpenFlow, where the optical network component of the data plane was extended to support circuit flows by adding the cross-connect tables to the packet circuit flow tables [9]. The circuit flows in the PAC.C approach are defined by fields for each input and output optical port as follows: port number, wavelength, virtual port associated with the Virtual Concatenation Group (VCG), and starting time slot of the SONET/SDH, enabling packet convergence.

The presented solution provides the interconnection between the packet and circuit domains [9]. However, the practical implementation of this solution turned out challenging due to the limited support for the extended version of the OpenFlow protocol in the optical devices.

Moreover, the rapid development of the GMPLS architecture in the core network, proposed by the IETF, encouraged researchers to examine the SDN architecture based on GMPLS solutions.

The idea of integrating the GMPLS functionality with the SDN concept has been developed under the European project OFELIA [9], where two approaches have been proposed for controlling optical networks based on the GMPLS architecture. Both solutions are based on the concept of the OpenFlow Agent (OF Agent): Hybrid GMPLS-OpenFlow and Pure Extended OpenFlow [9]. These approaches are presented in Fig. 2 and Fig. 3, respectively.

In the hybrid OpenFlow model, the controller plane is divided into two components: Extended OpenFlow controller (extended NOX [10] controller) and GMPLS Control Library with OpenFlow Path Computation Entity (OF PCE) [11], [12].

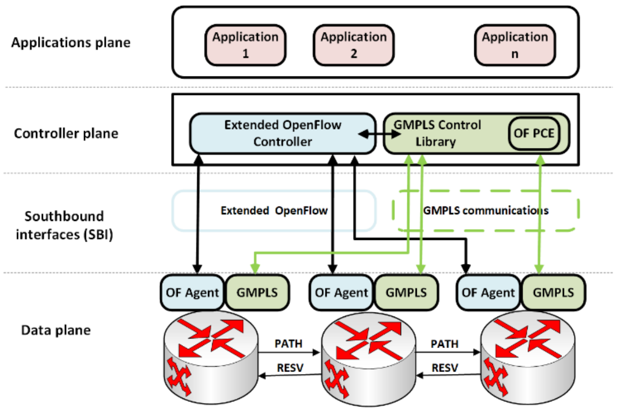

FIGURE 2. Hybrid GMPLS-OpenFlow approach.

The controller plane has separated tasks defined for the Extended OpenFlow controller using the OpenFlow protocol and those performed using the GMPLS control plane.

The Extended OpenFlow controller communicates with the optical network component via the OpenFlow Agent and is responsible for providing resource and topology information for the applications plane. The GMPLS control library and OF PCE are used for lightpath computation, establishment, and modification. Within this approach, there are two methods of calculating the lightpath: loose and explicit.

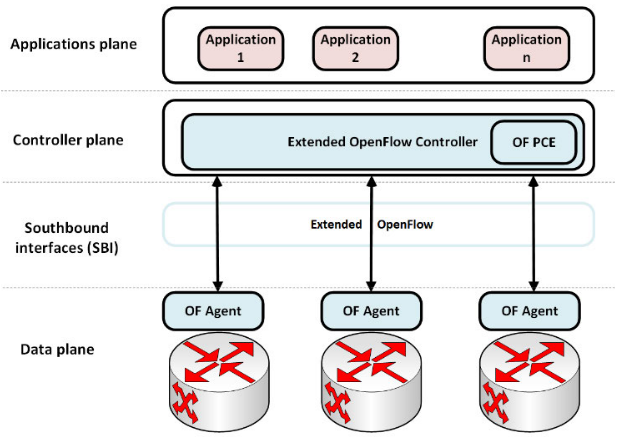

FIGURE 3. Pure extended openFlow approach.

In the first method, the Extended OpenFlow Controller determines the edge nodes and ports, while the GMPLS control plane calculates and establishes the Label switching Path (LSP) within domains using RSVP-TE signaling. In the second approach, the Extended OpenFlow Controller uses topology and resource information to compute paths. The LSP is explicitly defined and established using RSVP-TE signaling.

In the Pure Extended-OpenFlow model, the OF PCE is nested in the Extended OpenFlow Controller [12]. The OF PCE component is responsible for calculating the paths with proper consideration for switching constraints, based on the topology and resource information provided by the network devices via the OpenFlow Agent. The communication between the Extended OpenFlow Controller and the network device is performed by the OpenFlow Agent over an OpenFlow channel.

A real experimental testbed of an OpenFlow-based control plane for transparent optical networks, based on the Pure Extended-OpenFlow model, was presented in the ML-MG project [13], where switches (VOFS) communicated with the OpenFlow controller using the OF channel, while sending standard Transaction Language 1 (TL1) to configure the optical device.

New trends have led to the development of the Path Computation Entity (PCE) [14]. Much work on the potential of PCE has been carried out to expand its functionality and guarantee quality of service in optical networks. Experiments on implementing SDN with extended PCE and a replica management system for flexi-grid networks are presented in [15]. Interworking GMPLS control and centralized controller systems for use in the SDN network architecture are a subject of ongoing work of the IETF [16].

Summing up, most SDN concept studies have tended to focus on the GMPLS/PCE application with OpenFlow extensions, rather than ASON/GMPLS. The presented testbeds implemented Floodlight, NOX controller solutions, which were originally dedicated to packet networks, and their functionality was extended to support optical network control. The articles lack information on the details of the controller architecture. Because current optical node solutions do not support the OpenFlow protocol, the proposed solutions implemented an OpenFlow agent that communicates with optical nodes using existing solutions dedicated to management (e.g. via SNMP, vendor-specific APIs).

The ITU-T standard [3] was published in 2022. It proposes the use of ASON control plane controllers to implement the SDN controller functionality. Through this, the SDN controller can act as a master component implementing resource control functions in this network. The ITU-T organization proposes not only a comprehensive description of the controller architecture in [3], but also interfaces for control and management of the optical network in [1]. The standard [3] clarifies the SDN controller architecture and describes the controller functionality in terms of support for optical networks, but it does not provide detailed scenarios for controlling an optical network.

The ITU-T standardization community has also raised some issues about ASON and SDN integration in [3] and [5]. Despite the interest in ASON, the proposition named in this paper as SDN/ASON/GMPLS has been overlooked. In the literature, there has been little discussion and practical implementation of the SDN/ASON/GMPLS concept, prompting us to undertake this study.

We applied an SDN controller according to the recommendation [3]. We specified the requirements of [3] regarding communication between ASON components via defined interfaces in SDN/ASON/GMPLS networks. We added extensions of messages to control optical ASON/GMPLS networks and to integrate the ASON/GMPLS and SDN approaches. We proposed call service scenarios which are not defined for SDN/ASON/GMPLS networks. The functional correctness and performance of the proposed solution of the SDN/ASON/GMPLS network we evaluated under specific traffic conditions and different SDN controller placements using the OMNeT++ simulation environment. The standardization of the SDN controller according to [3] and the concept of the SDN/ASON/GMPLS architecture are presented in Section III.

III. CONCEPT OF SDN/ASON/GMPLS ARCHITECTURE

The ITU-T standard containing the recommendation [3] proposes a reference architecture for SDN control of transport networks applicable to optical transport networks in terms of abstract components provided by the ASON approach.

A. SDN CONTROLLER STANDARDIZATION

The application of the SDN concept in the ASON optical network is based on combining the existing control components of the control plane in the form of ASON call/connection controllers into one control unit named the SDN controller, thus ensuring centralized control of network resources. This single controller includes the control components of the ASON architecture, such as the Network Call Controller (NCC), Directory Service (DS), Connection Controller (CC), Routing Controller (RC), Termination and Adaptation Performer (TAP), and Link Resource Manager (LRM).

The role of the NCC is to process call requests from the clients. Due to the separate namespaces for the call and connection services, the NCC queries the DS to translate the call source and destination identifiers into control name identifiers and sends a connection request to the Connection Controller. The Connection Controller, based on a route query to the RC and resource state information from the Link Resource Manager, calculates the route and establishes resources via the link connection request in the LRM. The LRM interacts with the TAP to configure optical resources. The interactions between the control components in the SDN controller are illustrated in Fig. 4.

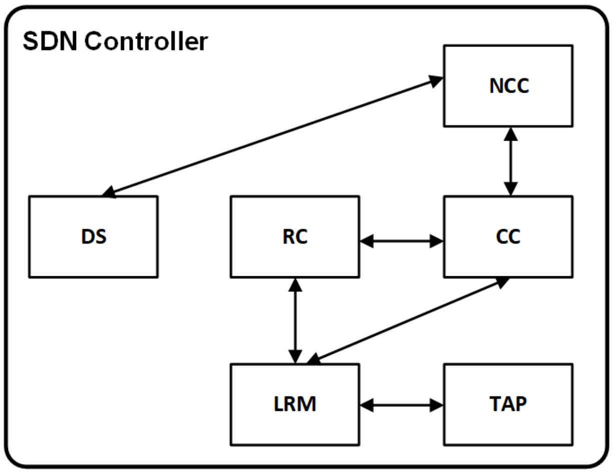

FIGURE 4. SDN controller based on ASON/GMPLS components.

The authors have experience in ASON/GMPLS and SDN research [17], [18], [19] and decided to implement the SDN concept in the ASON/GMPLS optical network. They rebuilt and extended the ASON architecture simulation model presented in [17], which provided GMPLS protocols and mechanisms, and they propose the SDN/ASON/GMPLS simulator.

B. SDN/ASON/GMPLS ARCHITECTURE

The proposed SDN/ASON/GMPLS network architecture is presented in Fig. 5. For clarity and to explain the concept, the SDN/ASON/GMPLS architecture is limited to two domains (Domain A and Domain B), each containing two optical nodes. The description of the SDN/ASON/GMPLS architecture also applies to a larger number of domains for different domain structures.

The SDN controller complies with the requirements of the ASON network and the ITU-T recommendation. The SDN controller contains the following controllers: NCC, CC, LRM, TAP, DS and RC.

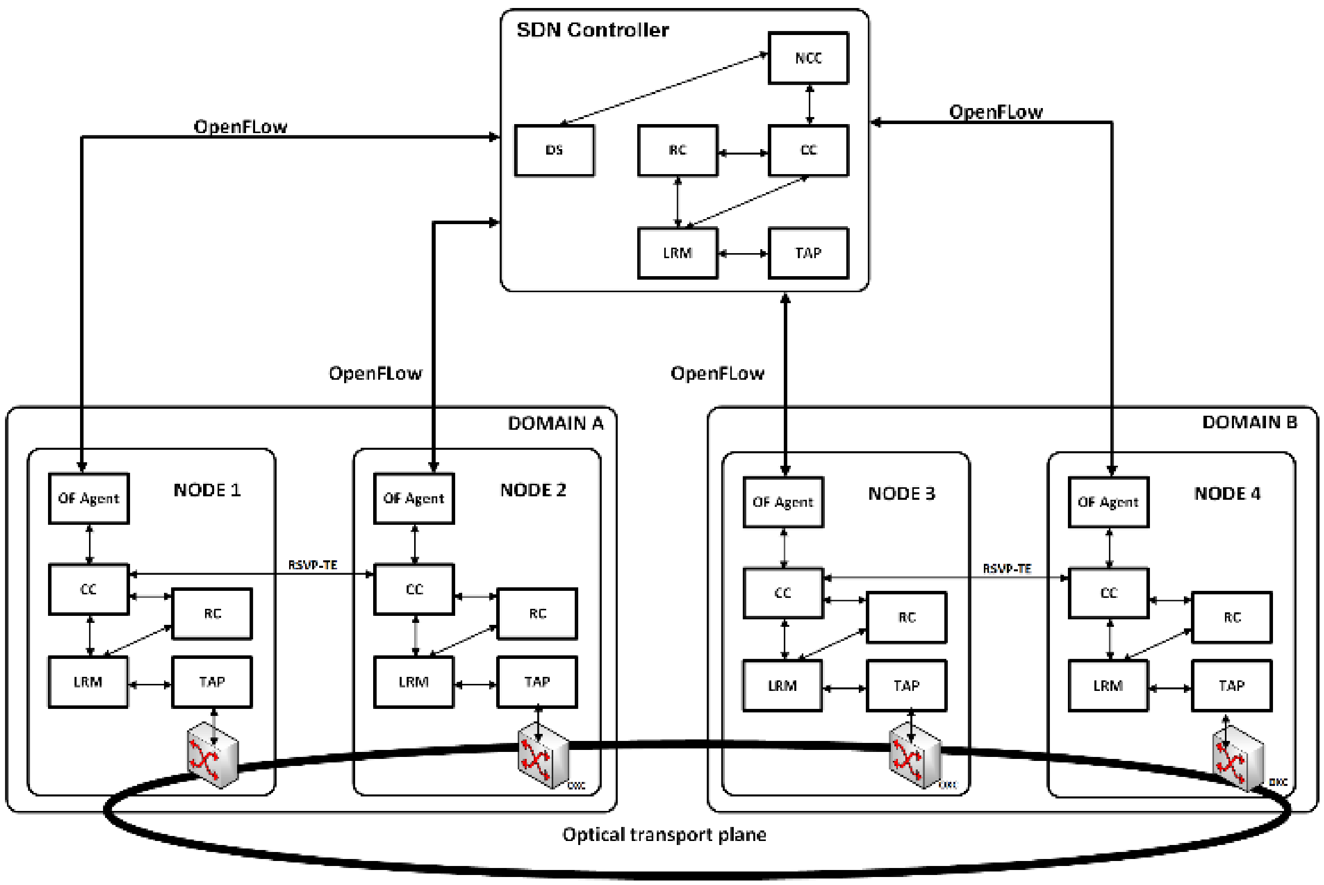

FIGURE 5. SDN/ASON/GMPLS network architecture.

The RC, LRM, and TAP controllers in the SDN controller are globally supported and related to cross-domain access. The NCC controller, based on identifier translation, transforms the call request to the connection request and transfers it to the CC. The CC sends a route query to the RC and, based on the topology and local connection status information provided by the LRM, reserves resources on an interdomain link.

Each node in the transport plane consists of several functional components: OF Agent, CC, RC, LRM, and TAP. In this case, the RC, LRM, and TAP components are within the scope of the domain, and LRM and TAP determine the available resources in the optical node.

The SDN controller has a view of the resources between domains, i.e. at their interfaces. Communication between the SDN controller and OF agents in optical nodes is achieved through the OpenFlow protocol extended by OpenFlow messages providing support according to [6].

The connection setup process begins with a request from the client to the SDN controller. The NCC controller processes the request and, based on the included information, takes specific actions. In the proposed network architecture, there are two types of connections, within one domain and between domains, for which the operating scheme is slightly different.

For a connection within one domain, the SDN controller processing the call request forwards the request to the OF Agent of the source node using the Open-Flow protocol extended by OpenFlow messages. The OF Agent communicates with the CC controller in the node. The CC controller in the node, in cooperation with the LRM and RC, reserves resources along the path. The path is therefore reserved within the domain with the use of CC coordination and the RSVP-TE protocol, without the participation of the SDN controller.

When the SDN controller receives a call request from the client to establish a connection between nodes located in different domains, it identifies the connection as multidomain and queries the resources and path to its LRM and RC controllers, which have a view of the resources located between domains. Once a decision is made, the CC in the SDN Controller sends a request to the source node, including details about the specific inter-domain optical link and the wavelength at the domain edge node. At this stage, the connection can be established in two ways.

The first one refers to establishing a hop-by-hop connection, i.e. the request is sent to the source node of one domain within which the connection is established, as in the case of a single-domain connection. Once a connection within one domain is established, the connection in the other domain is established, also using RSVP-TE. The second way determines establishing simultaneous connections in both domains. The request is directed to the source node of one domain and the edge node of the other domain, in which the destination node is located. Connections within the single domains are simultaneously established to the inter-domain edge nodes using the RSVP-TE protocol.

The details of the SDN/ASON/GMPLS architecture implementation are presented in Section IV.

IV. STRUCTURE AND IMPLEMENTATION OF SDN/ASON/GMPLS NETWORK SIMULATION MODEL

The SDN simulator was implemented in the OMNeT++ environment, which is an object-oriented modular discrete event network simulation framework developed by András Varga [7]. The simulator is equipped with functional blocks specified for the SDN concept (omnetpp.ini,.ned files, *cc files, *msg files) and the OMNeT++ environment components (simulation kernel and class library, model component library, envir-based libraries).

A. STRUCTURE OF SIMULATION MODEL

The functional components of the SDN controller in the SDN/ASON/GMPLS network, control plane components of nodes, and the optical transport plane were implemented in C files. The structure of the network is defined in the NEtwork Description (NED) language in a.ned file. The assumed simulation parameters were set in the omnetpp.ini file. All messages were defined as.msg files.

The simulated network structure is shown in Fig. 6. The proposed implementation is open to challenges related to network scalability.

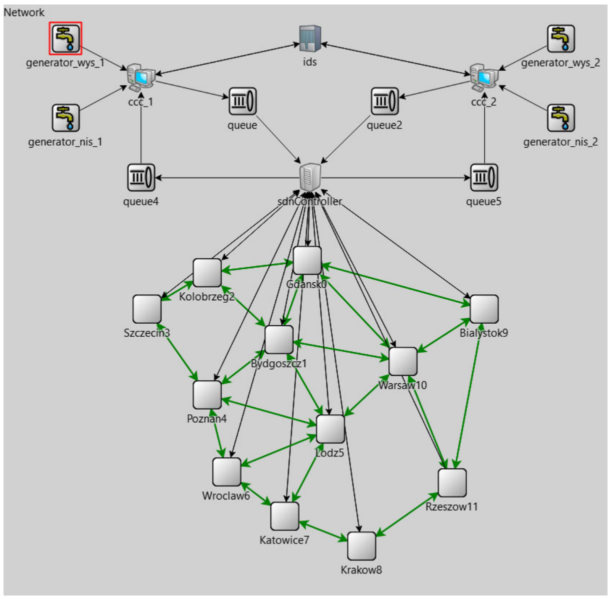

FIGURE 6. SDN/ASON/GMPLS simulated network structure.

The number of domains, nodes and inter-network connections can be modified, as these parameters are configurable in configuration files. Similarly to the model presented in [17], it can be used to study various structures of real networks based on SNDlib [20] and other network structures defined in.ned files, which proves its practical usefulness in designing telecommunication networks with SDN integration.

The central component of the network is the SDN controller, which is located in the Warsaw node and has separate connections to each node. The CCC components (ccc_1 and ccc_2) are Calling/Called Party Call Controllers that are located on the client side.

Components marked as generators (generator_wys_1, generator_wys_2, generator_nis_1, generator_nis_2) are responsible for generating high- or low-priority service requests from the client. An additional ids component is responsible for assigning identifiers to call requests. The model assumes that each link introduces a delay related to the propagation of signals through the optical fiber, which has been set to 5 [μs/km]. To calculate the propagation time through the optical fiber on specific connections between nodes, the actual distances measured between cities are assumed.

In the proposed solution, the SDN controller is equipped with ASON control components. The CC, NCC, and LRM controllers are explicitly implemented in the SDN controller, while the functionality of the RC is implemented in the CC, the TAP is provided in the LRM, and the DS is supported in the NCC.

The authors propose adding an additional component to the SDN controller, called the PC (Protocol Controller), to map connection request messages to extended OpenFlow messages. The structure of the SDN controller and the Gdansk0 optical node are presented in Fig. 7 and Fig. 8, respectively.

In the presented model, the NCC controller in the SDN controller is responsible for communication between the CCC controllers located on the client side. It also maintains a database of nodes and determines whether the connection is single or multidomain. The NCC communicates with the CC and LRM components located in the SDN controller for multidomain connections, and forwards the connection request directly to the source node in the domain for single domain connections. The PC controller located in the SDN controller has a direct connection to the OFA agents located in each of the optical nodes. The OpenFlow agent maps the OpenFlow messages to the control messages used in the ASON/GMPLS network control plane.

Each node, such as the SDN controller, contains CC and LRM controllers. The controllers in the node fulfill the same tasks as the SDN controller, but they operate on resources within the node. In addition to being responsible for communication between components within the node, the CC controller in the node is also responsible for communicating with the CC controllers located in the same domain, in order to reserve resources between nodes via the RSVP-TE protocol.

For the purposes of communication between the SDN controller and the OFA agent, new OpenFlow protocol messages have been introduced, which have a structure consistent

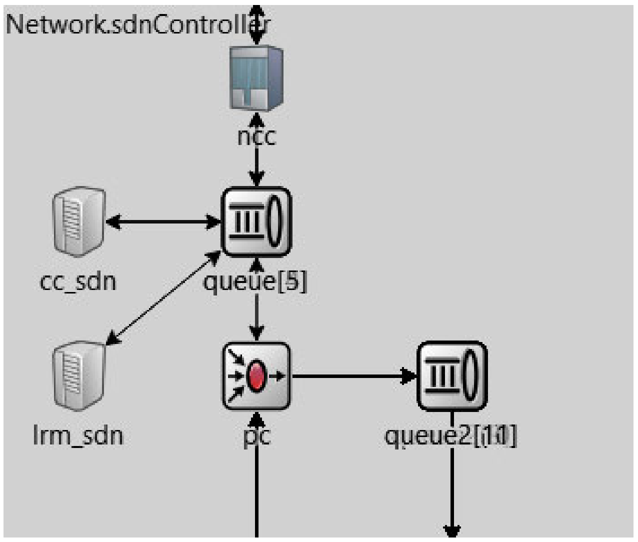

FIGURE 7. Structure of SDN controller implemented in the simulation model.

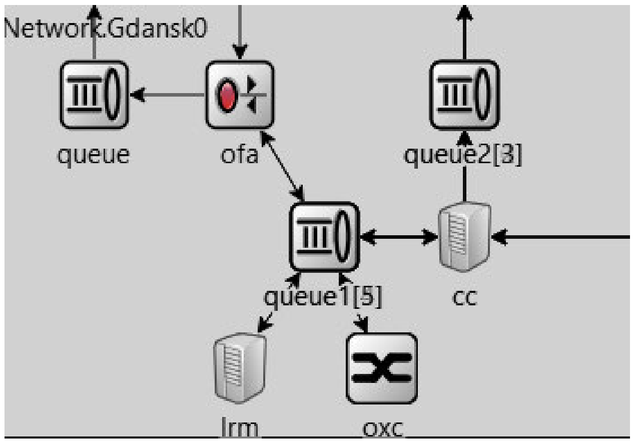

FIGURE 8. Structure of node Gdansk0.

with the protocol and contain a typical Open Flow protocol header [6], which is 32 bits in size. The authors of the paper adapted the data carried by the messages to meet the needs of controller-node communication. The proposed OpenFlow messages are presented in Table 1.

The process of building and running simulations in the OMNeT++ was provided with the use of simulation environment. The description of the process is beyond the scope of the paper and is available in the OMNeT++ documentation [7].

After the execution of the SDN/ASON/GMPLS simulation, the simulation model results were recorded in separate.sca,.vec and.vci files and analyzed using dedicated scripts written in the Python language.

B. FLOW OF MESSAGES IN SDN/ASON/GMPLS NETWORK

In this section, the authors present flows of messages in the SDN/ASON/GMPLS architecture for the structure depicted in Fig. 6, in which the Gdansk and Bydgoszcz nodes belong

TABLE 1. Proposed openflow messages.

Message name	Size [bits]	Description
OF_REQ_37	192	Connection setup request, message sent from PC to OFA to initialize connection setup within the domain.
OF_CONF_38	96	Confirmation of connection setup, message sent from OFA to PC to confirm connection setup.
OF_REL_39	96	Connection release request, message sent from PC to OFA to release connection.
OF_ERROR_40	112	Error, message sent from OFA to PC to inform about reservation failure due to resource unavailability.
OF_REL_CONF_41	96	Confirmation of connection release, message sent from OFA to PC to confirm connection release.
OF_REQ_BORDER_42	576	Connection setup request for the border node, message sent from PC to OFA to establish connection between border nodes.
OF_CONF_BORDER_43	96	Confirmation of connection establishment from border node, message sent from OFA to PC to confirm connection establishment between border nodes.
OF_REL_BORDER_44	96	Request to release connection in border node, message sent from PC to OFA to release connection between border nodes.
OF_REL_CONF_BORDE R_45	96	Confirmation of connection release from border node, message sent from OFA to PC to confirm connection release between border nodes.

to domain A, Lodz and Katowice belong to domain B, and the SDN controller is located in Domain C in the Warsaw node.

The single-domain connection setup process is implemented as follows (Fig. 9). The client-side controller *CCC_1* initiates a call request to the *CCC_2* controller by sending a *Call_Request* message to the SDN controller. Then, the SDN controller informs the *CCC_2* controller about the accepted call request with a *Call_Indication* message. *CCC_2* confirms the acceptance of the request by sending a *Call_Confirmed* message to the SDN controller, which further directs the *OF_REQ_37* connection setup request directly to the source node, and more precisely, to the OFA agent located in this node. This starts the process of establishing a connection in Domian 1 with the use of RSVP-TE messages. The PATH message is sent toward the destination node to reserve optical resources, appropriate optical resources are reserved in the LRM module and then a RESV message is sent to confirm this reservation toward the source node. The CC sends a *lrm_req_of* query to the LRM asking for the information required to add an entry to the flow table, which is a copy of the flow table in the OXC. The LRM returns this information in an *lrm_response_of* message, then the CC sends a *set_of_entry* message to the OFA agent that maintains these tables. Sending a set of entry messages invokes installation of rules in the OXC. The mapping flow tables in the LRM optimize the resource reservation process. In the scenario of a reservation failure, there is no need to remove the entry from the flow table in the OXC.

The connection is successfully established in the connection control plane in Domian 1 when the source node responds with an *OF_CONF_38* confirmation. After this message, the PC controller sends *Connection_Confirmed* to the NCC, and a *Call_Confirmed* confirmation message is sent to the client CCC controller from which the initial request came, thereby ending the call setup request. The cooperation between the controllers in the source and destination nodes is presented in Fig. 9.

The single-domain call release scenario is presented in Fig. 10, and it is similarly described as the setup scenario.

The client-side controller CCC_1 initiates a call release to the CCC_2 controller by sending a *Call_Release* message to the SDN controller. Then, the SDN controller informs the *CCC_2* controller about the accepted call release with a *Release_Indication* message. The *CCC_2* confirms the acceptance of the request by sending a *Release_Confirmed* message to the SDN controller, which further directs the *OF_REL_39* connection release request directly to the OFA agent located in the source node. The OFA transforms the message into a *connection_release* message, which is sent to the CC of the source node. As a result, the CC controller initiates the resource release process in the LRM by sending a *lrm_release* and sends a release request to the optical switch OXC. The process of releasing a connection in Domian 1 is performed with the use of RSVP-TE messages. At the beginning, the *PATH release* message toward the destination node is sent and the appropriate optical resources are released in the LRM modules. Then, a *RESV release* message is sent to confirm this release toward the source node. Once the resources have been released along the path, and the rules in the flow tables have been deleted, the source node sends *OF_REL_CONF_41*to the SDN controller. After this message, the PC controller sends *Connection_Release_Confirmed* to the NCC, and a *Release_Confirmed* confirmation message is sent to the client CCC controller from which the initial release request came, thereby ending the call release request.

The multidomain connection setup scenario is presented in Fig. 11. When the SDN controller receives a call request to establish a connection, for example, from the Gdansk node, which is in Domain A, to the Katowice node in Domian B, the call request is classified as multidomain. The SDN controller must allocate cross-domain resources for this connection because it maintains resource information on the cross-domain links between Bydgoszcz and Lodz. Therefore, the NCC controller located in the SDN controller sends a *Connection_Request_SDN* message to the CC_SDN controller to reserve an interdomain link and allocate resources for the connection.

The CC_SDN controller first queries the RC_SDN controller to designate the interdomain link. Interdomain links are precomputed during the initialization process of the simulation based on Dijkstra's algorithm. The code of the RC_SDN is contained within the CC_SDN, so no messages are transmitted. After receiving the response, the CC_SDN, forwards an *LRM_Path_SDN* query for free resources to the LRM_SDN controller. The LRM_SDN controller reserves the resources and sends a response to CC_SDN.

Then CC_SDN, having all the information needed to set up the connection, forwards it to the NCC controller using a *Connection_Response_SDN* message. The NCC makes four connection requests. The first *Connection_Request* is sent to the PC controller, which translates this message into an *OF_REQ_37* OpenFlow protocol message. The request to establish a connection between the source node and the edge located in the source domain (Domain A) is handled by the same algorithm as applied in the single domain connection presented in Fig. 9.

The second and third connection requests are *Connection_Request_Border* messages, which are sent to the PC, where they are translated into *OF_REQ_BORDER_42* messages and then sent to the edge nodes (to OFA agents placed in the nodes) in order to establish a connection between domains using CC, RC, LRM cooperation.

The last *Connection_Request* is sent to set up the connection within Domain B. When the optical resource reservations in the nodes are successfully completed, each of the nodes to which the requests were transferred sends either a *Connection_Confirmed* message or a *Connection_Confirmed_Border* message to the corresponding OFA agent.

Agents send acknowledgments translated into OpenFlow *OF_CONF_38* or *OF_CONF_BORDER_43* protocol messages to the PC controller in the SDN controller. The PC-translated confirmations are then sent directly to the NCC. The NCC controller sends a connection setup confirmation to the CCC from which the connection setup request came. This only takes place when all previously submitted connection setup requests are confirmed.

The multidomain call release scenario is presented in Fig. 12. When the NCC component in the SDN controller receives a *Call_Release* message from CCC_1 to release the connection from the Gdansk node to the Katowice node, it informs the CCC_2 controller about the accepted call release with a *Release_Indication* message. The CCC_2 confirms acceptance of the call release by sending a *Release_Confirmed* message to the NCC, which further directs the *Connection_Release_SDN* message to the CC_SDN.

Then, *LRM_Release_SDN* is transferred to release the resources in the LRM, and the *Release_Response_SDN* message is sent to the NCC.

Once the LRM resources have been changed, the appropriate *Connection_Release and Connection_Release_Border* messages are sent to the PC, whose component processes this message into OpenFlow messages, which are sent to the OFA of the Gdansk, Bydgoszcz, Lodz, and Katowice nodes.

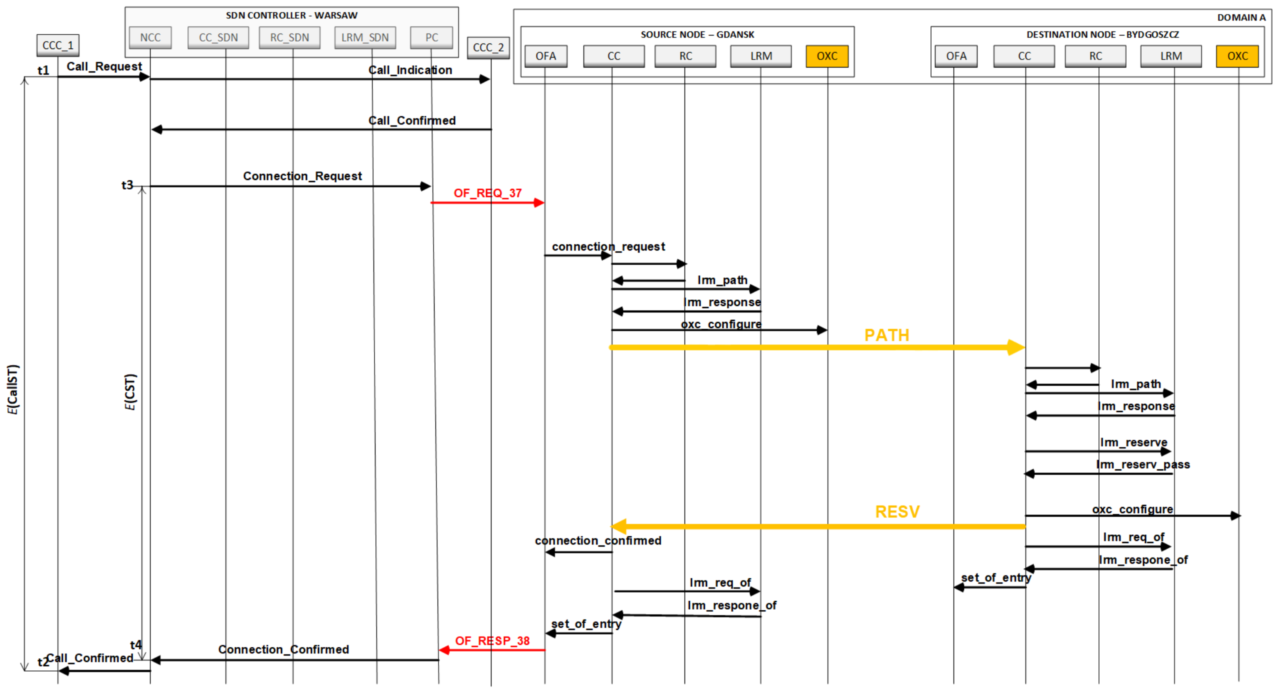

FIGURE 9. Message flow for call setup scenario.

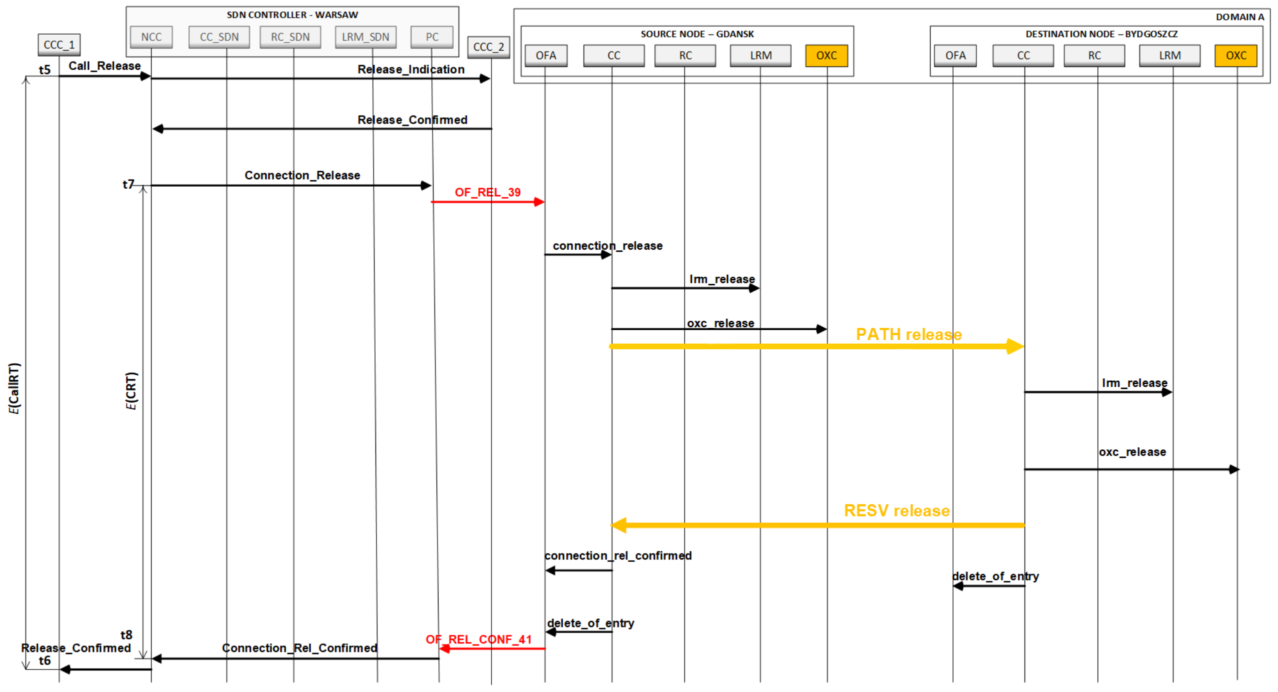

FIGURE 10. Message flow for call release scenario.

The CC in the Gdansk node, having all the information needed to release the connection, then forwards RSVP-TE messages to release the resources within domain A. The CC in the Lodz node, having all the information needed to release the connection, forwards RSVP-TE messages to release the resources within domain B.

The multidomain link resources are released with the use of *OF_REL_BORDER_44* messages. When the multidomain call release scenario is finished, a *Release_Confirmed* message is transferred to *CCC_1*.

The presented call setup and call release scenarios include sequences of operations and message exchanges. However,

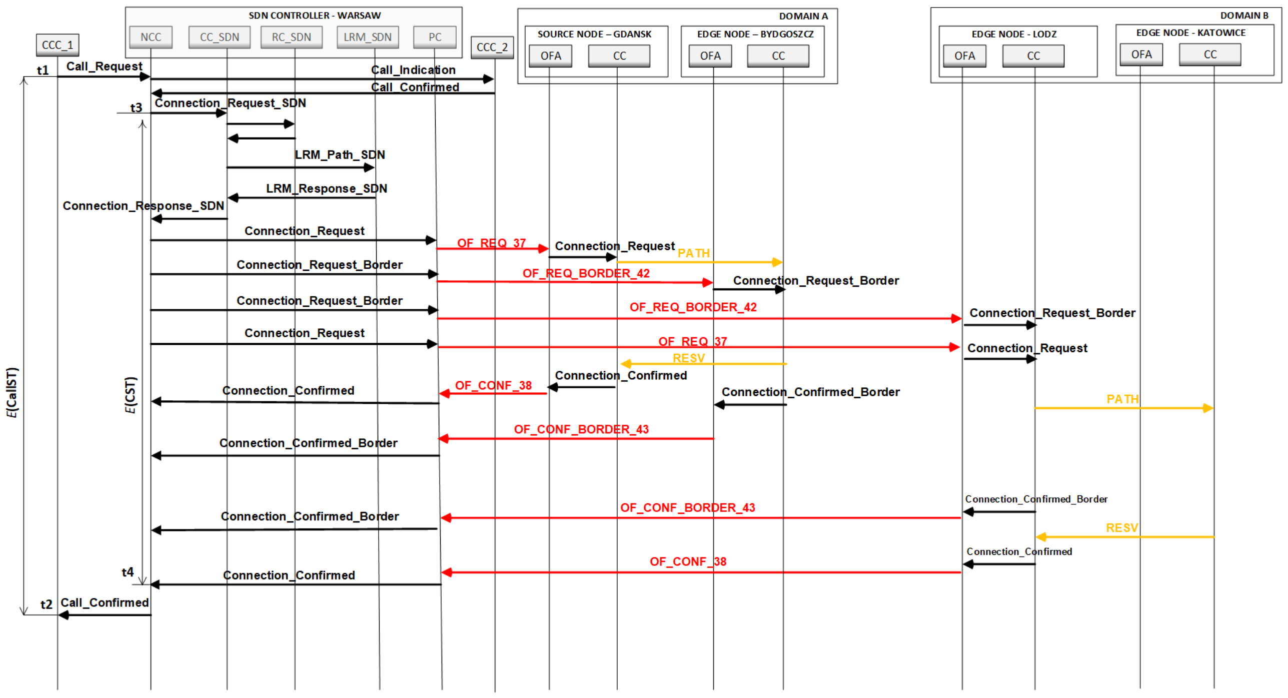

FIGURE 11. Message flow for multidomain call setup scenario.

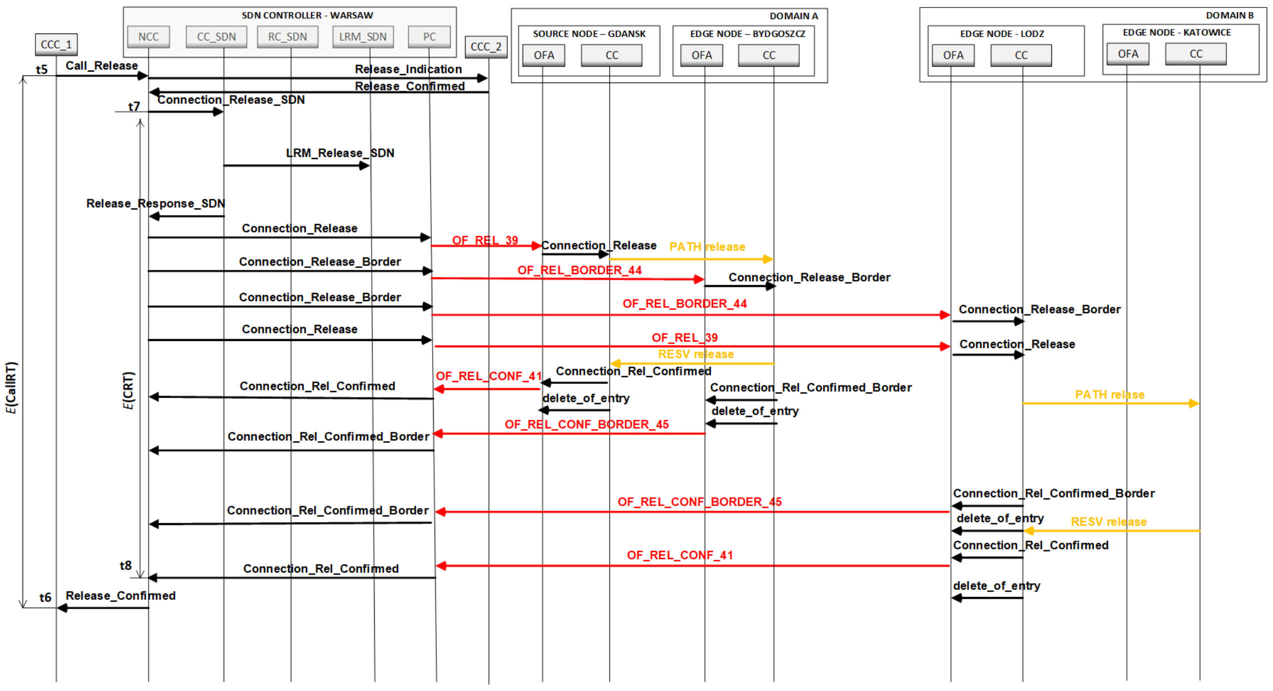

FIGURE 12. Message flow for multidomain call release scenario.

to reflect the actual phenomena occurring when exchanging and processing these messages, a number of different parameters were assumed in the simulation model.

The call service request handling times included the message transmission time, propagation time, and message processing time in individual functional components. When sending a message, the model took into account the size of the transmitted data, the size of the IP protocol header, and the size of the Ethernet header. The size of the signaling message was calculated based on the number of fields carrying information (dependent on the message) and their type.

The SDN/ASON/GMPLS simulation model was tested in two main stages:

- partial tests,
- comprehensive tests.

The partial tests involved verifying the communication between the modules of the simulation model. Functional partial tests for the network of four nodes were performed. The structure of the simplified network was divided into two domains containing two nodes each. The connection setup and release scenarios were first implemented in the simplified network in which they were tested. After achieving the proper functionality of the functions, the simplified network was expanded with additional nodes. The target network was obtained with functionality that meets the specified requirements. The tests included the ability to send messages between components, correctness of the sequence of subsequent events (in accordance with the scenarios), correct code compilation and logical correctness of the programmed functions. A tool enabling graphical simulation and event records (EventLogs) were used to conduct the functional partial tests.

Comprehensive tests for the expanded network took place once the code was fully implemented. In this stage, the following scenarios were tested in the simulator:

- call setup and call release for a single domain connection,
- call setup and call release for a multidomain connection.

The tests were performed for different conditions of optical resources (lack of link optical resources, blocking of OXC). The conducted tests proved that the SDN/ASON/GMPLS simulation model works correctly.

V. SIMULATION RESULTS

To evaluate the effectiveness of the implemented SDN controller solution for controlling the ASON/GMPLS optical network, the appropriate simulation research was performed using the SDN/ASON/GMPLS model presented in Section IV.

The main aim of the research was to examine the proposed SDN/ASON/GMPLS architecture under different traffic conditions, focusing on the impact of the architecture on the quality of service, and the effect of the SDN controller placement on the quality of service. The simulation was carried out for a Polish network based on the Survivable Network Design Library (SNDlib) [20], divided into three domains with the controller appropriately placed in the Lodz, Bydgoszcz and Warsaw nodes. The structure of the multidomain network is presented in Fig. 6.

The assumed parameters and simulation conditions were as follows:

- network simulation time: 2000000 seconds, 30000 seconds, 150000 seconds, 67000 seconds, 60000 seconds, 60000 seconds, 50000 seconds, 33400 seconds and 16300 seconds, corresponding to call request intensities of: 1 request per second, 2 requests per second, 4 requests per second, 8 requests per second, 9 requests per second, 12 requests per second, 16 requests per second, 18 requests per second (differences in times result from the assumed number of events),
- warmup period: 600 seconds,
- uniform traffic distribution for source-destination nodes,
- exponential distribution of call request (two classes: low priority (LP) and high priority (HP)),
- exponential distribution of connection release requests (connection duration time),
- percentage of high-priority requests in all generated traffic: 20%,
- bandwidth units for call request: 5Mb/s, 10Mb/s, 15Mb/s,
- mean connection duration time: 900 seconds,
- signaling link capacity:1Gb/s,
- 40 wavelengths on each link,
- wavelength capacity:1Gb/s,
- OXC blocking probability: 0.001.

In the simulation, the connection route was precomputed using Dijkstra's algorithm while maintaining the wave continuity constraint condition. As a result, the connection was established on the optical path between the source and destination nodes at the same wavelength. There were no wavelength converters in the network. The division of resources on the output port of the optical node took into account priorities, with resource reservation managed by the POOL algorithm [21]. The message processing time was assumed based on measurements performed in small-scale laboratory testbeds in GUT laboratory [22].

The examined parameters were Mean Call Set-up Time $E(\text{CallST})$, Mean Connection Set-up Time $E(\text{CST})$, Mean Call Release Time $E(\text{CallRT})$, Mean Connection Release Time $E(\text{CallRT})$, link loss probability (Bs) resulting from a lack of resources on an optical link, OXC blocking probability (Bb), and total loss probability (Bc) of a call request. Times were measured in accordance with the recommendations in [23].

The time measurement began and ended according to the events specified in the scenarios described in Section III. The probability of loss was calculated based on the number of occurrences of specific events. For each event that occurred during the simulation, information about the priority of the request and the cause of the loss was recorded in the result files (.vec,.vci,.sca). Based on this information, it was possible to calculate both the total number of events and the number of events differentiated by priority.

The loss probability was calculated after the simulation was completed as the ratio of the number of failed call requests with specific parameters (cause of failure, priority) to the total number of requested call requests. As the values are statistical, it was necessary to estimate the accuracy of the obtained values. To achieve this, the interval estimation method and Student's t-distribution were used, and confidence intervals were calculated with a confidence level of 0.95.

All simulations were performed on a DELL server with the following hardware parameters:

- PowerEdge R650xs Motherboard,
- two Intel Xeon Silver 4310 processors, 12 cores (12C) and 24 threads (12T) per processor,
- $2 \times$ 16GB RDIMM,
- $2 \times$ 6TB HDD, $2 \times$ 2TB SSD.

The simulations were performed in the OMNeT++ environment installed on the Linux platform in the Proxmox Virtual Environment [24].

In this part of the section, the simulation results for different SDN controller placements will be first presented, considering both low- and high-priority call requests. Selected results will then be compared to assess quality parameters. For clarity, the results will be shown in separate charts.

There are two cases of calculated average times: call requests completed successfully and all requests (successfully and unsuccessfully established).

The Call Set-up Time (E(CallST)), measured as the time from the *Call_Request*(t1) to the *Call_Confirmed* message (t2), varied with the different SDN controller placements. The E(CallST) results for low- and high-priority call requests completed successfully and different SDN controller placements are presented in Figs. 13 and 14, respectively. For the SDN controller placement in the Lodz node, the (E(CallST)) did not exceed 14.42 milliseconds. The highest value for the Lodz placement occurred at an intensity of 12 requests per second. We found higher values of E(CallST) for Bydgoszcz and Warsaw than for Lodz, as well as for low- and high-priority requests. Figs. 14 and 15 present the E(CST) results, measured as the time from the *Connection_Request*(t3) to the *Connection_Confirmed* message (t4). We observe that the placement of the SDN controller has an impact on E(CallST) and E(CST) for call requests completed successfully. Considering that E(CallST) includes the handling of the connection, the time was longer for all call request intensities. The graphs in Figs. 16 and 17 show that the difference between E(CallST) and E(CST) for all requests, successfully and unsuccessfully established, did not exceed 4.3 milliseconds. For the results presented in Figs. 13–18, positioning the controller in Lodz resulted in the lowest values of E(CallST) and E(CST). The longest E(CallST) and E(CST) were obtained for the Warsaw placement, and for a call request intensity of 12 requests per second. The times did not exceed 14.8 milliseconds and 10.6 milliseconds, respectively.

Figs. 19 and Fig. 20 present the mean values of the Call Release Times for low- and high-priority, measured from sending the *Call_Release*(t5) to the *Release Confirmed* message (t6). For low-priority requests, E(CallRT) was the shortest for the Lodz placement, and did not exceed 8.15 milliseconds, while for high-priority requests, it did not exceed 8.17 milliseconds. Fig. 21 and Fig. 22 present the mean values of the Connection Release Times for low- and high-priority, measured from sending *Connection_Release*(t7) to *Connection_Rel_Confirmed*(t8). Figs. 23–25 present the loss probabilities for the Polish SDN/ASON/GMPLS structure.

Taking into consideration the probabilities presented in Figs. 23–25 for all analyzed placements, we observe that the placement had no influence on the link loss or total loss probabilities for a given class of service, while the OXC blocking probabilities were comparable. As was expected, increasing the load on the network resulted in a nonlinear growth of the loss probability, which was higher for low-priority requests. However, the placement of the SDN controller had no influence on the loss probabilities because the path calculation algorithm was the same regardless of the placement of the SDN controller.

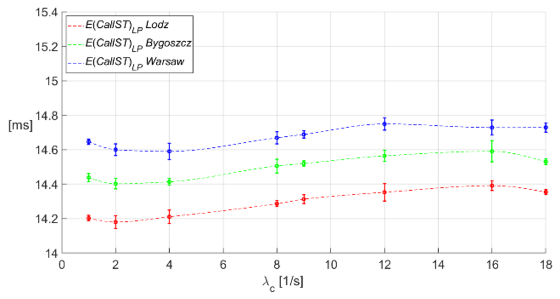

FIGURE 13. Call setup time for LP.

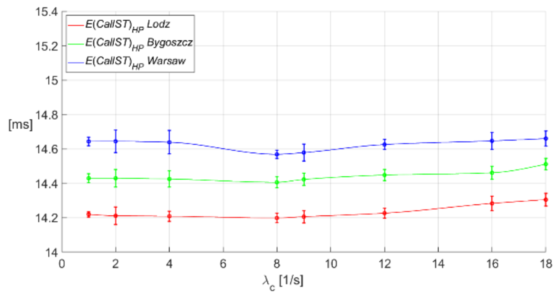

FIGURE 14. Call setup time for HP.

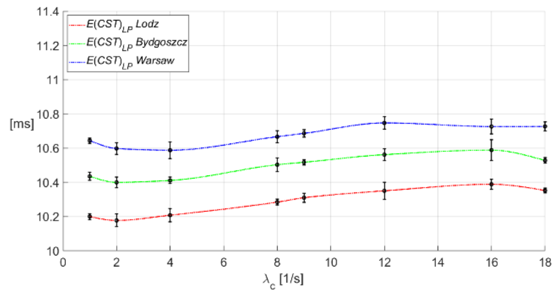

FIGURE 15. Connection setup time for LP.

It is apparent from the figures that Lodz was the best location for the SDN controller in the three analyzed variants. For this location, we obtained the lowest values of E(CallST), E(CST), E(CallRT) and E(CRT) for call requests successfully and unsuccessfully established, while the loss

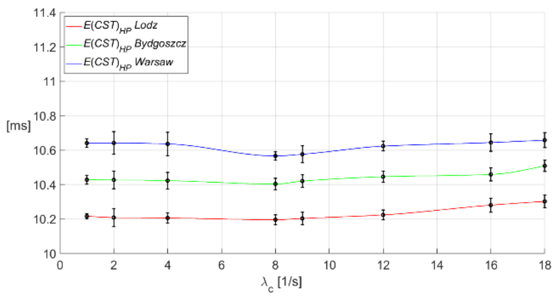

FIGURE 16. Connection setup time for HP.

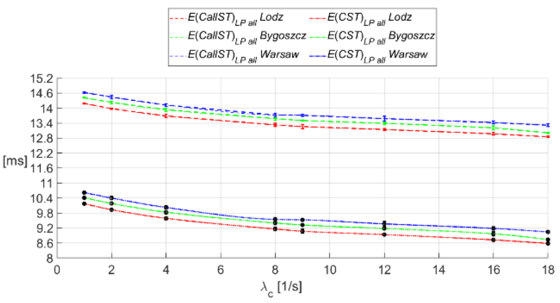

FIGURE 17. Call setup time and connection setup time for LP.

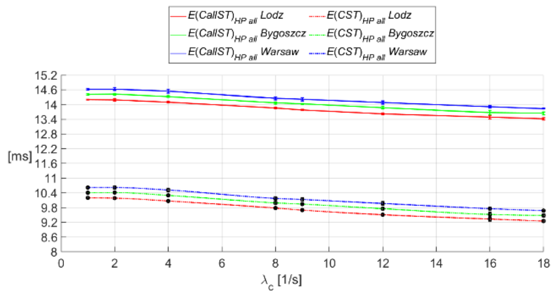

FIGURE 18. Call setup time and connection setup time for HP.

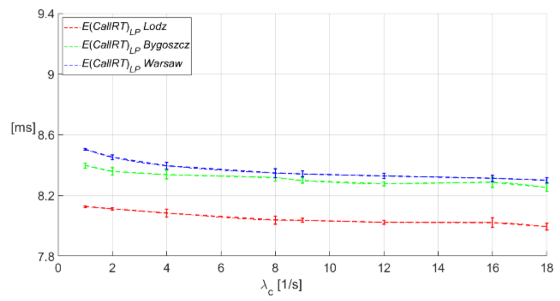

FIGURE 19. Call release time for LP.

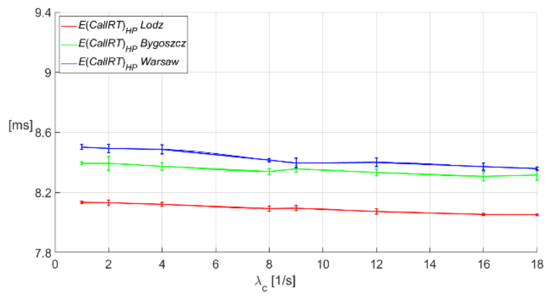

FIGURE 20. Call release time for HP.

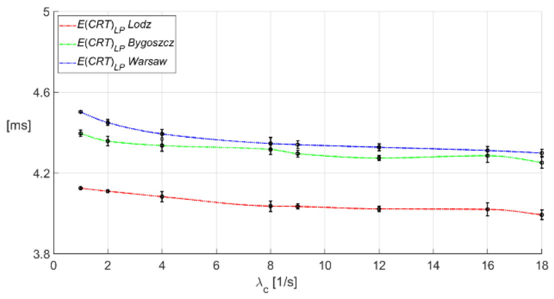

FIGURE 21. Connection release time for LP.

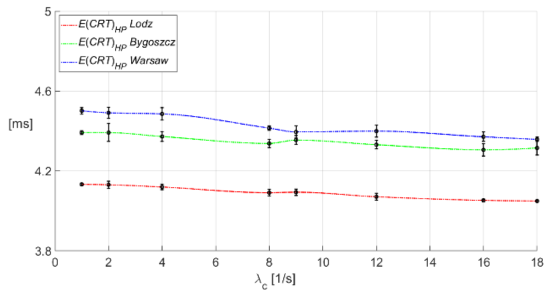

FIGURE 22. Connection release time for HP.

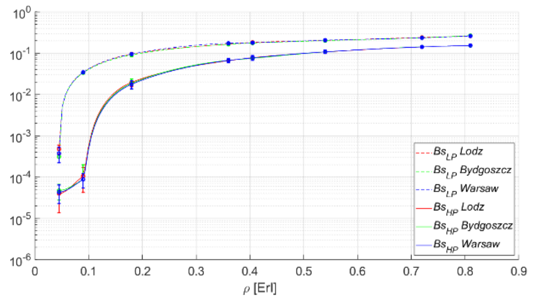

FIGURE 23. Link loss probability for LP and HP.

probabilities for all the analyzed placements were statistically insignificant for the same call request priority, and the confidence intervals overlapped.

In the next part of this section, selected simulation results are presented in order to compare and determine whether it is possible to guarantee appropriate quality for two extremes regarding the quality parameters for the SDN controller placements: Lodz and Warsaw. The comparison is presented in Figs. 26–33 for the SDN controller placements in the Lodz and Warsaw nodes.

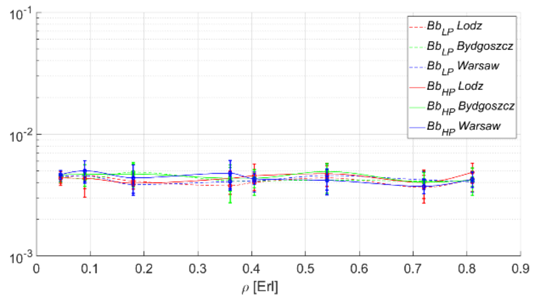

FIGURE 24. OXC blocking probability for LP and HP.

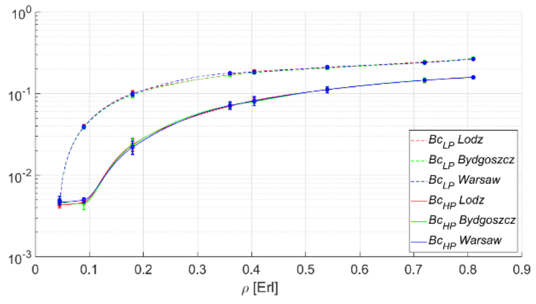

FIGURE 25. Total loss probability for LP and HP.

Fig. 26 and Fig. 28 show a comparison of $E(\text{CallST})$ and $E(\text{CST})$ for the call requests completed successfully.

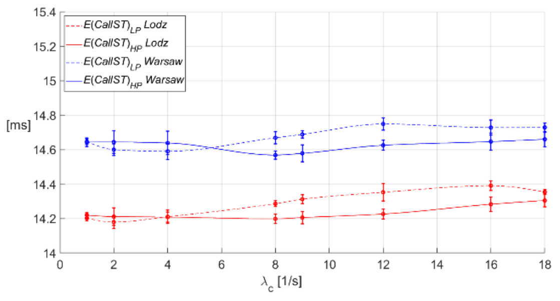

FIGURE 26. Call setup time for LP and HP.

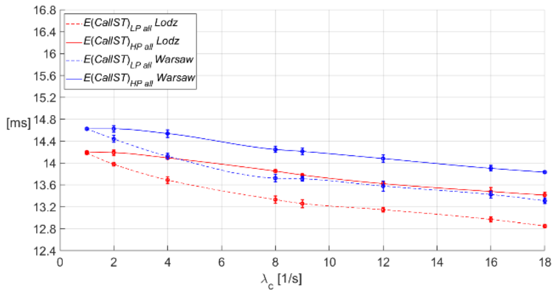

FIGURE 27. Call setup time for LP and HP.

While the results of $E(\text{CallST})$ and $E(\text{CST})$ for low- and high-priority requests were not statistically different for intensities below 8 requests per seconds, above this value, there were significant differences in the times for both the Lodz and Warsaw placements.

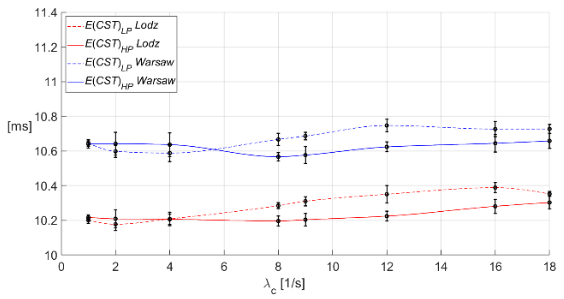

FIGURE 28. Connection setup time for LP and HP.

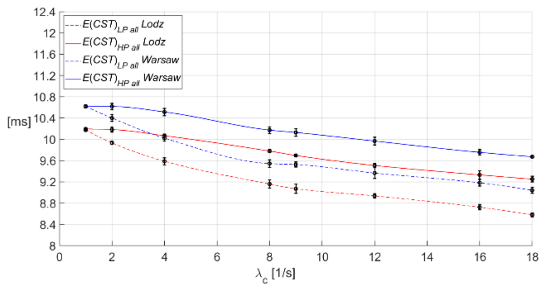

FIGURE 29. Connection setup time for LP and HP.

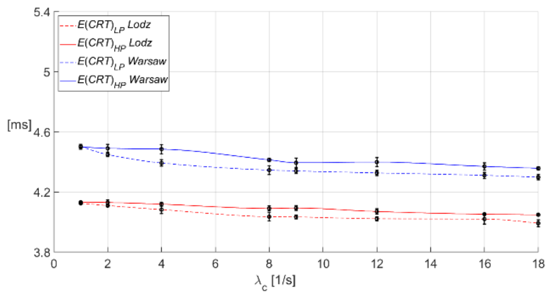

FIGURE 30. Call release time for LP and HP.

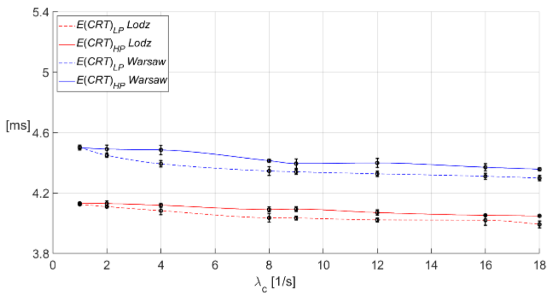

FIGURE 31. Connection release for LP and HP.

Taking into consideration all requests successfully and unsuccessfully established, Fig. 27 and Fig. 29 show a clear trend toward lower $E(\text{CallST})$ and $E(\text{CST})$ values with the request intensity as a result of the growing total loss probabilities. The implemented model has no wavelength converters and for higher intensities, optical lightpaths were established on smaller distances. A comparison of $E(\text{CallRT})$ and $E(\text{CRT})$ is presented in Fig. 30 and Fig. 31.

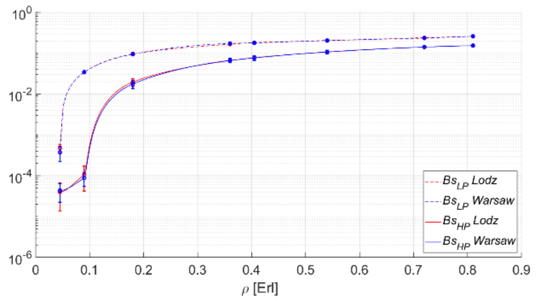

FIGURE 32. **Link loss probability for LP and HP for Lodz and Warsaw.**

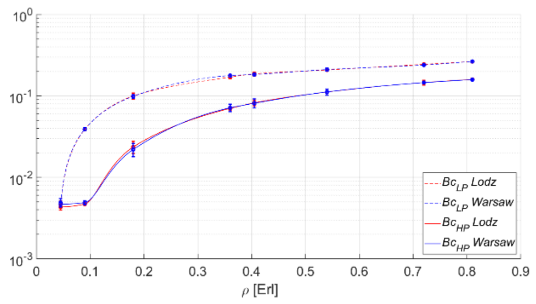

FIGURE 33. **Total loss probability for Lodz and Warsaw.**

A comparison of the results of the loss probabilities for the Lodz and Warsaw placements is presented in Fig. 32 and Fig. 33.

The presented results make it reasonable to draw synthetic conclusions on how the proposed SDN controller affects all the analyzed Quality of Service parameters:

- The proposed SDN controller is sufficient to guarantee an appropriate quality of service for different classes of services.
- The placement of the proposed SDN controller affects E(CallST), E(CST), E(CallRT), E(CRT).
- The E(CallRT) placement of the SDN controller has no influence on loss probabilities.

VI. CONCLUSION

Due to the complexity of optical transport networks, there are still many difficulties ahead in implementing the SDN concept, which was designed for packet networks and cannot be simply applied to optical architectures. However, as we look ahead, the role of SDN in optical networks will continue to expand, shaping the future of optical network control.

The presented implementation of an SDN controller is the first proposition in which an SDN controller is built on ASON network controllers, and meets the functional requirements specified in the [3] recommendation. We obtained satisfactory simulation results, proving that the presented SDN controller implementation is suitable for guaranteeing an appropriate quality of service. Notwithstanding this, the results provide valuable insights into the issue of whether the implemented SDN controller is suitable for optical network control. The proposition may be applicable to multi-controller SDN architectures due to SDN controller modular design and possibility of connection controllers (CC) coordination, as proposed in [1].

The key advantage of the solution we propose here is the implementation of an SDN controller compliant with the guidelines outlined in [3], while ensuring compatibility with the standards established for OpenFlow. Moreover, we extended OpenFlow protocol with the messages necessary for the optical network according to the ONF specification. Concurrently, we proposed a solution in which the ASON/GMPLS control plane components support the operations of the SDN controller in order to provide better scalability. The solution proposed in the paper was verified by simulation using the OMNeT++ simulation environment.

Based on the positive results of the conducted functional and performance tests, future work should concentrate on more comprehensive research. The model structure and configuration of the simulation model make it possible to carry out research under various sets of traffic conditions, assuming different network structures, link lengths, and throughputs, including different optical transport resources. The presented simulator supports providing simulations for static grid optical wavelengths. Currently, the authors are implementing a flex grid optical network and would like to execute simulations for the SDN/ASON/GMPLS architecture, assuming single and multidomain networks. Finally, the authors are developing an analytical model for analyzing and synthesising the performance of the SDN/ASON/GMPLS network.

REFERENCES

[1] *Architecture for the Automatically Switched Optical Network*, document ITU-T Rec. G.7703, May 2021.

[2] *Generalized Multi-Protocol Label Switching (GMPLS) Architecture*, document IETF RFC 3945, Oct. 2004.

[3] *Architecture for SDN Control of Transport Networks*, document ITU-T Rec. G.7702, Apr. 2022.

[4] S. Manzoor, N. I. Ratyal, and H. G. Mohamed, "Achieving QoS in smart cities using software defined Wi-Fi networks," *IEEE Access*, vol. 11, pp. 98256–98268, 2023, doi: 10.1109/ACCESS.2023.3313249.

[5] *Common Control Aspects*, document ITU-T Rec. G.7701, Apr. 2022.

[6] *OpenFlow Switch Specification, Version1.5.1*, ONF Technical Specification-0.25, Mar. 2015.

[7] *OMNeT++ Discrete Event Simulator*. Accessed: Oct. 8, 2024. [Online]. Available: https://omnetpp.org

[8] SDN Architecture, ONF Technical Reference-502, Jun. 2014.

[9] R. Alvizu and G. Maier, "Can open flow make transport networks smarter and dynamic? An overview on transport SDN," in *Proc. Int. Conf. Smart Commun. Netw. Technol. (SaCoNeT)*, Vilanova i la Geltru, Spain, Jun. 2014, pp. 1–6, doi: 10.1109/SaCoNeT.2014.6867771.

[10] M. N. A. Sheikh, M. Halder, S. S. Kabir, M. W. Miah, and S. Khatun, "SDN-based approach to evaluate the best controller: Internal controller NOX and external controllers POX, ONOS, RYU," *Proc. Global J. Comput. Sci. Techn.*, vol. 19, pp. 30–42, Jan. 2019, doi: 10.34257/GJCSTEVOL19IS1PG21.

[11] M. Shirazipour, Y. Zhang, N. Beheshti, G. Lefebvre, and M. Tatipamula, "OpenFlow and multi-layer extensions: Overview and next steps," in *Proc. Eur. Workshop Softw. Defined Netw.*, Darmstadt, Germany, Oct. 2012, pp. 13–17, doi: 10.1109/EWSDN.2012.22.

[12] M. Channegowda, R. Nejabati, M. Rashidi Fard, S. Peng, N. Amaya, G. Zervas, D. Simeonidou, R. Vilalta, R. Casellas, R. Martínez, R. Muñoz, L. Liu, T. Tsuritani, I. Morita, A. Autenrieth, J. P. Elbers, P. Kostecki, and P. Kaczmarek, "Experimental demonstration of an OpenFlow based software-defined optical network employing packet, fixed and flexible DWDM grid technologies on an international multi-domain testbed," *Opt. Exp.*, vol. 21, no. 5, pp. 5487–5498, Mar. 2013, doi: 10.1364/oe.21.005487.

[13] L. Liu, T. Tsuritani, I. Morita, H. Guo, and J. Wu, "Experimental validation and performance evaluation of OpenFlow-based wavelength path control in transparent optical networks," *Opt. Exp.*, vol. 19, no. 27, pp. 26578–26593, Dec. 2011, doi: 10.1364/oe.19.026578.

[14] S. Sultana, R. R. Reyes, and T. Bauschert, "Control-plane traffic modelling for connection management in T-SDN optical networks using transport PCE and OpenROADM," in *Proc. Photonic Netw., 23th ITG-Symp.*, Berlin, Germany, May 2022, pp. 1–9.

[15] R. Casellas, R. Vilalta, R. Martínez, and R. Muñoz, "Highly available SDN control of flexi-grid networks with network function virtualization-enabled replication," *J. Opt. Commun. Netw.*, vol. 9, no. 2, pp. 207–215, Feb. 2017, doi: 10.1364/JOCN.9.00A207.

[16] (Sep. 2024). *Interworking of GMPLS Control and Centralized Controller Systems, TEAS Working Group, Internet Draft, Haomian Zheng*. [Online]. Available: https://www.ietf.org/archive/id/draft-ietf-teas-gmpls-controller-inter-work-17.txt

[17] S. Kaczmarek and M. Mlynarczuk, "Simulator for performance evaluation of ASON/GMPLS network," *IEEE Access*, vol. 9, pp. 108293–108304, 2021, doi: 10.1109/ACCESS.2021.3101021.

[18] S. Kaczmarek, M. Sac, and K. Bachorski, "Implementation of IMS/NGN transport stratum based on the SDN concept," *Sensors*, vol. 23, no. 12, p. 5481, Jun. 2023, doi: 10.3390/s23125481.

[19] S. Kaczmarek and J. A. Litka, "Impact of SDN controller's performance on quality of service," *IEEE Access*, vol. 12, pp. 8262–8282, 2024, doi: 10.1109/ACCESS.2024.3352433.

[20] S. Orlowski, R. Wessäly, M. Pióro, and A. Tomaszewski, "SNDlib 1.0—Survivable network design library," *Netw., Int. J.*, vol. 55, no. 3, pp. 276–286, Oct. 2009.

[21] A. Szymanski, A. Lason, J. Rzasa, and A. Jajszczyk, "Grade-of-service based routing in optical networks [quality-of-service-based routing algorithms for heterogeneous networks]," *IEEE Commun. Mag.*, vol. 45, no. 2, pp. 82–87, Feb. 2007, doi: 10.1109/MCOM.2007.313400.

[22] S. Kaczmarek and M. Mlynarczuk, "DWDM network laboratory solution for telecommunication education engineering," in *Proc. Int. Conf. Softw., Telecommun. Comput. Netw. (SoftCOM)*, Hvar, Croatia, Sep. 2021, pp. 1–6, doi: 10.23919/SoftCOM52868.2021.9559118.

[23] *SIP-based Call Processing Performance*, document ITU-T Rec. Y.1531, Nov. 2007.

[24] *Proxmox Virtual Environment*. Accessed: Oct. 30, 2024. [Online]. Available: https://www.proxmox.com/en/proxmox-virtual-environment/overview

SYLWESTER KACZMAREK received the M.Sc. degree (Hons.) in electronics engineering and the Ph.D. (Hons.) and D.Sc. degrees in switching and teletraffic science from Gdańsk University of Technology (GUT), Poland, in 1972, 1981, and 1994, respectively.

From 1972 to 1981, he was an Assistant with the Faculty of Electronics, Telecommunications and Informatics (FETI), GUT. He was an Assistant Professor (1981–1994) and an Associate Professor (1994–2005). Since 2005, he has been a Professor with FETI. From 2007 to 2018, he was the Head of the Teleinformation Networks Department, FETI. He has published two monographs and more than 270 articles. Among his promoted students, there are 304 engineers and M.Sc. and seven Ph.D. students. He has participated and directed projects and research and development grants. His research interests include IPQoS, GMPLS, ASON, OTN and SDN networks, switching, QoS routing, teletraffic, and multimedia services. His current research interests include the development and applicability of VoIP, IMS/NGN, OTN, and SDN technologies. Among his distinctions, there is a TPC membership of conferences, such as ITNAC and KRiT.

MAGDALENA MŁYNARCZUK received the M.Sc. (Eng.) degree in telecommunication systems and networks from the Faculty of Electronics, Telecommunications, and Informatics (FETI), Gdańsk University of Technology (GUT), in 2004, the M.Sc. degree in economy and finance from the Faculty of Management and Economics, GUT, in 2007, and the Ph.D. degree (Hons.) in information and communication technology in engineering and technology sciences, in 2022.

From 2008 to 2011, she was an Assistant with FETI, GUT, where she was also a Lecturer with the Department of Teleinformation Networks, from March 2011 to September 2017. From October 2017 to February 2022, she was an Assistant with FETI, GUT. Since March 2022, she has been an Assistant Professor, where she gives classes related to her research. She has also actively participated in several research and development projects/grants conducted with the Department of Teleinformation Networks. She has published more than 50 research works. Her research interests include control of optical networks in ASON, GMPLS, and NGN networks, transmission and switching technology, teletraffic engineering, and voice over IP technology.

MACIEJ SAC received the M.Sc. (Eng.) degree in telecommunications and the Ph.D. degree in information and communication technology from the Faculty of Electronics, Telecommunications, and Informatics (FETI), Gdańsk University of Technology (GUT), Poland, in 2009 and 2022, respectively.

Since 2013, he has been with the Department of Teleinformation Networks (DTN), FETI, GUT, as a Lecturer (2013–2016), an Assistant (2016–2022), and an Assistant Professor (since 2022). He teaches classes related to the broad area of telecommunications systems and networks, transmission techniques, information theory and coding, digital signal processing, and probabilistic methods. He has published more than 60 research works. He has also actively participated in several research and development projects/grants conducted in DTN. His research interests include IP QoS, VoIP, IMS/NGN, SDN networks, multimedia services, and teletraffic engineering.

PATRYK MIKLASZEWSKI was born in Ciechanow, Masovian Voivodeship, Poland, in 1998. He received the B.Eng. degree in electronics and telecommunications, with a specialization in radio communications systems and networks and the M.Sc. degree in electronics and telecommunications, with a specialization in teleinformation systems and networks from Gdańsk University of Technology, Gdańsk, Pomeranian Voivodeship, Poland, in February 2021 and October 2022, respectively.

From September 2022 to September 2024, he was an Associate System Integration Test Engineer with Adva Optical Networking (currently Adtran Networks). Since October 2024, he has been a System Integration Test Engineer with Adtran Networks. His research interests include software defined networks (SDN) and in particular, the use of centralized software-based control to manage network operations and introduce new network services, the development of optical technologies for these networks, and strategies to enhance their overall efficiency and performance.