ZSM PoC#3 Final Report

1 PoC Project Details

1.1 PoC Project Review

PoC Number:	3
PoC Project Name:	Automation of Intent-based cloud leased line service (Phase two)
PoC Project Host:	Huawei Technologies Co.,Ltd, Canadian Research Centre
Short Description:	This PoC demonstrates the automation of the Intent-based cloud leased line (CLL) use-case. The use-case is developed on ONAP, and its implementation complies with the ZSM standards. The PoC demonstrates management services (defined in ZSM 002), cross-domain orchestration (ZSM 008), closed-loop (ZSM 009), and Intent-based networking (ZSM 011). In this demonstration, as opposed to our demo in phase one, we use real hardware (i.e. NEs) and real commercial network controllers to show a real network traffic flows (i.e. real data-plane).
PoC Project Status : *(Ongoing/Completed)*	Completed

1.2 PoC Team Members Review

	Organisation name	ISG ZSM participant (yes/no)	Contact (Email)	PoC Point of Contact (*)	Role (**)	PoC Components
1	China Telecom	Yes	**Dong Wang** wangd5@chinatelecom.cn	X	Network operator	(1) ONAP development; (2) provide lab facilities to host ONAP for the PoC
2	China Mobile	No	**Keguang He** hekeguang@chinamobile.com		Network operator	ONAP development
3	China Unicom	Yes	**Yanlei Zheng** zhengyanlei@chinaunicom.cn		Network operator	(1) Contribute to technical details of data-plane operations in the Transport Domain (2) Provide lab facilities to host the network equipment for the PoC
4	Huawei Technologies	Yes	**Henry Yu** henry.yu1@huawei.com **Hesam Rahimi** hesam.rahimi@huawei.com	X	Network supplier	(1) ONAP development; (2) Provide domain controller(s) and network equipment for the PoC
5	AsiaInfo Technologies	No	**Lei Shi** shilei8@asiainfo.com		ONAP project partner	ONAP development
6	Xidian University	No	**Chungang Yang** cgyang@xidian.edu.cn		University research partner	Research on Intent translation algorithms to improve the translation accuracy
(*) Identify the PoC Point of Contact with an X. (**) The Role will be network/service provider, supplier, or other (universities, research centers, test labs, Open Source projects, integrators, etc…).						

All the PoC Team members listed above declare that the information in this report is conformant to their activities during the PoC Project.

1.3 PoC Project Scope Review

1.3.1 PoC Topics

Report the status of all the PoC Topics and Expected Contributions anticipated in the PoC Proposal.

PoC Topic Code	PoC Topic Description	Related WI	Submitted Contribution link	Date	Status (*)
Topic 3 (Intent-driven	Automation of the Intent-based cloud leased line service deployment and the	ZSM 009, ZSM 011	The PoC showcased at the ETSI ZSM(#21) meeting in Sophia-Antipolis, France as well as the	December 2022	Completed

Closed-loop automation)	closed-loop operations for service assurance		Layer123 World Congress in London, England		
Topic 4 (Cross-domain user-driven E2E services)	Demonstration of the cross-domain orchestration and automation between the E2E MD and the Transport MD	ZSM 002, ZSM 008	The PoC showcased at the ETSI ZSM(#22) meeting in Sophia-Antipolis, France as well as the Layer123 World Congress in London, England	December 2022	Completed
(*) Planned, On-going, Completed, delayed (new target date), Abandoned					

1.3.2 Other topics in scope

List here any additional topic for which the PoC plans to provide input/feedback to the ISG ZSM.

PoC Topic Code	PoC Topic Description	Related WI	Submitted Contribution link	Date	Status (*)
(*) Planned, On-going, Completed, delayed (new target date), Abandoned					

1.4 PoC Project Milestones Review

PoC Milestone	Milestone description	Target Date	Additional Info
P.S	PoC Project Start	Jan 2022	
P.D1	PoC Demo 1	May 2022	Automated CPL service creation and modification using ZSM framework (ZSM 002) (simulated hardware)
P.D2	PoC Demo 2	May 2022	Closed-loop in Transport MD (ZSM 009) (simulated hardware). Please note that the ONAP software and setup used in Demo 1 will also be used in Demo 2. In other words, Demo 2 will be built upon Demo 1.
P.D1 and P.D2	PoC Demo 1 and 2	June 2022	The Demo 1 & 2 was also presented at Linux Foundation (LFN) Developer and Testing Forum (DTF) held in June 2022 in Porto, Portugal
P.C1	Contribution on lessons learned from Demo 1 & 2	July 2022	From Demo 1 & 2, collect feedback about the ONAP design and lessons learned about the PoC, in order to make improvements in Demo 3
P.D3	PoC Demo 3	Nov 2022	Demo with real hardware and real data traffic. This PoC relates to ZSM 002, ZSM 008, ZSM 009, and ZSM 011.
P.C2	Contribution on lessons learned from Demo 3	Dec 2022	Lessons learned and potential improvements of the PoC is documented in the PoC Report.
P.R	PoC Report	Dec 2022	
P.E	PoC Project End	Dec 2022	

1.5 Confirmation of PoC Events Occurrences

This PoC was presented in two separate face-to-face events and the details are captured below:

- **Events Names and Occurrences**:
    - ETSI ZSM#19 held from May 15 to 19, 2022 in Sophia-Antipolis, France.
    - ETSI ZSM#21 held from November 22 to 24, 2022 in Sophia-Antipolis, France.
    - Layer123 World Congress held from December 5 to 7, 2022 in London, UK.
- **Events descriptions**:
    - ETSI ZSM#21 meeting: Performed Demo 1 and Demo 2
    - ETSI ZSM#21 meeting: Performed Demo 3
    - Layer123 World Congress: A ZSM workshop was held at the conference. A presentation on PoC#3, which describes the use-case, technical design (high level), and demo procedures and results, was given at the workshop.

1.6 Other dissemination activities

To reach a wider audience, the ISG ZSM PoC#3 (phase 1) was also presented at the Linux Foundation (LFN) Developer and Testing Forum (DTF), held on June 13-16, 2022 in Porto, Portugal. LFN project technical communities were gathered face-to-face to progress their releases; discuss project architecture, direction, and integration points; and further innovate through the open source networking stack. This PoC#3 was presented for LFN's ONAP community:

https://wiki.lfnetworking.org/display/LN/2022-06-15+-+ONAP%3A+Automation+of+Intent-based+Cloud+Leased+Line+Service

2 ZSM PoC#3 Technical Details

2.1 PoC Overview

This section details the rationale and motivation behind this PoC#3, followed by its objectives, its use-case description, its scope and content.

2.1.1 Problem statement

Cloud services are constantly reshaping social production and lifestyles. Home broadband applications (such as cloud games), large enterprises (such as governmental and finance enterprises), and small- and medium-sized enterprises (such as hospitals and e-commerce enterprises) are greatly facilitated by cloud services. Enterprises are using cloud services to reduce O&M costs, and applications are gradually being migrated to the cloud. The multi-cloud strategy and multi-cloud services have become a trend in the industry. One-hop cloud access via OTN has been widely used in the industry and has become the mainstream choice for streamlining enterprises and cloud leased line services due to its various advantages, including high bandwidth, low latency, hard isolation, high reliability, and one-hop cloud access. However, it has always been difficult to manage services on optical transport networks due to network configuration complexity and multi-vendor interoperability and collaboration challenges. This PoC proposes a solution to these challenges.

2.1.2 PoC objectives

To cope complexity and the challenges mentioned above, the architectural design of this PoC, which is based on ZSM 002, 008, 009-1 and 011, is a turnkey solution which provides fully automated and Intent-based service fulfilment and assurance of cloud leased lines. Secondly, the solution uses IETF ACTN MPI standard interfaces to solve the challenges of multi-vendor and multi-domain interoperability in the transport management domain. The solution implementation is open source and is released in LFN ONAP Kohn release.

2.1.3 Use-case Description

Cloud private line (CPL) services connect cloud service users to edge or cloud data centres, and edge or cloud data centres to each other, with deterministic connection performance. They may represent point-to-point, point-to-multipoint, multipoint-to-point, or multipoint-to-multipoint connectivity service topologies, and may be implemented using connection-less or connection-oriented paradigm-supporting technologies. Data flow mapping to CPL services is port, packet- or frame-oriented and CPL services may or may not include inspection and differential mapping of frames to connection services at connection service ingress ports. Service and/or supporting technology types include Ethernet, MPLS (Multi-Protocol Label Switching), OTN (Optical Transport Network) and DWDM (Dense Wavelength Division Multiplexing), solely or in combination. Collections of CPL services may be considered as Transport "Slices" and defined and provisioned collectively; here, we consider CPL services as defined and provisioned individually.

Fig. 1 illustrates an example of a CPL service infrastructure that uses EOO (Ethernet-over-OTN) technology. User data (e.g., IP packets or Ethernet frames) are carried by Carrier Ethernet Services, which operate over Ethernet Virtual Connections (EVCs). The EVCs provide service OAM (Operations, Administration and Maintenance), while service isolation and traffic protection are offered by the underlay OTN services. Right-sizing and topological configuration of service tunnels, and groups of tunnels, is provided by reconfigurable ODUFlex and DWDM technologies.

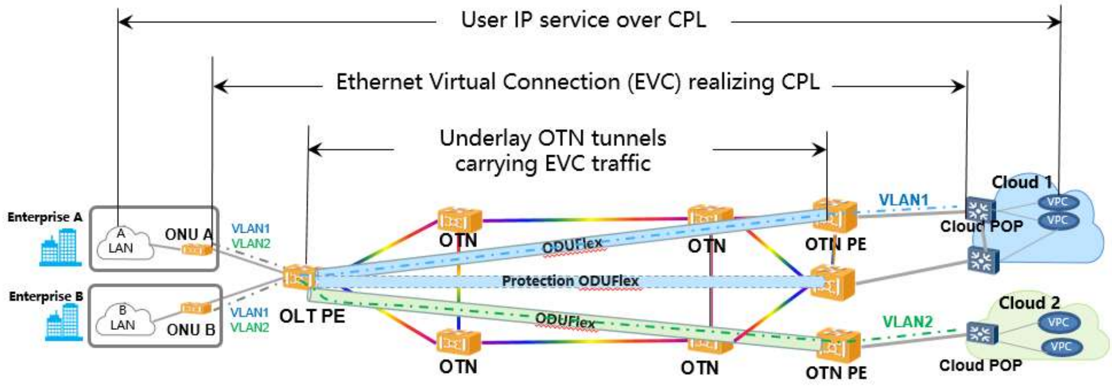

Figure 1. Illustration of a CPL service infrastructure based on EOO (Ethernet-over-OTN) technology.

Data centres may be operated by the CPL service customer, by the CPL services provider, by some other service provider(s), or by any combination of these. CPL service traffic consists in machine-to-machine data flows with a range of characteristics. Some data flows are essentially continuous, may require low or medium bandwidths, and may be anywhere from relatively latency-insensitive to highly latency-sensitive (e.g., synchronous data mirroring). Other data flows may comprise block data transfers, of varying sizes and completion time requirements, may occur on varying schedules, and may require small to very large bandwidths; they may also have varying latency sensitivities. The application drivers of individual data flows may depend on a range of application circumstances that may vary in time. Even CPL service availability and restoration requirements are variable and derive from application requirements associated with particular data flows.

There is demand, in multiple market segments, for dynamically user-driven mass-customized CPL services, having deterministic connection performance when in operation, to serve these requirements. Such a paradigm would replace both static port-to-port "large pipe" private line service, and statically-configured services that rely on statistically-based sharing of transport resources among data flows that require - perhaps significant - overprovisioning of resources to prevent service performance degradation under unfavourable aggregate data flow conditions. This new paradigm is useful to both service consumers and service providers, as: services may be closely matched to specific needs; service performance is deterministic in operation, providing e.g., determinism in block data transfer times; service delivery is network resource-efficient, as resources may be allocated to closely match the minimum detailed needs of every service; and services may be better-monetized, as no service parameter needs to be "given away free". Obviously, however, a dynamically mass-customized service paradigm, operated at any reasonable scale, requires a high degree of automation of service delivery and maintenance processes.

Dynamically mass-customized CPL services are driven operationally by CPL service consumers, either semi-manually (e.g., through a user-facing provisioning portal) or – more usefully – directly by consumer scheduling software systems. Intent is – obviously - a useful API and service-driving paradigm to support such capability.

2.1.3 Scope and Content of the PoC

This PoC demonstrates the service automation of the Intent-based CPL. The design architecture of the use-case follows the ZSM framework (ZSM 002) and other relevant ZSM standards (ZSM 009 and ZSM 011). Figure 2 depicts the use-case realization using the ZSM framework.

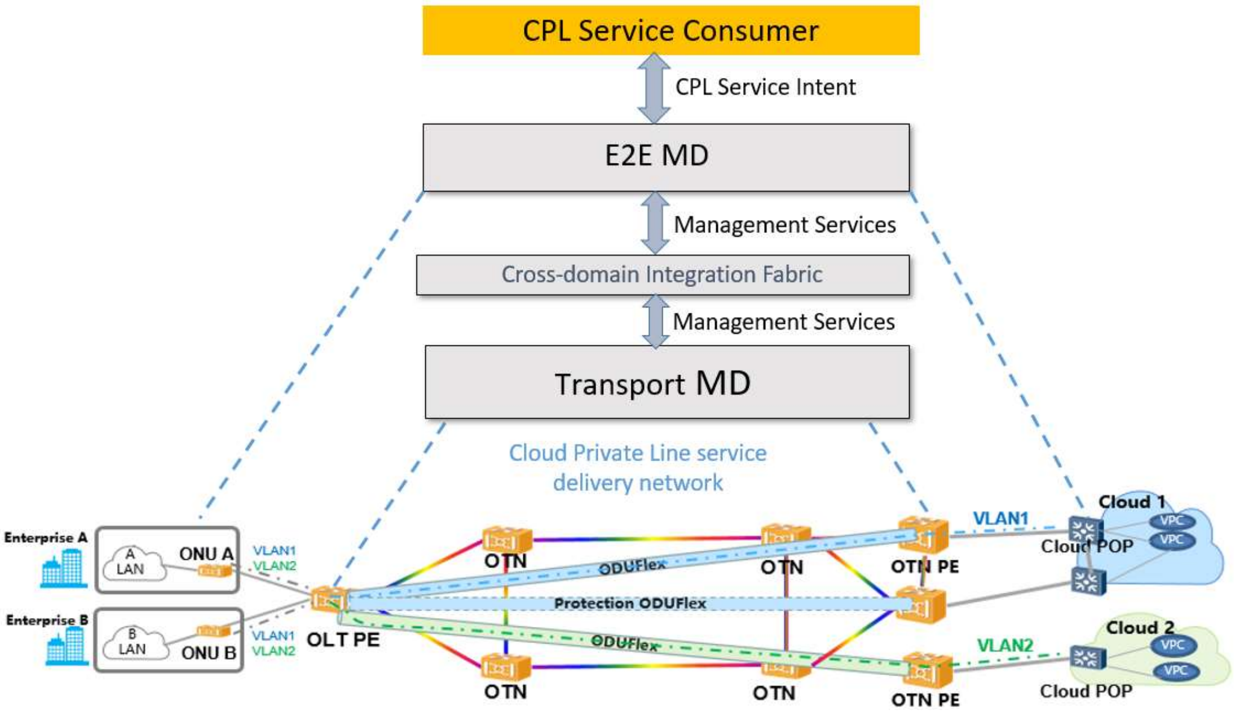

Figure 2 CPL use-case realization using ZSM framework

The service consumer expresses and sends the requests of a CPL service to the E2E MD via its Intent interface. The E2E MD is responsible for the E2E service orchestration. It translates the consumer's Intent into corresponding Management Services which are offered by the Transport MD(s). The cross-domain integration fabric facilitates the communication between the E2E and the Transport MD. The Transport MD registers its Management Services to the cross-domain integration fabric. The E2E MD then discovers and consumes these services.

The Transport MD is responsible for the service provisioning and network configurations of the physical network. Its northbound interfaces (i.e., Management Service APIs) are intent-like (in the case of this PoC), model-driven, technology agnostic, and open standard-based. Furthermore, the MD also implements the closed-loop defined in ZSM 009-1.

The PoC will demonstrate management services (defined in ZSM 002), cross-domain orchestration (ZSM 002), closed-loop (ZSM 009-1), and Intent-based networking (ZSM 011). Table 1 summaries the content of the PoC.

Table 1 Content of the PoC

Use-case scenario to be demonstrated	Demonstration of the ZSM features and capabilities which support the use-case scenario

The CPL service capability, offered by the Transport MD, is automatically discovered by the service discovery mechanism.	• Transport MD registers its management services to the cross-domain integration fabric • E2E MD discovers Transport services via the cross-domain integration fabric
The user (or Intent consumer) expresses an Intent of creating (or terminating) a CPL service. This Intent is then automatically fulfilled by provisioning (or de-provisioning) the corresponding services and allocating (or de-allocating) the required resources on the physical network.	• E2E MD interacts with the user (or Intent consumer) to fulfil the Intent and manages the Intent life cycle • E2E MD consumes the management services of the Transport MD in order to create (or terminate) the CPL services • Transport MD provides three management services, including control, orchestration, and data collection. They are necessary for the delivery of the CPL service
The Intent-based system monitors the SLA parameters of the CPL service (e.g., bandwidth usage), and automatically triggers the closed-loop actions (e.g., increase max bandwidth) in order to guarantee the SLA.	• Transport MD implements the closed-loop defined in ZSM 009-1. The PoC will demonstrate control flow consisting of monitoring, analysis, decision, and execution.

3 PoC architecture

Figure 3 depicts the PoC architecture. The transport network in this demo is a multi-domain (2-domain), multi-vendor (2-vendor) optical network. Each network domain is managed by a network controller. The network controllers, as well as their managed network equipment, may be supplied by different vendors, yet the northbound interface (NBI) of the controllers comply with the IETF/ACTN standards, in order to assure interoperability.

The total automation that we want to show in this PoC is to receive customer's need (intent) using natural language processing (NLP) techniques. For example, a simple user input describing customer's need in a natural language may be "I need a connection from company XYZ to Cloud ABC, with a bandwidth of 2Gbps". We demonstrate that after this, it will be a total automation process and the cloud leased-line will be provisioned on some real hardware (i.e NEs) and will have some real traffic.

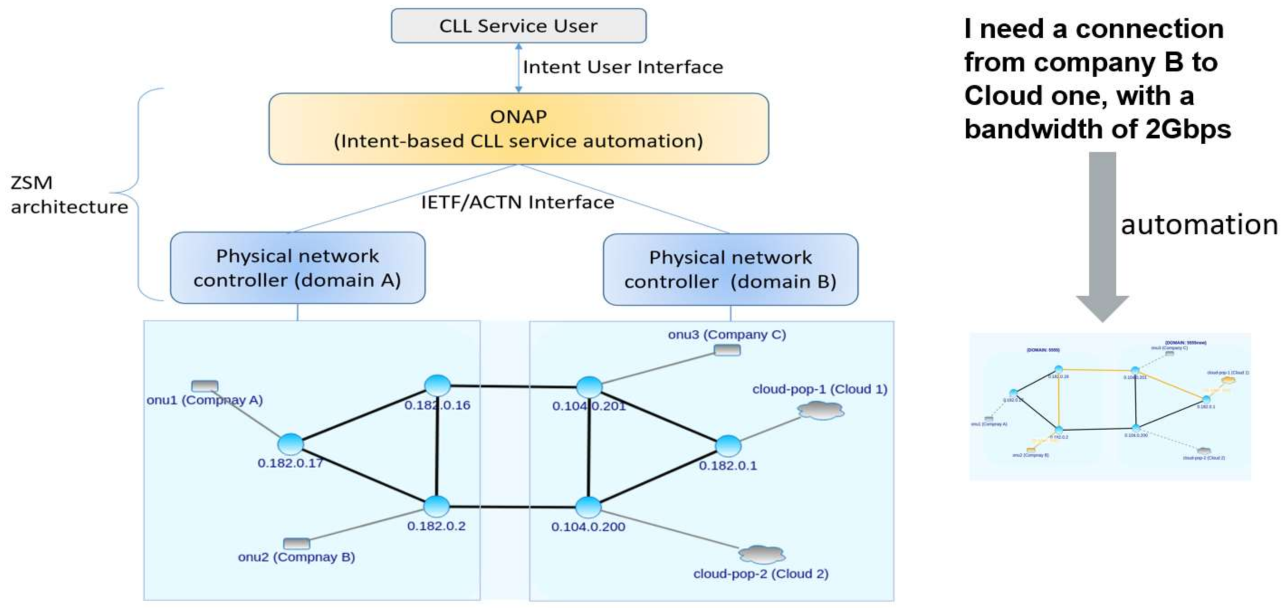

Figure 3: **PoC Architecture. The network automation framework follows the ZSM standards, and the interfaces of the network controllers (which serve as Management Functions) follow the IETF/ACTN standards**

The network controllers are controlled and orchestrated by ONAP. The overall architecture of this system follows the ZSM framework. More precisely, the E2E MD and the cross-domain integration fabric is implemented on ONAP. The Transport MD is implemented on both ONAP and network controllers. ONAP is a cloud-based platform. It consists of a collection of microservices, each of which has a dedicated network management responsibility and performs specific tasks.

The Intent-based CPL use-case, therefore, is developed on multiple ONAP microservices. Table 2 provides a list of the ONAP microservices on which new features/code are developed in order to realize the use-case.

Table 2 List of ONAP microservices which implement E2E MD, Cross-domain integration fabric, and Transport MD

ZSM component	ONAP microservices which implement the ZSM component
E2E MD	• UUI (Use-case UI) • DCAE (Data Collection, Analytics, and Events) • AAI (Active and Available Inventory)
Cross-domain integration fabric	• MSB (Microservices Bus) • SO (Service Orchestrator)
Transport MD	• SO • SDNC (SDN Controller) • AAI • DCAE • Policy • Network controllers (Note 1)

Note 1: Network controllers are outside of ONAP. But they are part of the Transport MD, serving as the Management Functions (MF).

Figure 4 depicts the implementation of the ZSM architecture on ONAP. ONAP's cloud-based architecture has many advantages. However, its distributed nature also introduces challenges on service management, discovery, orchestration, and deployment. This PoC shows how the adoption of the ZSM framework on ONAP may help overcome these challenges while still keeping the advantages of a cloud-based system.

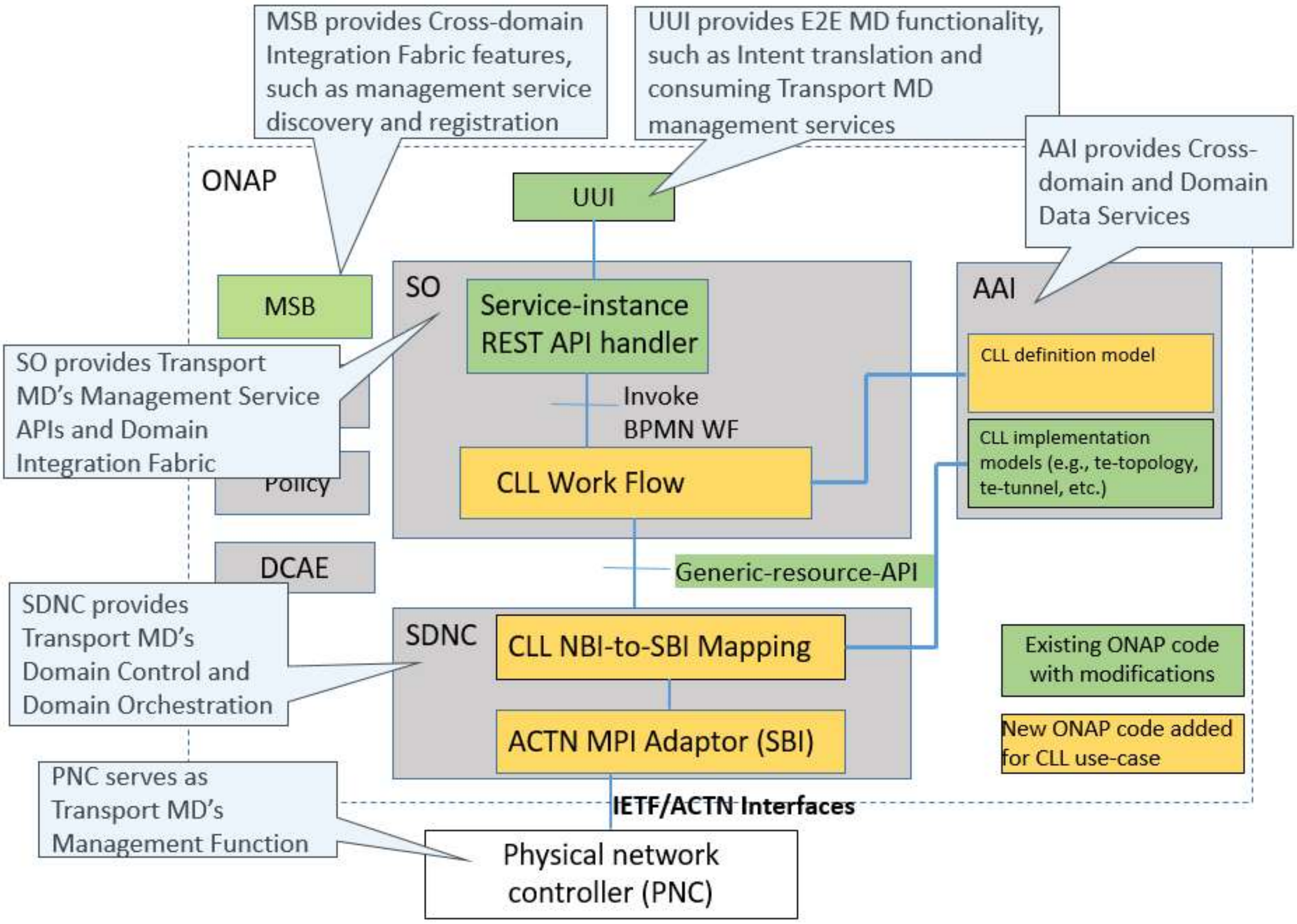

Figure 4 ONAP implementation of ZSM architecture

Another aspect of the PoC is to demonstrate the closed-loop capability of the Transport MD. Figure 6 depicts the closed-loop implementation on ONAP. The features are built top of the ZSM framework depicted in **Figure 4**, and the implementation follows ZSM 009-1.

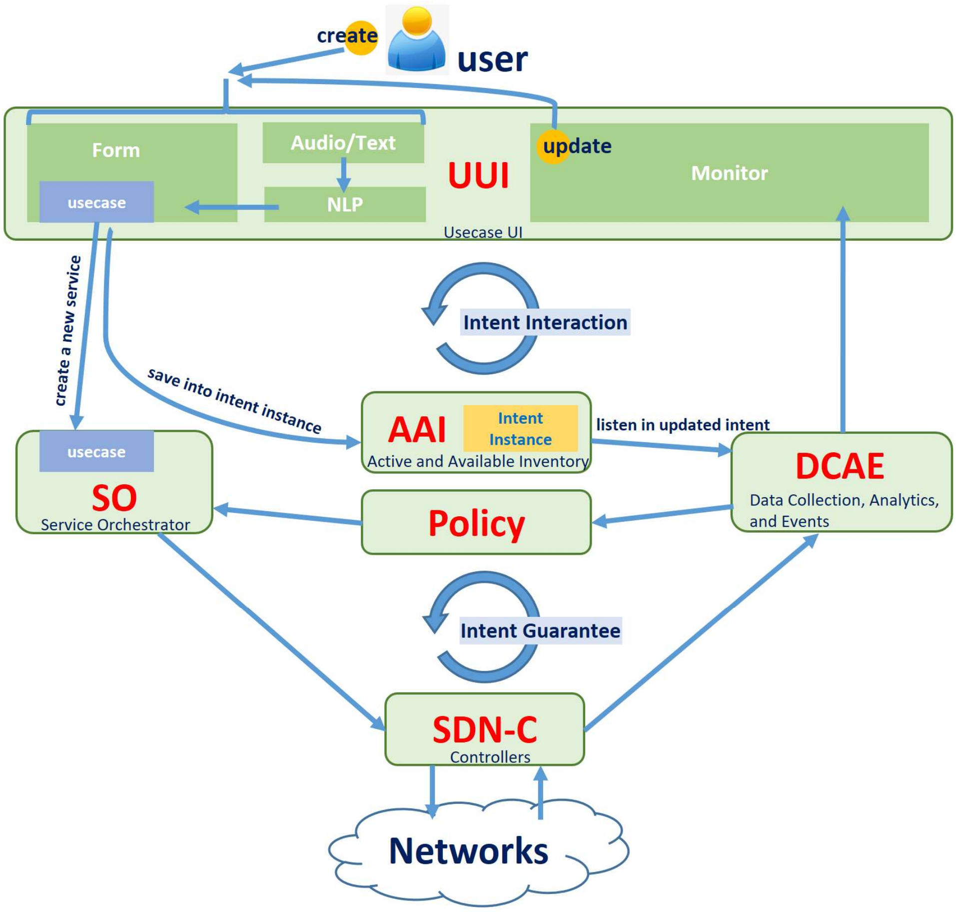

Figure 5 **Architecture of ONAP Intent-driven Closed-loop Autonomous Networks**

Intent interaction between users and the networks follows the intent interaction closed-loop of ONAP. As shown in Figure 5, this work refers to the architecture of ONAP intent-driven closed-loop autonomous networks and aligns with ZSM 011. Intent instance is applied to save users' real-time intent. The network provides performance monitoring and dynamic orchestration functions to satisfy users' real-time intent.

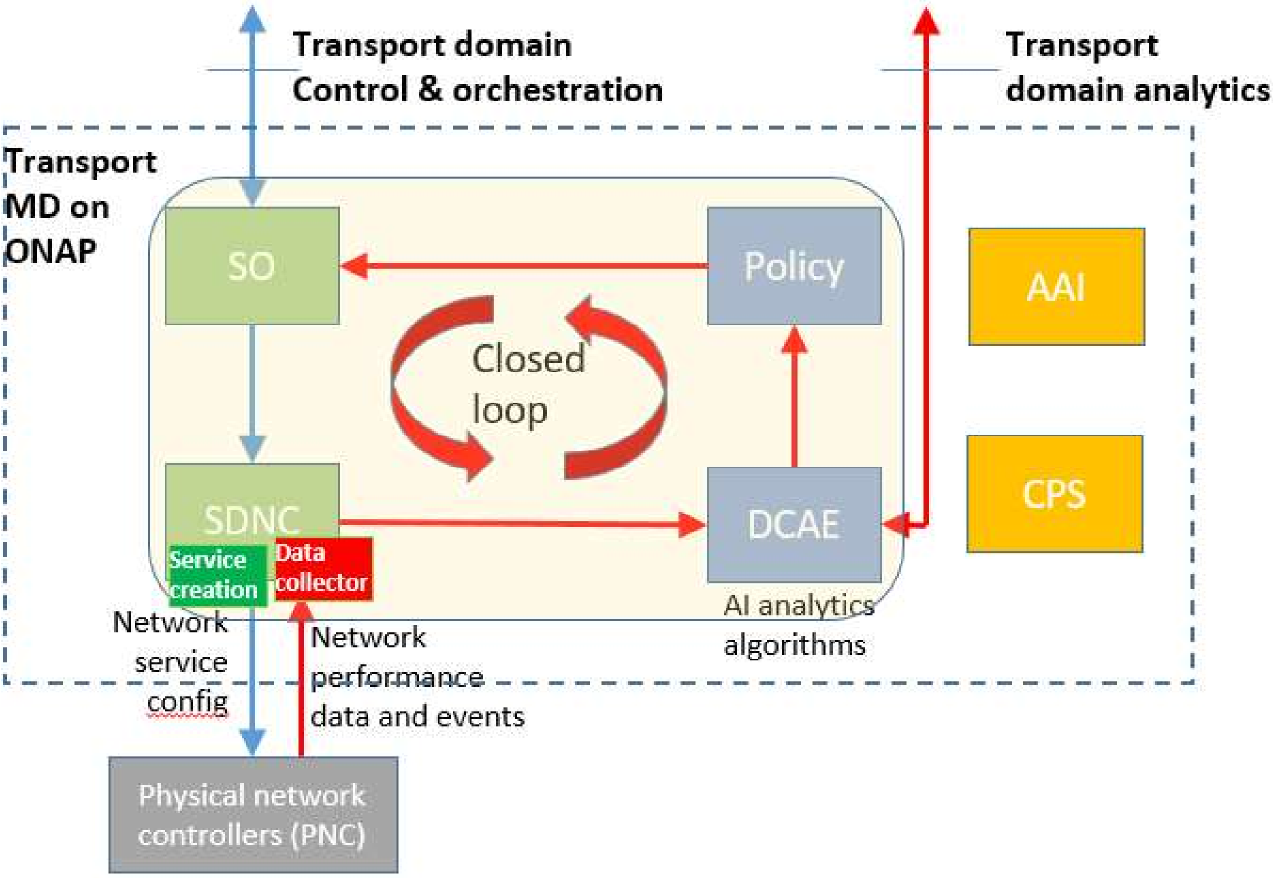

Figure 6 ONAP implementation of the closed-loop in the Transport MD

As specified in ZSM 009-1, the closed-loop control flow contains four stages: Monitoring, Analysis, Decision, and Execution. The Monitoring stage collects the performance data from the physical network, and then generates and sends data events to the Analysis stage for further processing. The Monitory stage is implemented on ONAP SDNC and physical network controllers (PNC).

The Analysis stage processes the incoming data events and generates and sends insights to the Decision stage. The Analysis stage is implemented on ONAP's DCAE.

The Decision stage creates the action plans based in the insights received from the Analysis stage, and then triggers the Execution stage to execute the actions. The Decision stage is implemented on ONAP's Policy microservice.

The Execution stage is realized by the domain control and domain orchestration management services, which are implemented on ONAP's SO and SDNC.

4 PoC Demonstration Procedure

4.1 PoC Setup

For the PoC demonstration setup, six Huawei POTN devices shown in Fig. 7 and two Huawei optical network controllers (iMaster NCE-T) shown in Fig. 8 are used to construct a two-domain network testbed for the cloud leased line test cases. This testbed is physically located in Wuhan, China in China Unicom and Huawei joint lab.

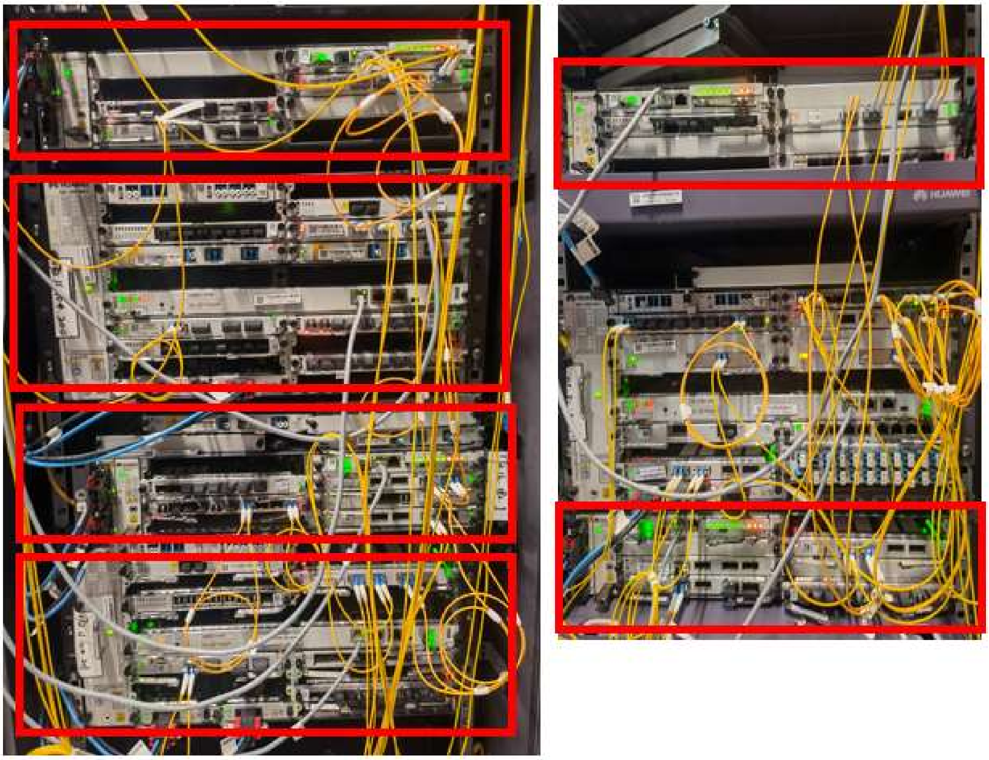

Figure 7: Six Network Nodes (providing Ethernet over OTN service)

Figure 8: two Huawei optical network controllers (iMaster NCE-T)

Apart from the lab in Wuhan China, we also have a cluster of ONAP servers physically located in Huawei Canada lab in Ottawa. ONAP serves as the user intent lifecycle manager and also the E2E network orchestrator.

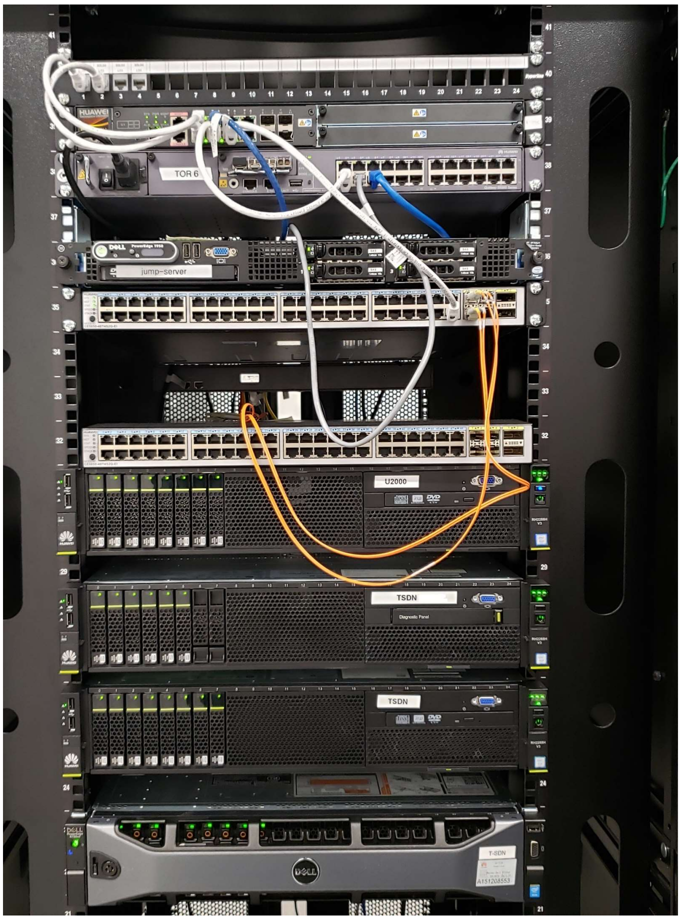

Figure 9: ONAP servers

Lastly, for this PoC demo, we used a Spirent Traffic Generator device shown in Fig. 10 to generate real-time Ethernet packets.

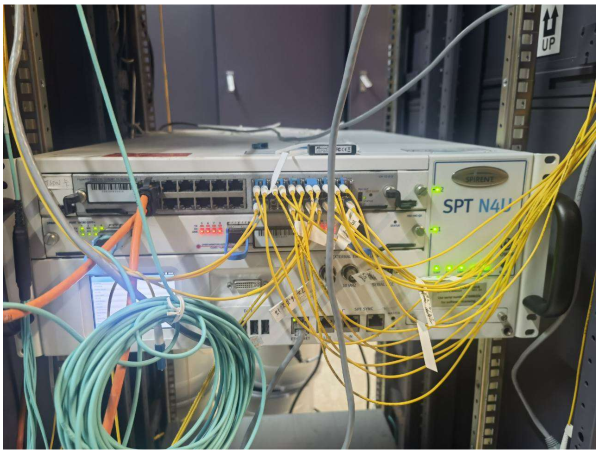

Figure 10: Spirent Traffic Generator (Ethernet Packets)

This demo setup confirms that this is indeed a distributed architecture where ONAP in Canada remotely controls the hardware in China. Since ONAP is a cloud-based platform, it allows this to happen.

4.2 Content of the demo and our test cases

For the content of the demo, we will create three cloud leased-line services on the six-node network described above. We will show that using natural language processing (NLP) techniques, the user's intent gets translated into a so-called intent records, and later gets deployed onto the hardware (i.e. NEs). Also, we show the intent life-cycle management by creating, deleting, and modifying user's intent. Last but not least, we show the closed-loop feature by performing a real-time bandwidth monitoring of the traffic of one of the CLL services and show that a closed-loop action will be automatically triggered when ONAP detects that the bandwidth usage of a service exceeds the originally provisioned bandwidth of that service. We show that ONAP automatically increases the provisioned bandwidth of that service. These are all shown in Fig. 11 below.

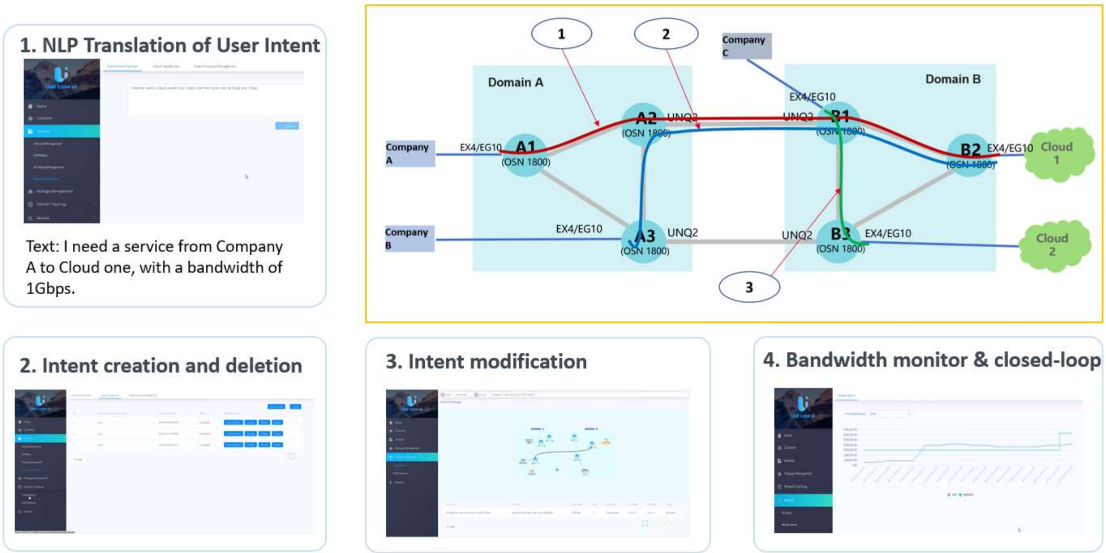

Figure 5: demo contents

4.2.1 Three-level verification of each test case

For each of the above-mentioned test cases, our test method contains three steps. First, at the intent level, we verify that if the intent record is created, modified, and deleted properly as expected. Then, at the network management level, we verify if the service is provisioned correctly on the hardware and lastly, at the data-path level, using the traffic generator, we verify that the data traffic patterns are consistent with the network configurations and intent operations. These three steps are illustrated in Fig. 12 below.

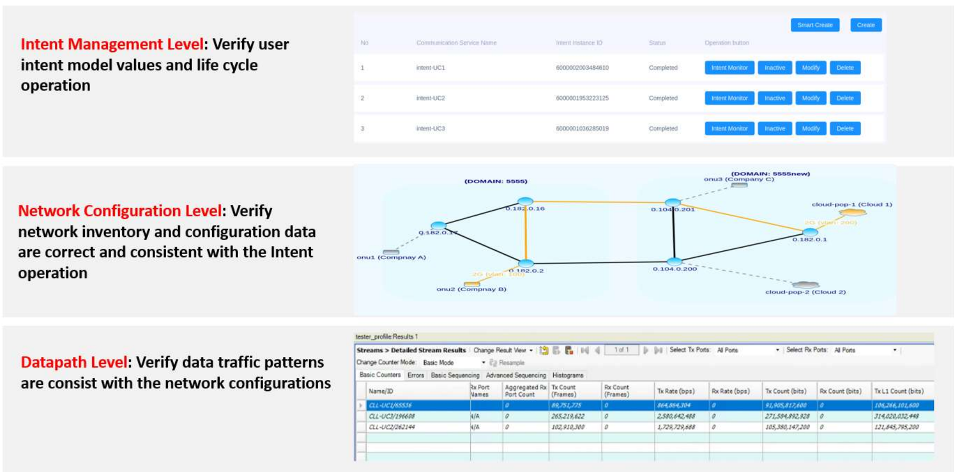

Figure 6: three-level verification of each test case

4.2.2 Service creation steps

The top part of the Fig. 13 below shows the initial condition of our demo topology in ONAP's UUI (i.e. Use-case UI) portal. This initial condition, as described above, consists of six nodes (NEs) in two domains. On the bottom part of the Fig.13, we are showing three streams of traffic flows (Ethernet packets) corresponding to the three cloud leased-lines that we will demonstrate in this PoC. These traffic flows are generated from three endpoints (i.e. Company A, Company B, and Company C) using our traffic generator device. As can be seen below, the initial condition of these streams only have Tx Counter and the Rx counter is 0, meaning that there is no service defined yet.

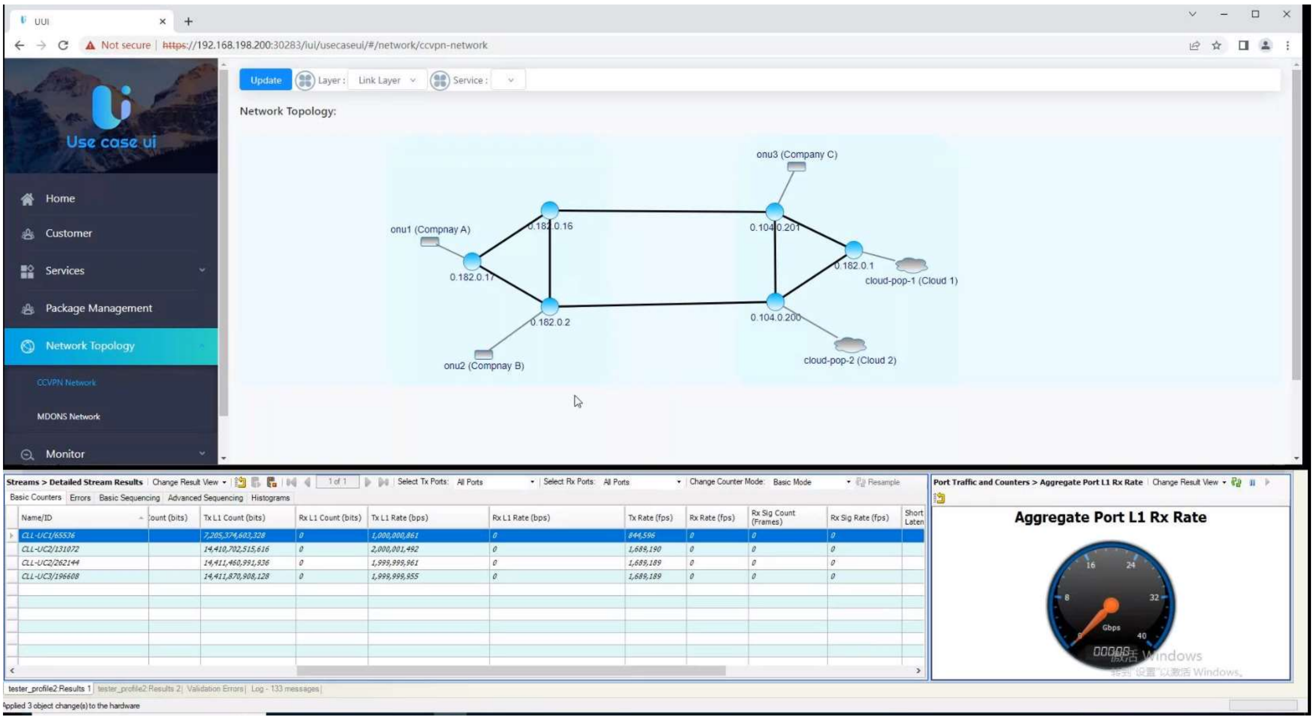

Figure 7: The initial condition of our demo setup

Fig. 14 below is a screenshot of ONAP's UUI portal where we can create Intent-Based Services. The user's input will be typed into the box shown in "red" and the "submit" button will be clicked, after which user's intent will be transformed into a so-called "intent form" shown in Fig. 15.

For our first test-case, the user's input is: "I need a service from Company A to Cloud one, with bandwidth of 1Gbps".

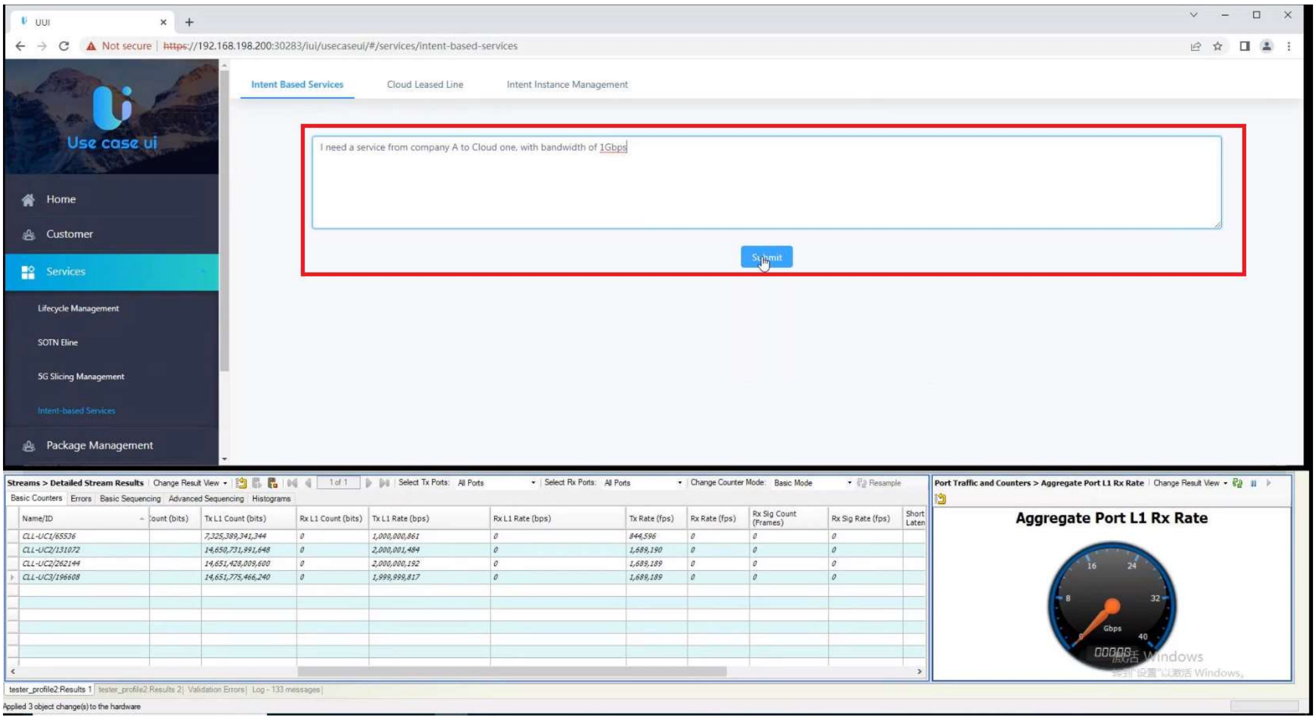

Figure 8: ONAP's UUI portal where user inputs will be typed

The intent form is basically a data structure that captures the essential attributes of the intent information.

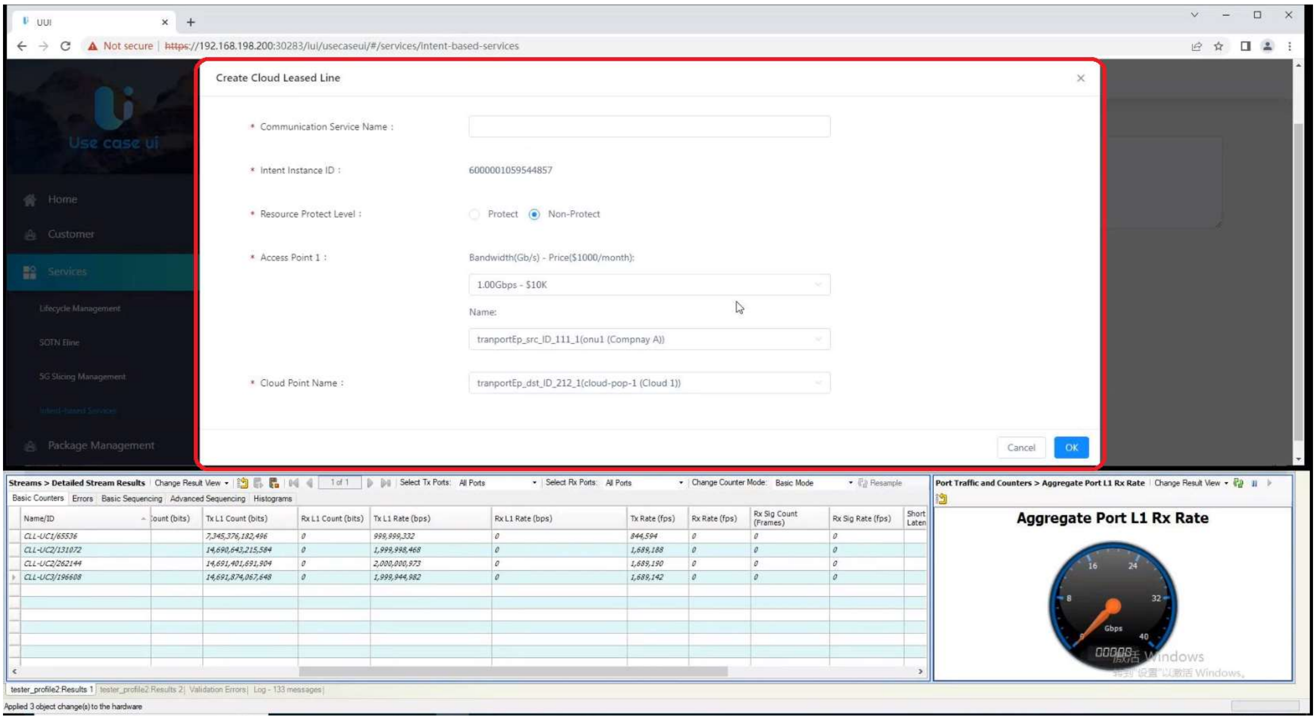

Figure 9: Intent form

Each specified bandwidth has a price-tag which is to show that the user has options to enter an intent-negotiation phase to select a desired bandwidth. If the user is satisfied with the intent attributes, he/she will click on "ok" and at this point our full automation process starts.

After the above step, we can switch to the “Network topology” tab to verify our service at the network configuration level. As can be seen below in Fig. 16, the yellow line shows our intended service (i.e. CLL) from Company A to Cloud one. And at the data-path level, on the bottom of the page we can see that the Rx Counter of our first traffic stream starts to show the received packets which is similar to the Tx counter confirming that there is no traffic loss.

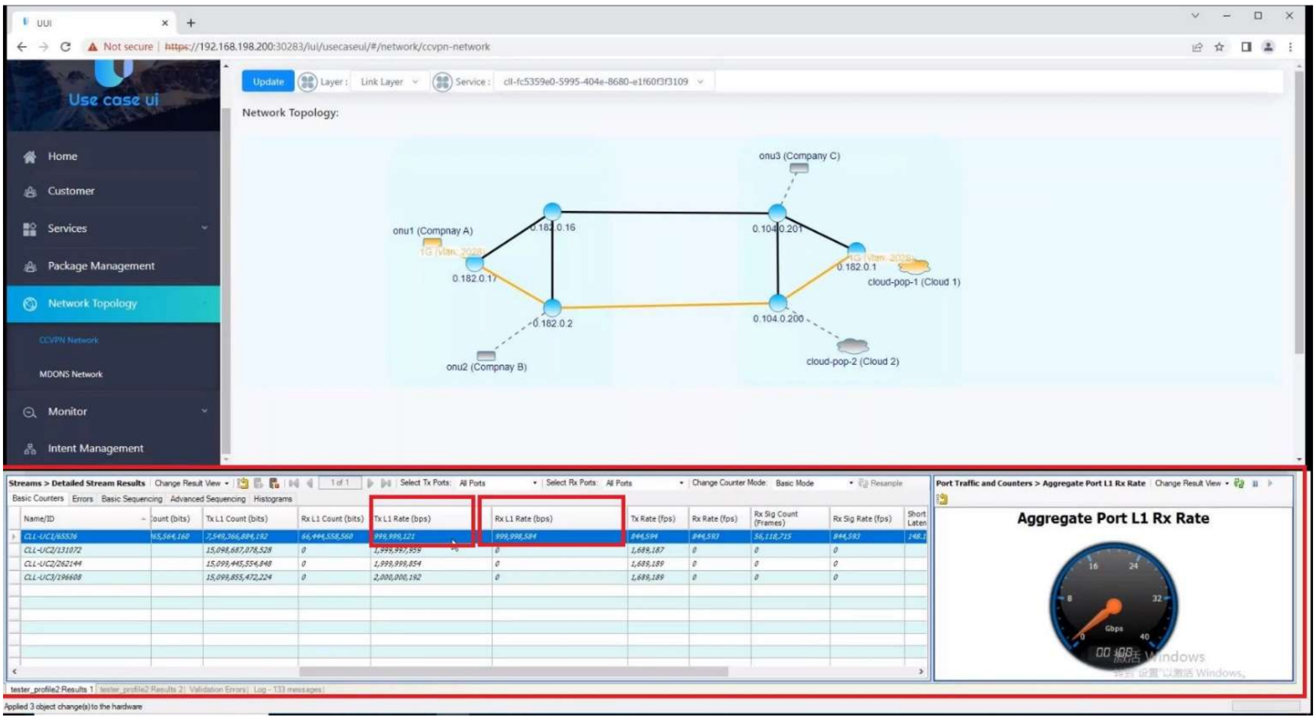

Figure 10: Service creation confirmation at the network configuration level

In order to better explain our full automation process and to know what exactly happens in the background once user clicks on the “ok” button in the intent form, we need to go over the call flow.

First step is to create an OTN tunnel in each domain. This will be initiated by ONAP and will be handled by each domain’s SDN controller. Fig. 17 below shows a sample JSON payload to create an OTN tunnel in one domain.

POST {{domain20}}/restconf/data/ietf-te:te/tunnels

Params | Authorization | Headers (9) | **Body** | Pre-request Script | Tests | Settings

none | form-data | x-www-form-urlencoded | raw | binary | GraphQL | JSON


```
{
    "ietf-te:tunnel": [
        {
            "source": "0.182.0.2",
            "destination": "0.0.0.0",
            "encoding": "ietf-te-types:lsp-encoding-oduk",
            "name": "4bdbfd92-9eb0-4cfd-8562-d351754b1fdb",
            "restoration": {
                "enable": "false",
                "restoration-reversion-disable": "true",
                "restoration-type": "ietf-te-types:lsp-restoration-restore-any",
                "wait-to-revert": 600
            },
            "te-bandwidth": {
                "ietf-otn-tunnel:odu-type": "ietf-otn-types:prot-ODUFlex-gfp"
            },
            "switching-type": "ietf-te-types:switching-otn",
            "te-topology-identifier": {
                "client-id": 6666,
                "provider-id": 5555,
                "topology-id": "11"
            },
            "p2p-primary-paths": {
                "p2p-primary-path": [
                    {
                        "name": "primary-path",
                        "path-scope": "ietf-te-types:path-scope-segment",
                        "explicit-route-objects": {
                            "route-object-include-exclude": [
                                {
                                    "index": "0",
                                    "explicit-route-usage": "ietf-te-types:route-include-ero",
                                    "unnumbered-hop": {
                                        "hop-type": "LOOSE",
                                        "node-id": "0.182.0.16",
                                        "link-tp-id": 117374978,
                                        "ietf-otn-tunnel:otu-port-type": "ietf-otn-types:port-otu2"
                                    }
                                },
                                {
                                    "index": "1",
```

```
                    "explicit-route-usage": "ietf-te-types:route-include-ero",
                    "label-hop": {
                      "te-label": {
                        "ietf-otn-tunnel:tpn": 1
                      }
                    }
                  }
                ]
              },
              "optimizations": {
                "optimization-metric": [
                  {
                    "metric-type": "ietf-te-types:path-metric-distance"
                  }
                ]
              }
            }
          ]
        },
        "ietf-otn-tunnel:dst-tributary-slot-count": 2,
        "ietf-otn-tunnel:src-tributary-slot-count": 2,
        "provisioning-state": "ietf-te-types:tunnel-admin-state-down"
      }
    ]
}
```

Figure 11: Sample OTN tunnel creation payload based on IETF ACTN YANG models

What happens in the background is the payload gets generated by ONAP and specifically by its SDNC module. Fig. 18 is a screenshot from SDNC logs confirming that a correct payload has been generated.

```
roject.version}, md5sum=5746db3bfb09691b1ef9e96f20e0847e] - 103 (execute)  | Parameter templateFileName: [/opt/onap/sdnc/restapi/templates/p2p-otn-tunnel-dst-domain.json]
2022-12-01T19:31:52,090 | INFO  | qtp1007696028-47510 | RestapiCallNode          | 283 - .onap.ccsdk.sli.plugins.restapi-call-node-provider - 1.5.2 | SvcLogicGraph [module=GENERIC-RESOURCE-API, rpc=p2p-vnf-topology-operation-p2p-activate, mode=sync, version=${p
roject.version}, md5sum=5746db3bfb09691b1ef9e96f20e0847e] - 103 (execute)  | Parameter restapiUrl: [http://123.60.230.198:32925/restconf/data/ietf-te:te/tunnels]
2022-12-01T19:31:52,090 | INFO  | qtp1007696028-47510 | RestapiCallNode          | 283 - .onap.ccsdk.sli.plugins.restapi-call-node-provider - 1.5.2 | SvcLogicGraph [module=GENERIC-RESOURCE-API, rpc=p2p-vnf-topology-operation-p2p-activate, mode=sync, version=${p
roject.version}, md5sum=5746db3bfb09691b1ef9e96f20e0847e] - 103 (execute)  | Parameter restapiUser: [canada_onap]
2022-12-01T19:31:52,090 | INFO  | qtp1007696028-47510 | RestapiCallNode          | 283 - .onap.ccsdk.sli.plugins.restapi-call-node-provider - 1.5.2 | SvcLogicGraph [module=GENERIC-RESOURCE-API, rpc=p2p-vnf-topology-operation-p2p-activate, mode=sync, version=${p
roject.version}, md5sum=5746db3bfb09691b1ef9e96f20e0847e] - 103 (execute)  | Parameter restapiPassword: [**********]
2022-12-01T19:31:52,090 | INFO  | qtp1007696028-47510 | RestapiCallNode          | 283 - .onap.ccsdk.sli.plugins.restapi-call-node-provider - 1.5.2 | SvcLogicGraph [module=GENERIC-RESOURCE-API, rpc=p2p-vnf-topology-operation-p2p-activate, mode=sync, version=${p
roject.version}, md5sum=5746db3bfb09691b1ef9e96f20e0847e] - 103 (execute)  | Parameter format: [json]
2022-12-01T19:31:52,090 | INFO  | qtp1007696028-47510 | RestapiCallNode          | 283 - .onap.ccsdk.sli.plugins.restapi-call-node-provider - 1.5.2 | SvcLogicGraph [module=GENERIC-RESOURCE-API, rpc=p2p-vnf-topology-operation-p2p-activate, mode=sync, version=${p
roject.version}, md5sum=5746db3bfb09691b1ef9e96f20e0847e] - 103 (execute)  | Parameter httpMethod: [post]
2022-12-01T19:31:52,091 | INFO  | qtp1007696028-47510 | RestapiCallNode          | 283 - .onap.ccsdk.sli.plugins.restapi-call-node-provider - 1.5.2 | SvcLogicGraph [module=GENERIC-RESOURCE-API, rpc=p2p-vnf-topology-operation-p2p-activate, mode=sync, version=${p
roject.version}, md5sum=5746db3bfb09691b1ef9e96f20e0847e] - 103 (execute)  | Parameter responsePrefix: [otn-oof]
2022-12-01T19:31:52,091 | INFO  | qtp1007696028-47510 | RestapiCallNode          | 283 - .onap.ccsdk.sli.plugins.restapi-call-node-provider - 1.5.2 | SvcLogicGraph [module=GENERIC-RESOURCE-API, rpc=p2p-vnf-topology-operation-p2p-activate, mode=sync, version=${p
roject.version}, md5sum=5746db3bfb09691b1ef9e96f20e0847e] - 103 (execute)  | Building JSON started
2022-12-01T19:31:52,092 | INFO  | qtp1007696028-47510 | RestapiCallNode          | 283 - .onap.ccsdk.sli.plugins.restapi-call-node-provider - 1.5.2 | SvcLogicGraph [module=GENERIC-RESOURCE-API, rpc=p2p-vnf-topology-operation-p2p-activate, mode=sync, version=${p
roject.version}, md5sum=5746db3bfb09691b1ef9e96f20e0847e] - 103 (execute)  | Building JSON completed. Time: 1
2022-12-01T19:31:52,104 | INFO  | qtp1007696028-47510 | RestapiCallNode          | 283 - .onap.ccsdk.sli.plugins.restapi-call-node-provider - 1.5.2 | SvcLogicGraph [module=GENERIC-RESOURCE-API, rpc=p2p-vnf-topology-operation-p2p-activate, mode=sync, version=${p
roject.version}, md5sum=5746db3bfb09691b1ef9e96f20e0847e] - 103 (execute)  | {   "ietf-te:tunnel": [     {       "source": "0.182.0.1",       "destination": "0.0.0.0",       "encoding": "ietf-te-types:lsp-encoding-oduk",       "name": "1c2eaeea-ecdf-4887-afd6-b98df91646bd",
    "restoration": {       "enable": "false",       "restoration-reversion-disable": "true",       "restoration-type": "ietf-te-types:lsp-restoration-restore-any",       "wait-to-revert": 600     },     "te-bandwidth": {       "ietf-otn-tunnel:odu-type": "ietf-otn-type
s:prot-ODUFlex-gfp"     },     "switching-type": "ietf-te-types:switching-otn",     "te-topology-identifier": {       "client-id": 6666,       "provider-id": 5555,       "topology-id": "11"     },     "p2p-primary-paths": {       "p2p-primary-path": [         {
   "name": "primary-path",           "path-scope": "ietf-te-types:path-scope-segment",           "explicit-route-objects": {             "route-object-include-exclude": [               {                 "index": "0",                 "explicit-route-usage": "ietf-te-typ
es:route-include-ero",                 "unnumbered-hop": {                   "hop-type": "LOOSE",                   "node-id": "0.104.0.201",                   "link-tp-id": "218038273",                   "ietf-otn-tunnel:otu-port-type": "ietf-otn-types:port-otu2"
 }               },               {                 "index": "1",                 "explicit-route-usage": "ietf-te-types:route-include-ero",                 "label-hop": {                   "te-label": {                     "ietf-otn-tunnel:tpn": "3"                   }
     }                 }               ]             },           "optimizations": {             "optimization-metric": [               {                 "metric-type": "ietf-te-types:path-metric-distance"               }             ]           }         }       ]     },
  "ietf-otn-tunnel:dst-tributary-slot-count": 2,     "ietf-otn-tunnel:src-tributary-slot-count": 2,     "provisioning-state": "ietf-te-types:tunnel-admin-state-down"     }   ] }
2022-12-01T19:31:53,541 | INFO  | qtp1007696028-47510 | RestapiCallNode          | 283 - .onap.ccsdk.sli.plugins.restapi-call-node-provider - 1.5.2 | SvcLogicGraph [module=GENERIC-RESOURCE-API, rpc=p2p-vnf-topology-operation-p2p-activate, mode=sync, version=${p
roject.version}, md5sum=5746db3bfb09691b1ef9e96f20e0847e] - 103 (execute)  | Response received. Time: 1437
2022-12-01T19:31:53,541 | INFO  | qtp1007696028-47510 | RestapiCallNode          | 283 - .onap.ccsdk.sli.plugins.restapi-call-node-provider - 1.5.2 | SvcLogicGraph [module=GENERIC-RESOURCE-API, rpc=p2p-vnf-topology-operation-p2p-activate, mode=sync, version=${p
roject.version}, md5sum=5746db3bfb09691b1ef9e96f20e0847e] - 103 (execute)  | HTTP response code: 201
```

Figure 12: OTN Tunnel creation payload generated by ONAP's SDNC

After creating an OTN tunnel in each domain, an Ethernet Service needs to be created for each domain. Fig. 19 below shows a sample JSON payload to create an Ethernet Service based on IETF ACTN Yang models.

POST {{domain20}}/restconf/data/ietf-eth-tran-service:etht-svc...

Params | Authorization | Headers (9) | Body | Pre-request Script | Tests | Settings

none | form-data | x-www-form-urlencoded | raw | binary | GraphQL | JSON


```json

{
    "ietf-eth-tran-service:etht-svc-instances": [
        {
            "etht-svc-name": "ebce9b56-15be-4eca-9558-323391c37934",
            "etht-svc-title": "actn-eoo008",
            "etht-svc-type": "ietf-eth-tran-types:p2p-svc",
            "te-topology-identifier": {
                "provider-id": 5555,
                "client-id": 6666,
                "topology-id": "33"
            },
            "underlay": {
                "otn-tunnels": [
                    {
                        "name": "4bdbfd92-9eb0-4cfd-8562-d351754b1fdb",
                        "encoding": "ietf-te-types:lsp-encoding-oduk",
                        "switching-type": "ietf-te-types:switching-otn"
                    }
                ]
            },
            "resilience": {
                "protection": {
                    "enable": "true",
                    "protection-reversion-disable": "true",
                    "protection-type": "ietf-te-types:lsp-protection-unprotected",
                    "wait-to-revert": 0
                }
            },
            "admin-status": "ietf-te-types:tunnel-admin-state-up",
            "etht-svc-end-points": [
                {
                    "etht-svc-end-point-name": "0",
                    "etht-svc-access-points": [
                        {
                            "access-point-id": "0",
                            "access-node-id": "0.182.0.2",
```

```
                    "access-ltp-id": 33488897,
                    "access-point-type": "ietf-eth-tran-types:optical-ge"
                }
            ],
            "service-classification-type": "ietf-eth-tran-types:vlan-classification",
            "outer-tag": {
                "tag-type": "ietf-eth-tran-types:classify-c-vlan",
                "vlan-value": 2028
            },
            "ingress-egress-bandwidth-profile": {
                "bandwidth-profile-type": "ietf-eth-tran-types:mef-10-bwp",
                "CIR": "1000000",
                "EIR": "0"
            }
        }
      ]
    }
  ]
}
```


Figure 13: Sample Ethernet Service creation payload based on IETF ACTN YANG models

Fig. 20 is a screenshot from SDNC logs confirming that the Ethernet Service payload has been generated properly by ONAP.

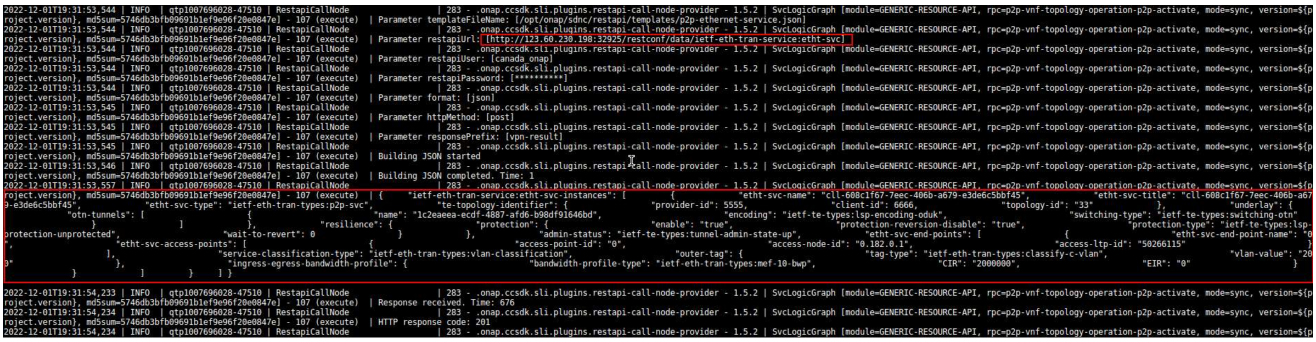

Figure 20: SDNC logs confirming the generated payload for Ethernet Service creation

Same steps will happen for the second domain as well i.e. OTN tunnel and Ethernet Service get created for the 2nd domain.

Once the service is up and running, its performance monitoring (PM) data will be periodically pulled by ONAP through SDNC and this data will be sent to ONAP's DCAE and Policy for further analysis and if any modification is required, then this will be done through SO.

4.2.3 Overall summary of the demo

ONAP's Intent manager module resumes the Intent Owner's role, as defined in ZSM 011. It receives and translates user intents into intent data models and sends them to the service orchestrator, which is the Intent Handler. It converts the intent data models into interfaces that comply with the IETF ACTN standards, invokes the ACTN interfaces of the vendor's domain controller to implement E2E automatic provisioning of cloud access services, and implements closed-loop management of user intents for on-demand and flexible bandwidth adjustment based on the telemetry data provided by the domain controllers. For the leased line service creation, both symmetric VLAN and asymmetric VLAN scenarios have been tested.

5 Additional information

The instructions for setting up the ONAP SDNC and the physical network controller simulator for this POC are available in the following ONAP wiki pages:

- https://wiki.onap.org/display/DW/Cloud+Leased+Line+%28CLL%29+Configuration+and+Operation+Guidance
- https://wiki.onap.org/display/DW/ACTN+Simulator+User+Guide

6 PoC Suggested Action Items

The interaction mechanism between the intent user and the ONAP intent-based system (IBS), which is developed by this PoC, needs to be further enhanced. Intent interactions may consist of two phases. The first is a negotiation of the fulfilment of the user's intent. It implies the communication of a desired intent from the user to IBS and response from IBS regarding its ability to fulfil. This phase cannot really be dispensed with, as it is always possible that a requested service intent cannot be provided at a point in time. Yet, the PoC only demonstrated the case where the network resources and conditions are always adequate to accommodate the user requests.

The second phase is intent assurance/maintenance, which is the automation closed-loop trying to maintain the service state against the intent-negotiated requirements. However, this phase may require a mechanism to deliver unavoidable exception communication from IBN to the intent user, which can trigger consequent action by the user, re-negotiation of intent, or both. This mechanism also needs to be further studied and developed.

References

[1] ETSI GS ZSM 002, “Zero-touch network and Service Management (ZSM); Reference Architecture”, v1.1.1, August 2019 [Online]. Available: https://www.etsi.org/deliver/etsi_gs/ZSM/001_099/002/01.01.01_60/gs_ZSM002v010101p.pdf

[2] ETSI GS ZSM 008, “Zero-touch network and Service Management (ZSM); Cross-domain E2E service lifecycle management”, v1.1.1, July 2022 [Online]. Available: https://www.etsi.org/deliver/etsi_gs/ZSM/001_099/008/01.01.01_60/gs_ZSM008v010101p.pdf