Evaluation of the abstraction of optical topology models in Blockchain-based data center interconnection

Pol Alemany[1,*], Ricard Vilalta[1], Raul Muñoz[1], Ramon Casellas[1], and Ricardo Martínez[1]

[1] *Centre Tecnològic de Telecomunicacions de Catalunya (CTTC/CERCA), Castelldefels (Barcelona), 08860 Spain*
[*] *Corresponding author: pol.alemany@cttc.cat*

Compiled January 10, 2022

Data Center (DC) interconnection allows to have optical transmissions between DCs directly connected to optical networks, avoiding the use of a packet-based infrastructure. Thanks to the use of next generation pluggable coherent optics, it is possible to create Connectivity Services (CSs) across multiple optical transport domains. In this Multi-domain CSs scenario, cloud operators and transport operators have to work together in the most dynamically way possible. To do so, they need a common place (i.e., a market) where the transport operators may expose their available optical resources and the cloud operators request (e.g., rent) them to be used in order to create End-to-End (E2E) CSs between DCs. Having multiple transport operators exposing their resources information in a common place requires a set of common rules (i.e., how much of the topology to show) to create E2E CSs requested between cloud operators.

This paper makes use of the Blockchain technology to present a Blockchain-based extension for the Software-Defined Network (SDN) architecture to allow each optical transport operator domain to become a peer in a Blockchain network. In there, each peer follows the same rules and shares the same exact level of topology information by using a specific abstraction model to map the optical domain resources. This article uses a set of three different abstraction models to validate their behaviour on a Blockchain system when managing multiple domain resources and the deployment of CSs across these domains. To do so, an experimental comparison on how the different abstraction models affect the performance of the Blockchain system is presented.

http://dx.doi.org/10.1364/ao.XX.XXXXXX

1. INTRODUCTION

Data Center (DC) interconnection aims to allow a new architectural model by giving the possibility to have third party cloud domains connected between them across flexi-grid Wavelength Division Multiplexing (WDM) networks of one or several telecommunications operators, avoiding packet-based aggregation networks. Using next generation plugabble coherent optics on their router devices, cloud operators can request a Connectivity Service (CS) between DCs with a specific spectrum slot (e.g., 50GHz) instead of an amount of Bandwidth (BW) (e.g., 100 or 200 GB/s). With this direct relationship and compared to the traditional "contract between players" model, a new and more dynamic model might be necessary.

In this model, transport operators put a set of their available resources in a common place (i.e., a market), where cloud operators can request some of these resources to deploy CSs between DCs across the transport operators domains. In order to make all the involved players equal, this market should define a set of conditions or rules such as no cloud or transport operator being able to access the resources of another one. To bring this equality at inter-domain level, the right option is to define a multi-operator peer-to-peer (p2p) solution using Distributed Ledger Technologies (DLT) such as Blockchain together with the use of Smart Contracts (SCs).

Blockchain is a geographically distributed database organised as a p2p system in which any data needs to pass a consensus procedure to be accepted by the majority of the peers. All accepted data is stored in all the nodes, so the system is transparent and no peer should be able to get advantage over another without the majority knowing it. The stored data in the Blockchain is structured in blocks (e.g., a block is a set of data transactions between peers) and each new block is linked to the previous one, making the system to be tamper-proof. Despite all its advantages, Blockchain has some drawbacks like scalability. For this reason, taking into account that transport domains may contain a massive amount of information, it becomes interesting to study how a transport resources market may be implemented using Blockchain. To study this aspect, abstraction models are required.

Abstraction models [1] allow to reduce the amount of information that an element in a control plane layer sends to the element on the upper layer as done in Software-Defined Network (SDN) control infrastructures. By doing so, the element on top has an overview of the domain network topology below instead of all the details, while keeping the capability to request CSs over the physical infrastructure.

This paper extends the work presented in [2] in which a Blockchain-based module was presented allowing an SDN domain to become part of a multi-domain scenario to create E2E CSs (i.e., a set of Domain CSs). As novelties, this paper presents how the E2E CS created has an ensured spectrum continuity alongside all its Domain CSs and more importantly, it provides the comparison of using blockchain services on top of three network abstraction models (i.e., transparent, Virtual Link and Virtual Node) and the abstracted information management within the Blockchain-based market.

This paper is organized as follows. Section 2 presents a state of the art on the use of Blockchain and abstraction models on optical scenarios. Section 3 presents the architecture of the market with the elements involved: the optical data model used, the new module for a domain to become a peer, the abstraction models and the Routing and Spectrum Assignment (RSA) used. Section 4 describes how Blockchain is used to manage multi-domain SDN connections with the SC designed and the processes to distribute the abstracted optical domain information and the creation of an E2E CSs. Section 5 presents the experimental results by comparing the use of the different optical topology abstraction models. Finally, Section 6 presents the conclusions and future work.

2. RELATED WORK

The optical network resource management has been and it is still an issue being researched essentially due to: a) the complexity of having an efficient resources usage; b) the transport domains massive amount of information (e.g., nodes, links, node ports); and c) the fulfillment of optical requirements such as spectrum continuity.

Multiple works have studied the previous issues focusing for example on the use of abstraction models to solve them. Đerić et al. [3] evaluate how the amount of information used in different abstraction models may affect the available hardware resources (e.g., CPU) in an SDN hypervisor. In their study, they use two abstraction models: transparent and Virtual Node (VNode), also referred to as Big Switch. Licciardello et al. [4] compare the Big Switch and the weighted Virtual Link (VLink) abstraction models in a DC Interconnection network scenario. While these two works presented interesting results, in [3] only two abstraction models were used omitting the VLink model and in [4], the authors assumed that all CSs deployed had wavelength continuity. Casellas et al. [5] used the capability to abstract network resources to manage disaggregated optical networks resources. The work done in [5] has been used as a reference to carry out the development of the infrastructure and of the results later described.

In addition, Blockchain is being studied to be used in different management aspects of optical domains. Ding et al. [6] presented an algorithm to trade spectrum resources between Elastic Virtual Optical Networks (EVONs) using Blockchain. While their results are quite interesting and present a way to have a fair spectrum trade between EVONs, they do not describe the cost of distributing the network resources information in the Blockchain and their usage. Yang et al. [7] proposed a distributed Blockchain-based trusted multi-domain collaboration for mobile edge computing called BlockTC. In there, all SDN controllers share their topology information with the other domains, so they can verify if the routes selected are legitimate. In their work, they only use the transparent abstraction model (i.e., the complete topology), and so no comparison with other possible abstracted models is done to validate if the routes generated might still be legitimate while using less information. Derhab et al. [8] described a security architecture that integrates SDN and Blockchain technologies in order to improve the security among intercontroller communications using a reputation mechanism that classified the controllers using two possible historical-based actions strategies. Similarly, Gorla et al. [9] present an architecture based on Blockchain specifically dedicated to be used on edge domains as it allows different Mobile Network Operators to share their spectrum in order to achieve the most efficient use of their resources. While in [8] no optical aspects are checked and in [9] their focus is on spectrum resources in edge domains, our work aims to fulfill both aspects: the use of a Blockchain-based architecture to manage spectrum resources on optical transport domains. Fernando et al. [10] designed an architecture similar to ours focused on security aspects such as the messages integration between controllers, the detection of malicious hosts and the SLA enforcement managing Bandwidth (BW) resources. In their work no optical aspects are taken into account as BW is used instead of spectrum to create the E2E CSs between domains. Finally, Fichera et al. [11] designed and developed a Blockchain architecture for optical multi-domain scenarios similar to the one presented in this article, but with the objective of keeping the data record associated to multiple SDN events to validate their accomplishment, and in case of failure, to register which component is the responsible one. These last works did not check how the amount of information affects the Blockchain. This aspect is studied in our work by using the different abstraction models.

To the best of our knowledge, no other work focused on the scenario previously introduced with a market place in which transport operators may share their optical resources and make them public for cloud operators to request E2E CSs. Moreover, we present how the three selected abstraction models (i.e., transparent, VLink and VNode) affect the designed Blockchain-based SDN architecture [2] in terms of time delay and Blockchain data processing costs when E2E CSs deployments are requested.

3. BLOCKCHAIN-BASED SDN ARCHITECTURE

When managing a Blockchain-based SDN architecture an important aspect to consider is the amount of information required to manage the network resources exposed by each domain controller to the upper domain controller. The more details a controller has about its below domain, the more precise its network control and management becomes but, in parallel, it also takes longer due to the need of processing the higher amount of information. This trade-off (i.e., detailed management vs. processing time) together with the fact that a network is composed by multiple transport domains raised the necessity to use abstraction models to control the available resources.

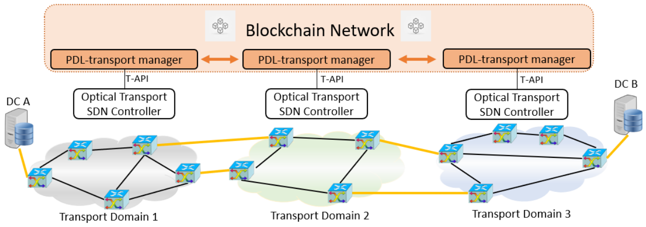

Fig. 1. Blockchain-based SDN Controller Architecture with intra-domain links (black) and IDLs (yellow).

To make the SDN architecture of our optical transport domains become a Blockchain-based SDN architecture, a new module called Permissioned Distributed Ledger (PDL)-transport Manager was designed. This new module allows an SDN Controller to become part of a collaborative system in which a set of optical transport network domains share their internal resources in a collaborative methodology without needing an E2E SDN Controller on top managing the complete E2E transport network resources. As illustrated in Fig.1, the extension adds a new architecture layer placed on top of each optical SDN Controller: PDL-transport Manager, Optical Transport SDN Controller and Optical Transport Networks. The communication between the PDL-transport Manager and the underlying Transport SDN Controller in the south-bound interface is done through the use of the Transport -Application Programming Interface (T-API) data model [12]. On the north-bound interface, REST requests are defined to get, distribute and manage the network information and the E2E CSs deployment and terminate actions.

A. Modelling Optical SDN domain networks

The process to abstract the topology of an optical network domain is a complex procedure due to the massive amount of information required to define all the nodes, links and the spectrum information for each of the previous elements. There are multiple data models (i.e., YANG, NETCONF, etc.) available to define the context and topology of an optical network domain, but one of the most used is the T-API data model defined by the Open Networking Foundation (ONF). An example of its implementation can be found in the T-API-enabled Transport SDN controller described in [13]. T-API allows to define SDN control plane functions to a set of service interfaces. T-API bases its data model on a set of elements that allows to abstract the existing physical resources information into a single data object and to apply actions over the physical resources based on the abstracted information. One of the benefits of using T-API is the use of REST commands and the fact that it allows to constantly abstract the topology from one layer to another layer above it as many times as required through a relationship between a T-API provider (SDN Controller) and an T-API Client (Application, Orchestrator or a parent SDN Controller). By using the T-API v2.1.3 [14] photonic media model it is possible to manage the connectivity, topology and path computation services. So, the WDM is modelled as a (single layer) T-API forwarding domain (FD) and the Media Channel (MC) protocol qualifier is covered within the "PHOTONIC_MEDIA" layer (within the context data object).

Before presenting the essential elements of T-API, it is important to understand that, when dealing with abstraction processes, the concept of node has to be understood as an entity that may be a single physical network element (i.e., router, switch, bridge, etc.) belonging to a domain but it may also be a complete domain infrastructure, for example when dealing with other domains. As illustrated in Fig.2, the network resources in the optical transport domain (top left) may be abstracted with all the physical nodes in detail (top right) or the whole optical transport domain may become a single node (bottom right) in the resulted abstraction process.

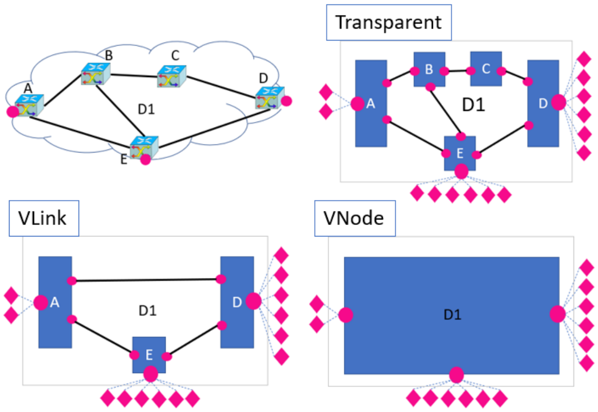

Fig. 2. Original and abstraction model topologies.

Regarding the essential T-API elements, the most basic abstraction possible is the definition of a context with a list of Service Interface Points (SIPs). As described in [14], a context is an abstraction allowing logical isolation or grouping of other abstracted network resources, and a SIP (pink diamonds in Fig.2) is the logical representation of the external view of any port placed in the edge of a node. So, with only a list of SIPs in the context, it is possible to request and configure CSs between two edge points in the domain without the need to know the details of the network resources infrastructure associated to the context domain. Having the list of SIPs enables a T-API client to request MC CSs between endpoints, optionally specifying optical spectrum to be provisioned. To support this possibility, each SIP element within the context is augmented with specific MC resource availability, referred to as MC pool. The MC pool encompasses information about the supportable, available and occupied frequency slots.

A context may contain a more detailed definition of the real network infrastructure. To do so, the use of Node Edge Points (NEPs), Nodes, Links and Topology becomes essential. A NEP (pink circles in Fig.2) is the representation of each physical port available in a real network element. A NEP may be considered internal (i.e., no SIPs associated) or external. If a NEP is internal, it contains the information regarding the available, supportable and occupied spectrum (i.e., MC pool). On the other hand, if a NEP is external, one or more SIPs may be associated to allow multiple CSs over the same physical port and the spectrum information is specific for each SIP. A Node (blue boxes in Fig.2) is the abstract representation of a set of network resources (i.e., a single physical network element or a set) and it essentially aggregates a set of NEPs. Moreover, a Link (black lines in Fig.2), is the representation of the association between two or more NEPS. Finally, a Topology defines the abstracted topological characteristics from a set of network resources. To sum up, a Topology is composed by a set of Nodes and Links, a Node is a set of NEPs and a Link is the association of at least two NEPs.

Once the context with the desired abstracted information is ready, it is possible to create CSs. From the T-API point of view, a CS is the representation of the interconnection request between two SIPS. As a result, a Node Connection element is configured between each pair of NEPs belonging to the same node for each of the nodes involved in the route between the source and the destination SIPs. So, as previously presented, the creation of a CS only needs two different SIPs. However, more complex requests are possible. For example, in the case presented in this article, the use of optical network resources required to specify the spectrum slot and optionally the internal links for each requested CS in order to ensure the spectrum continuity alongside all the optical domains involved. One last aspect to consider on how the CSs were used in this article is the differentiation between E2E CS and Domain CS: an E2E CS is a composition of multiple Domain CSs. Once the Domain CSs are provisioned, each domain topology with any of the Domain CSs is updated with the corresponding Connection Endpoints (CEPs), the elements that define T-API Connection elements composing each Domain CS.

Finally, together with each domain context, the PDL-transport managers need another information to complete the E2E vision of the whole physical infrastructure. The links that allow the interconnection among optical domains are called Inter-Domain Links (IDLs). The IDLs (yellow lines in Fig.1) data objects do not follow the T-API data model as they do not have SIPs, but they make use of the NEPs identifiers to join multiple domains.

B. PDL-transport Manager

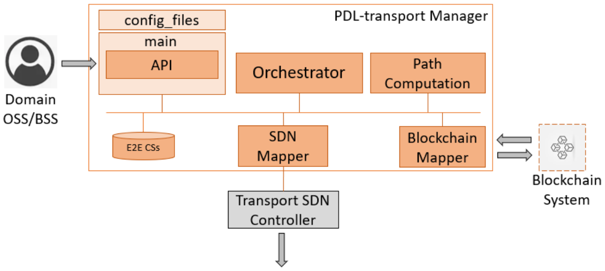

Fig. 3. PDL-transport Manager internal architecture.

The PDL-transport Manager internal architecture is presented in Fig.3 and its main components are: a) the main contains the API with all the possible actions to be requested and the configuration parameters (e.g., abstraction model used); b) the Orchestrator takes care of managing the requested actions coming from the API (e.g., E2E CS deployment); c) the Path Computation manages the local graph with the E2E multi-domain topology to generate the possible routes for each E2E CS; d) the E2E CSs is the database where the deployed E2E CS are stored; and e) the SDN and the Blockchain Mappers are in charge of generating the T-API Domain CS requests depending if they have to be requested to the local (SDN Mapper) or to another domain (Blockchain Mapper) SDN Controller.

This new module is in charge of the following functionalities:

- Context abstraction: Once the PDL-transport Manager is launched and depending on the configured abstraction model, the PDL-transport Manager obtains the local optical domain context and processes it in order to generate the new (transparent, VLink or VNode) abstracted context.
- Context resources distribution: The abstracted context is split in SIPs, NEPS, Nodes, Links and Context metadata (i.e., uuid, name, list of SIPs uuids, topology) and distributed in the Blockchain. This division was done to avoid the distribution of the complete context once a resource (i.e., NEP, SIP) is selected in an E2E CS, so only the specific resource updated information is distributed in a new transaction.
- IDLs distribution: In addition to the context, each PDL-transport Manager is in charge of distributing the information of the physical links that interconnect the domains among them. Before distributing this information, a PDL-transport Manager checks if an IDL has already been distributed, and in that case, the IDL is omitted. This means that the last domain to join does not distribute anything as all IDLs are already shared. The IDLs are not abstracted as they are not part of a single optical SDN domain but they are a shared element between two optical SDN domains.
- E2E CS management: Deploying and terminating the requests to connect or disconnect two SIPs from two different optical transport domains. During the deployment process, there are some important actions such as the path computation, the spectrum continuity enforcement and checking and updating of the resources availability information in the Blockchain.

C. Abstraction Models

Fig.2 illustrates the three selected abstraction models that have been implemented and compared in the experimental section.

The first abstraction model is the so-called "transparent" model. As its own name shows, there is a transparent abstraction procedure which means that the complete T-API context coming from the local SDN Controller is not processed and it is simply distributed to the PDL-transport Managers from the other peer domains. As presented in Fig.2, all the nodes, NEPS (pink circles), SIPs (pink diamonds) and links in the physical infrastructure (top left) are selected and the abstraction resulting (top-right) is an exact copy of the real resources.

Secondly, the VNode abstraction model processes the original network topology (Fig.2 top left) to generate a new T-API context in which the whole domain becomes a single node (Fig.2 bottom-right). In this abstraction model, all the real nodes and links information is omitted. Instead, the new abstracted T-API context contains the list of SIPs (mandatory in all three abstraction models) and the topology has a single node with the NEPs and their associated SIPs. So, the internal domain infrastructure is not distributed to all the other peer domains in the Blockchain.

Finally, the VLink model. The basic idea behind this model is to select a specific group of the real nodes and define a new set of "virtual" links that interconnect the selected nodes among them. The condition used in this article to select the nodes was to keep those that had NEPs with associated SIPs. As illustrated in Fig.2, the abstracted T-API context (bottom left) had only three nodes and three virtual links when originally (top right) they were five and six respectively. This abstraction model has a peculiarity compared to the other abstraction models in order to equalize the three of them in terms of routing path computation costs. While in the Transparent and VNode all the links have a cost equal to 1 (i.e., hooping), in the virtual links created when using the VLink model, each one had a weight equally proportional to the number of hoops in the real physical infrastructure. So, using the domain abstraction example illustrated in Fig.2, the link between the nodes A and D have a weight of 3, which is equal to the number of hoops in the transparent model domain.

D. Routing and Spectrum Assignment

As previously presented, all the PDL-transport Manager peers have the complete E2E vision of the infrastructure. So, each PDL-transport Manager peer is in charge of the Routing and Spectrum Assignment (RSA) process for any E2E CS request received.

Regarding the routing phase, all the route possibilities between the source and destination nodes are generated but only the 20 shortest paths among them are selected. Then, the PDL-transport Manager validates if the first route in the list is feasible. Instead, if the PDL-transport Manager discovers that there are no resources available on the selected route, it searches for the next viable route within the routes list until it finds one that fulfills the requirements.

The requirement used to decide if a route is feasible or not is the spectrum continuity among all the SIPs and NEPs involved in the selected route. To ensure it, the PDL-transport Manager managing the requested E2E CS gets the selected NEPs (only for the transparent and VLink models) and the SIPs involved in the route from the Blockchain and takes their available spectrum slots. Among all the available slots, a list with the common available slots is created. Finally, to select the slot among all the options, the PDL-transport Manager applies first an Exact-Fit policy and, if it does not exist, it applies a Best-Fit policy with the closest spectrum slot to the requested capacity. Finally, with the common spectrum continuity selected, the E2E CS can be created with the composition of all the required Domain CSs.

4. USING BLOCKCHAIN TO MANAGE SDN CONNECTIONS

This section presents the SC designed together with the optical domains resources distribution and the E2E CS deployment procedures.

A. Designed Smart Contract

Among the Blockchain technologies available, the selected one had to be composed only by peers fulfilling a set of requirements to write and validate data. Moreover, the Blockchain system also needed the capacity to have some autonomy to trigger some processes when data could be written. The solution for this last requirement are the SCs.

A SC is a program with a few set of functions known by all the Blockchain peers. A SC only runs under certain conditions and it is used to make procedures more automatic, specially when a transaction is requested. In the context of this work, the functionalities of the designed SC are:

- to distribute and store SDN information from each abstracted domain (e.g., T-API SIPs and T-API topology), together with the IDLs data between the SDN domains.

- to automate the multiple and specific requests generation to deploy Domain CSs. When a PDL-transport Manager has selected the best route based on the spectrum resources available, it distributes a transaction that generates an event with the specific Domain CS information and the associated Blockchain address of the peer owning those resources. Then, all the peers receive the event but only the peer with the specific Blockchain address will take the event information and process it to generate the Domain CS through a T-API CS request sent to the SDN Controller below.

- to limit the rights to apply actions over other domains physical resources. Only the peer that has requested a Domain CS (as part of an E2E CS) is able to request its termination aside the Domain CS owner itself, which can terminate it locally through its local SDN Controller.

B. A collaborative E2E Topology

Before any E2E CS may be requested, it is necessary that all domains distribute their local abstracted context to the other peer domains, so each of them may create the local graph with the E2E infrastructure view as presented in Fig. 4. Using Fig.1 as a reference, there are two sets of information to be distributed by each optical SDN domain (called PDL-SDN peer in Fig.4): a) the abstracted context and, b) the set of known IDLs. Once these two information sets are distributed, each PDL-Transport Manager is able to update the vision of the whole E2E physical infrastructure.

The workflow (Fig.4) has the following steps:

- Step 1 - The Domain Operations Support System/Business Support System (OSS/BSS) passes the IDLs information (previously defined between optical domain operators) and requests to distribute all this information.

- Step 2 - Based on the IDL information, the local E2E network topology view (e.g., a graph element) is updated with the IDL data by adding to the graph only the new nodes and links.

- Steps 3/4/5 - A transaction to distribute both sets of data (i.e., the IDLs and the local abstracted SDN Context (based on the T-API data model)) is generated. Once the transaction is done, all the other PDL-SDN peers (i.e., domains) receive an event of a "NEW DOMAIN".

- Steps 6/7/8 - The event reception is confirmed by all the PDL-SDN peers.

- Step 9 - The Domain OSS/BSS is informed about the correct distribution.

- Steps 10/11/12 - All the PDL-SDN peers update their local graph with the new SDN and IDL information.

Once this procedure is done, each single PDL-SDN peer knows about the existence of all the other PDL-SDN peers that together compose an E2E optical infrastructure with the distributed abstracted T-API contexts information and the associated IDLs. With all this distributed information, one final aspect to take into consideration is the possibility of data leakage to a peer that should not be part of the Blockchain system. To avoid this possible situation, the use of a permissioned (i.e., private) blockchain becomes necessary as only known peers are involved in the p2p system as the access to it, it is not public. Moreover, the use of the predefined data models (i.e., T-API), defines the same information to be shared by all the peers, making equal among them.

C. Blockchain-based E2E CS requests management

Once the multiple PDL-SDN peers domain contexts and the IDLs information are distributed, the whole E2E transport topology is ready to be used to deploy and terminate E2E CSs. The process to deploy an E2E CS is divided in four main actions as presented in Fig.5:

1. Action 1 - E2E CS request & data object creation

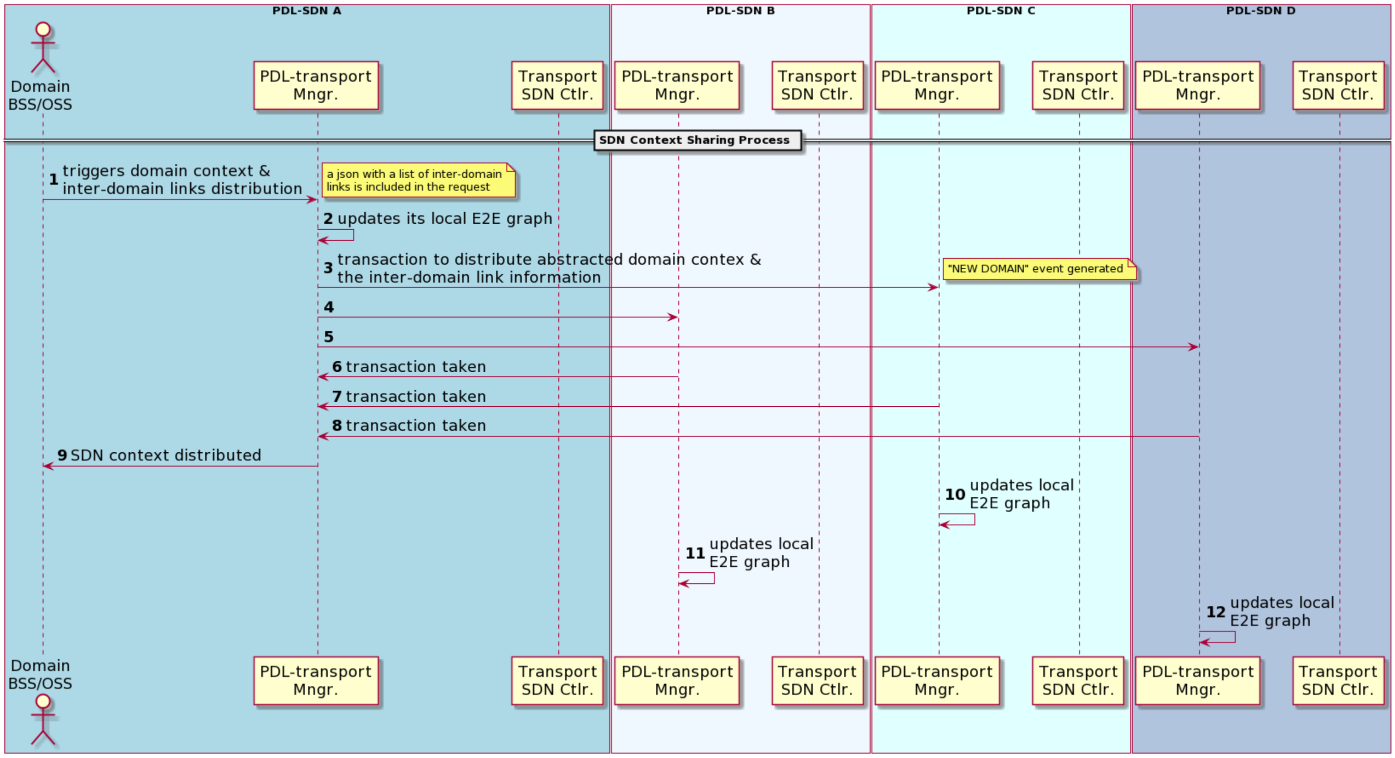
PDL-SDN A
PDL-SDN B
PDL-SDN C
PDL-SDN D
Domain BSS/OSS
PDL-transport Mngr.
Transport SDN Ctlr.
SDN Context Sharing Process
1 triggers domain context & inter-domain links distribution
a json with a list of inter-domain links is included in the request
2 updates its local E2E graph
3 transaction to distribute abstracted domain contex & the inter-domain link information
"NEW DOMAIN" event generated
4
5
6 transaction taken
7 transaction taken
8 transaction taken
9 SDN context distributed
10 updates local E2E graph
11 updates local E2E graph
12 updates local E2E graph

Fig. 4. SDN context and IDLs distribution.

- Step 1 - The Domain OSS/BSS (i.e., Cloud Operator) requests an E2E CS defining the source and destination (domain, node and SIP identifiers) and the desired capacity (e.g., as other values are accepted, GHz is the expected).
- Step 2 - The PDL-Transport Manager does a first check on the SIP availability by checking if they are already used. If so, the Domain OSS/BSS is informed about it and the E2E CS deployment finished. Otherwise, the E2E CS data object is created.

2. Action 2 - Path Computation (Routing & Spectrum Assignments)

- Step 3 - The PDL-transport Manager checking for a set of possible routes between the source and destination nodes using the local E2E graph.
- Step 4 - The shortest one is selected and the nodes information are mapped to the SIPs and internal NEPs information.
- Step 5 - With the chosen SIPs and internal NEPs, the PDL-transport Manager checks if they all have a common spectrum available with the requested capacity.
- Step 6/7 - If a common spectrum is found, a set of Domain CSs is defined to be deployed to compose the E2E CS. Otherwise, steps 4 and 5 are started again to search for the following route possibility, always from shortest to the longest one.

3. Action 3 - Connectivity Services Creation

- Step 8 - The PDL-transport Manager reads the list of Domain CSs and if they belong to the local PDL-SDN peer, the Domain CS request is sent to the Transport SDN Controller (Transport SDN Ctlr. in Fig.5) below.
- Steps 9/10 - The local Transport SDN Controller deploys the Domain CS and confirms it back to the PDL-transport Manager.
- Step 11/12/13 - If the Domain CS belongs to another PDL-SDN peer, a Blockchain transaction with the Domain CS information (i.e., domain SIPs, capacity and, when using the transparent abstraction model, the internal links) is generated and distributed to all the other peers.
- Step 14/15 - Only the owner of those resources takes the transaction, maps it and forwards it to its local Transport SDN Ctrl. below to deploy the Domain CS.
- Step 16 - At the same time of step 14 and 15, the PDL-transport Manager confirms that the transaction has been taken on.
- Step 17 - Once the Domain CS is ready, the Transport SDN Ctrl. informs its upper PDL-transport Manager which generates a new Blockchain transaction.
- Steps 18/19/20 - The new Blockchain transaction with the updated Domain CS information is distributed.
- Step 21 - Only the E2E CS owner processes the transaction with the updated Domain CS information and confirms its reception.

4. Action 4 - E2E CS Confirmation

- Step 22 - The E2E CS data object information is updated with the latest Domain CS status (i.e., ready).

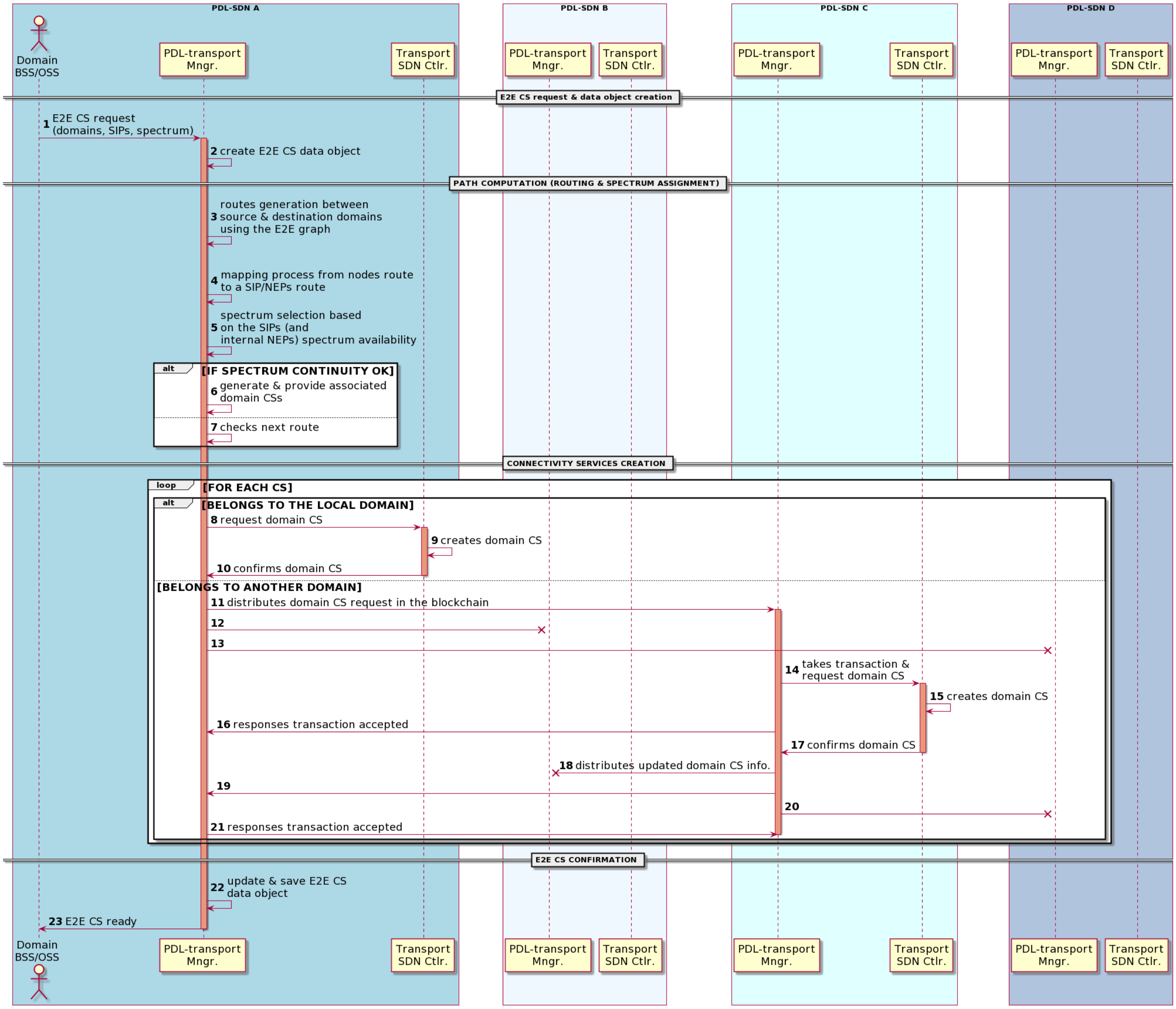
PDL-SDN A
PDL-SDN B
PDL-SDN C
PDL-SDN D
Domain BSS/OSS
PDL-transport Mngr.
Transport SDN Ctlr.
E2E CS request & data object creation
1 E2E CS request (domains, SIPs, spectrum)
2 create E2E CS data object
PATH COMPUTATION (ROUTING & SPECTRUM ASSIGNMENT)
3 routes generation between source & destination domains using the E2E graph
4 mapping process from nodes route to a SIP/NEPs route
5 spectrum selection based on the SIPs (and internal NEPs) spectrum availability
alt
[IF SPECTRUM CONTINUITY OK]
6 generate & provide associated domain CSs
7 checks next route
CONNECTIVITY SERVICES CREATION
loop
[FOR EACH CS]
alt
[BELONGS TO THE LOCAL DOMAIN]
8 request domain CS
9 creates domain CS
10 confirms domain CS
[BELONGS TO ANOTHER DOMAIN]
11 distributes domain CS request in the blockchain
12
13
14 takes transaction & request domain CS
15 creates domain CS
16 responses transaction accepted
17 confirms domain CS
18 distributes updated domain CS info.
19
20
21 responses transaction accepted
E2E CS CONFIRMATION
22 update & save E2E CS data object
23 E2E CS ready
Domain BSS/OSS
PDL-transport Mngr.
Transport SDN Ctlr.

Fig. 5. E2E Network Slice deployment.

- Step 23 - The Domain OSS/BSS is informed about the requested E2E CS being deployed and available to be used.

Finally, the process to terminate an E2E CS is not illustrated because it follows a simpler procedure. Once a terminate request is received, the PDL-transport Manager checks the multiple Domain CSs information composing the E2E CS and generates a set of Blockchain transactions (one per Domain CS) with the necessary information to terminate them (i.e., Domain CS identifier and PDL-SDN peer address owner). Once the Domain CSs are terminated and the E2E CS owner is informed about it, the resources availability is updated and distributed to the Blockchain for future E2E CSs.

5. EXPERIMENTAL VALIDATION

This section presents the designed use case and the resulting abstractions, then it describes the implemented architecture and, finally, the results obtained are discussed.

A. Use Case Description

In order to experimentally validate the presented Blockchain-based SDN architecture (i.e., the SDN Controller with the PDL-transport Manager module), a network of four optical domains was designed and each one of them is managed by its own SDN Controller. Each optical domain had a different number of nodes and none of their internal context and topologies was equal to the others. Fig. 6 shows the complete E2E network topology resulting of each abstraction procedure. On the top right, there is the E2E transparent abstracted topology which is exactly as the originally defined E2E topology (top left). On the bottom left, the VLink E2E abstracted topology and finally, on the bottom right, the VNode E2E abstracted topology.

The interesting point of our use case is that, as previously explained, each PDL-transport Manager has a vision of the whole E2E topology but it cannot directly request Domain CSs except

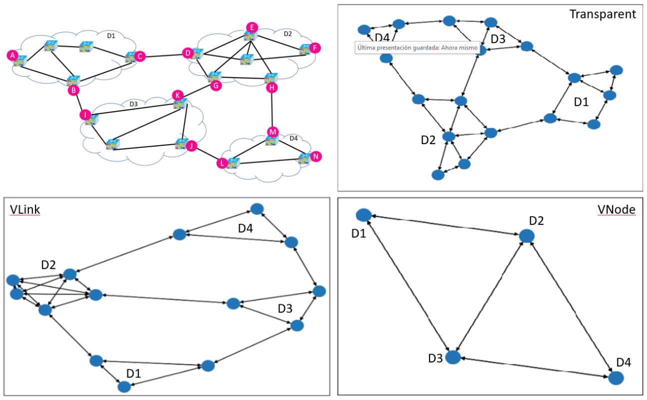

Fig. 6. Graphs representation of the original and the abstracted network topologies.

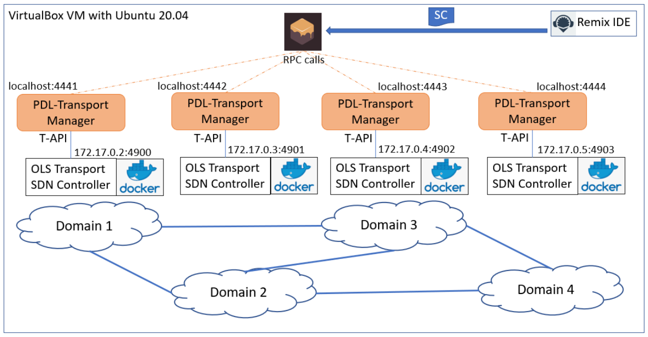

Fig. 7. Testbed architecture.

to its local SDN Controller below. Regarding the Domain CS in other domains, it must distribute them to all the Blockchain peers and the right domain owner will take the newly arrived event and create the desired Domain CS. The only common network resources in the three abstraction cases are the IDLs (i.e., Domain 1 to Domain 2, Domain 1 to Domain 3, etc.) among the optical domains.

B. Testbed Architecture

An environment (Fig. 7) was created to properly create a set of four different optical domains, each with its PDL-Transport manager on top, as introduced in section 5A.

At the top of everything there is a Blockchain network created using Ganache [15]. Ganache is an Ethereum-based [16] emulator that allows an easy and fast configuration and creation of a Blockchain network. It also allows to define multiple aspects such as the number of peers, the amount of Ether (i.e., Ethereum's cryptocurrency) per peer and others. The reasons to select Ganache as part of the testbed environment were: a) it allowed to build a Blockchain system able to accept and manage SCs without the need of applying long configuration actions in only few seconds, b) it allowed to have the complete environment, together with the rest of the described use case (i.e., optical SDN Controllers) and finally, c) Ganache has the capability to keep logs and a list of the Blockchain transactions, allowing a smooth PDL-transport Manager module development, integration and testing with the rest of the environment.

The use of Ganache means that the Blockchain selected in the experimental phase was a standard implementation of an Ethereum network. This implies that the consensus mechanism [17] used is the "de facto" Proof of Work (PoW) and the incentives for the Blockchain peers to participate in the network are the standard values defined in [18].

With the complete Blockchain and SDN optical domains environment ready, the next step is the creation and deployment of the SC. The SC was developed and deployed into the Blockchain using the Remix Integrated Development Environment (IDE) which is one of the sub-projects composing the Remix Project [19]. Remix IDE is an open source web and desktop application that makes the design and development process of SCs faster and lighter than other similar options. REMIX IDE was used to write the previously presented SC using the Solidity language (e.g., a high-level and object-oriented language). One of the most powerful tools from Remix IDE is the simplicity of deploying the written SCs to the Blockchain system. In order to communicate the PDL-transport Manager with the Blockchain, Remote Procedure Calls (RPC) were done using the python web3 library [20] which is specially dedicated to interact with Ethereum-based Blockchains.

Finally, an SDN-controlled disaggregated optical network simulator [21] was used to define and create the optical transport domains with an SDN-Controller on top ready to receive requests to either manage (Domain) CSs and to retrieve the SDN domain context using the T-API data model. Moreover, it is deployed using docker which ensures the isolation of the optical domains as if they would be physically deployed in different geographical locations.

C. Experimental Results

This subsection presents the results gathered from the multiple tests done using the different abstraction models applied on the designed network illustrated in Fig. 6. This subsection aims to compare each abstraction model performance used in terms of the time delay and the costs of managing the information volumes for each abstraction model in the Blockchain system. The results are presented using column graphs and each column is one of the three abstraction models; blue for the transparent, orange for the VLink and gray for the VNode.

Before discussing the results and in order to understand them better, it is important to be aware of the units used to study the costs results. Due to the fact of using an Ethereum-based Blockchain, each transaction generated requires an element called "Gas" to be processed. Gas [22] is referred to as the unit to measure the computational cost of a transaction in the Blockchain system. This cost depends on the amount of data within the transaction and the processes done with the SC action applied to the transacted data. So, the higher the amount of information and the number of code actions to be applied, the higher the cost will be.

Regarding the results associated to the E2E CS deployment time in Fig.8. The first aspect to take into consideration is the high values of the total time needed in all three abstraction models. Compared with the values presented in [2], the main reason for the increment in the deployment time is due to the times values of the second, third and fourth columns in Fig.8 which belong to: a) the time to get the IDLs information from the Blockchain and compose the data object, b) the time to compose the SDN Context data object and check if the selected resources are available and finally, c) the time to update the selected resources in the Blockchain. Checking the worst case (i.e., transparent model - blue columns), all these four time values give a total value of 384.405s which represents a 94.23% of the overall E2E CS deployment time, leaving 22.521s for the E2E CS data

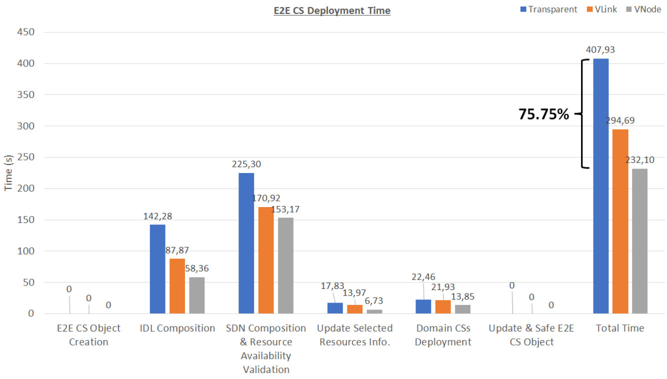

Fig. 8. E2E CS deployment time delay.

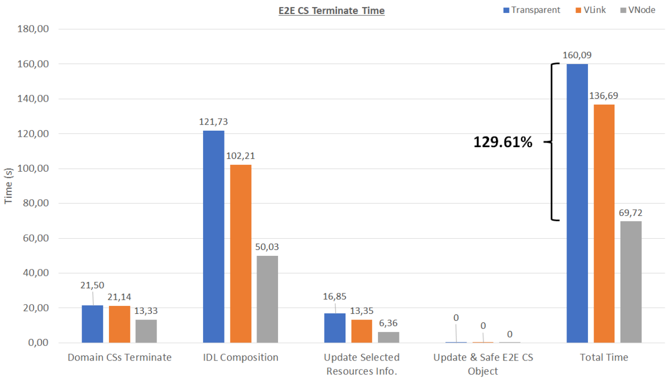

Fig. 9. E2E CS terminate time delay.

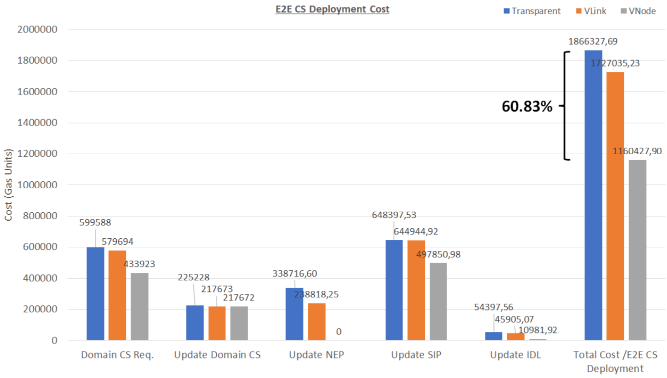

Fig. 10. E2E CS deployment cost.

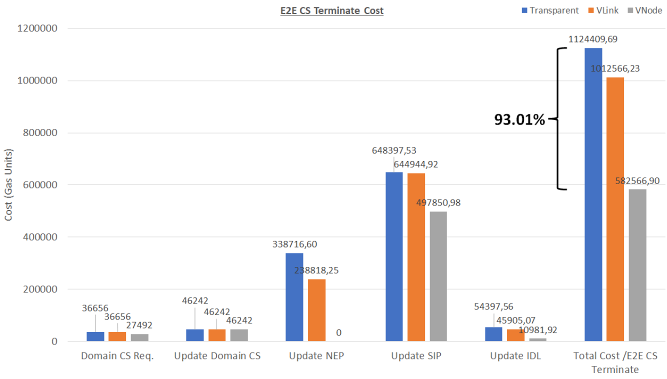

Fig. 11. E2E CS terminate cost.

object creation and update and the Domain CS deployment composing the E2E CS. A similar behaviour is illustrated in Fig.9 when terminating an E2E CS. The high values are due to the IDL composition and the update of the used (now free) resources in the Blockchain. In the E2E CS terminate case, there is no "SDN Composition" column as it was only necessary to update the specific SIPs and NEPs elements in the Blockchain, not like in the E2E CS deployment case where it was necessary to have all the SDN resources composed to find those that could ensure the spectrum continuity. When comparing the transparent abstraction model to deploy an E2E CS with the other two models, a time difference of 75.75% (Fig.8) is obtained compared with the VNode and a 38.42% compared with the VLink model. In the case of the terminate action, the difference becomes even bigger with a 129.61% (Fig.9) difference between the transparent and the VNode models and a 17.11% between the transparent and the VLink models.

From the costs point of view, the total mean value costs to complete an E2E CS deployment and terminate actions are presented in Figs.10 and 11, respectively. Both figures show the five procedure steps that generate transactions in the Blockchain. These steps are : a) requesting a Domain CS deployment or termination, b) updating the Domain CS data object once the action is done, c) updating the spectrum in the internal NEPs used, d) updating the spectrum in the SIPs used within the corresponding optical domain context and, finally, e) updating the IDLs information regarding the SIPs used. As illustrated in both figures, the step with the highest cost is the one associated to the SIPs update. This is due to the need of finding the specific element among all the SIP elements. But the main difference between the two procedures is the distribution cost of the Domain CS deployment and termination requests. While the Domain CS deployment request (first column in Fig.10) generates a transaction with multiple information parameters (i.e., uuid, SIPs, spectrum, etc.), the transaction generated for the Domain CS termination request (first column in Fig.11) only needs the Domain CS uuid. In both procedures, the results present the behaviour expected with the transparent model being more expensive than the other two models. For example, comparing the transparent abstraction model to deploy an E2E CS with the other two models, a cost difference of a 60.83% (Fig.10) is obtained compared with the VNode and a 8.06% compared with the VLink model. In the case of the terminate action, the difference is bigger with a 93.01% (Fig.11) difference between the transparent and the VNode models and an 11.04% between the transparent and the VLink models. In addition to the amount of information in the initial Domain CS requests, another aspect that influenced the total values presented is the non-usage of NEP resources when deploying/terminating E2E CS in the VNode model. As it can be seen in the third column of both figures, the cost is equal to 0.

6. CONCLUSIONS

Blockchain is a technology which is already present in multiple and different fields and its usage will be increased in the near future. Optical networks are not that different and aim to use Blockchain in order to improve multiple aspects such as their security, the control plane management and others. This article presented a new module to avoid the full management centralization of multiple operators optical SDN domains by proposing an extension of the common SDN architecture.

To accomplish this, the new module, called PDL-transport Manager, allowed an optical SDN Controller to become part of a Blockchain network in which each peer shared a set of optical resources to create E2E CSs across the different optical transport domains. Moreover, this article focused on the use of three abstraction models defining different levels of shared resources information and compared them to validate their performance in a Blockchain system. The SDN domain resources distributed in the Blockchain were structured using T-API data model as it allowed to easily define which information would be used for each abstraction model.

The obtained results for all three abstraction models show the expected relationship among them, with the transparent and the VNode models having the highest and lowest time delay and gas cost values respectively. According to these results, the best choice to apply the designed Blockchain-based SDN architecture to manage E2E CS in a collaborative way seems to be the VNode abstraction model. Despite choosing this model, which implies that many network resources details are hidden, it is still possible to create E2E CSs. Moreover, this model might bring more possibilities of improvement than the others, precisely due to the reduced amount of information it deals with. In order to have a fair comparison of all the three abstraction models, the complete abstracted T-API context (i.e., the SIPs and the topology with nodes, links and NEPs) was distributed and stored in the Blockchain. For the transparent and VLink models, this action is mandatory as these two models need to know these details. However, the VNode deployment time results may be improved by distributing and storing each domain context with only the SIPs list available (i.e., fulfilling the T-API data model definition). By doing so, the third column in Fig.8 would be reduced and so would the total deployment time.

Finally, future tasks are planned to check how to improve the previously described results, especially in terms of the time delay. To do so, one possibility the authors foresee are: a) to modify the characteristics of the Ethereum Blockchain used or use a different Blockchain solution with the same abstraction models to compare the new results with those presented in the current article and b) to reduce the T-API context information necessary to request the Domain CS that allow to compose the E2E CS.

FUNDING

Work partially funded by the Spanish MINECO AURORAS (RTI2018-099178-B-I00) and the EC H2020 TeraFlow (101015857) projects.

REFERENCES

1. M. Fiorani, A. Rostami, L. Wosinska, and P. Monti, “Abstraction models for optical 5g transport networks,” J. Opt. Commun. Netw. **8**, 656–665 (2016).
2. P. Alemany, R. Vilalta, R. Muñoz, R. Casellas, and R. Martínez, “Blockchain-based connectivity provisioning in multiple transport sdn domains,” in *2021 International Conference on Optical Network Design and Modeling (ONDM),* (2021), pp. 1–3.
3. N. Đerić, A. Varasteh, A. Basta, A. Blenk, and W. Kellerer, “Sdn hypervisors: How much does topology abstraction matter?” in *2018 14th International Conference on Network and Service Management (CNSM),* (2018), pp. 328–332.
4. M. Licciardello, M. Fiorani, M. Furdek, P. Monti, C. Raffaelli, and L. Wosinska, “Performance evaluation of abstraction models for orchestration of distributed data center networks,” in *2017 19th International Conference on Transparent Optical Networks (ICTON),* (2017), pp. 1–4.
5. R. Casellas, R. Martínez, R. Vilalta, and R. Muñoz, “Abstraction and control of multi-domain disaggregated optical networks with openroadm device models,” J. Light. Technol. **38**, 2606–2615 (2020).
6. S. Ding, G. Shen, K. X. Pan, S. K. Bose, Q. Zhang, and B. Mukherjee, “Blockchain-assisted spectrum trading between elastic virtual optical networks,” IEEE Netw. **34**, 205–211 (2020).
7. H. Yang, Y. Liang, J. Yuan, Q. Yao, A. Yu, and J. Zhang, “Distributed blockchain-based trusted multidomain collaboration for mobile edge computing in 5g and beyond,” IEEE Transactions on Ind. Informatics **16**, 7094–7104 (2020).
8. A. Derhab, M. Guerroumi, M. Belaoued, and O. Cheikhrouhou, “Bmc-sdn: Blockchain-based multicontroller architecture for secure software-defined networks,” Wirel. Commun. Mob. Comput. (2021).
9. P. Gorla, V. Chamola, V. Hassija, and N. Ansari, “Blockchain based framework for modeling and evaluating 5g spectrum sharing,” IEEE Netw. **35**, 229–235 (2021).
10. P. Fernando and J. Wei, “Blockchain-powered software defined network-enabled networking infrastructure for cloud management,” in *2020 IEEE 17th Annual Consumer Communications Networking Conference (CCNC),* (2020), pp. 1–6.
11. S. Fichera, A. Sgambelluri, F. Paolucci, A. Giorgetti, N. Sambo, P. Castoldi, and F. Cugini, “Blockchain-anchored disaggregated optical networks,” J. Light. Technol. **39**, 6357–6365 (2021).
12. V. Lopez, R. Vilalta, V. Uceda, A. Mayoral, R. Casellas, R. Martinez, R. Munoz, and J. P. Fernandez Palacios, “Transport api: A solution for sdn in carriers networks,” in *ECOC 2016; 42nd European Conference on Optical Communication,* (2016), pp. 1–3.
13. D. King and et al., “Rfc7491: A pce-based architecture for application-based network operations,” Internet Eng. Task Force (IETF) (2015).
14. Open Networking Foundation (ONF), “Tapi v2.1.3 reference implementation agreement,” Tech. rep., Open Networking Foundation, 1000 El Camino Real, Suite 100, Menlo Park, CA 94025 (2020).
15. ConsenSys Software Inc. 2021, “https://www.trufflesuite.com/ganache,” online. Accessed in November 3, 2021.
16. The Ethereum Foundation, “https://ethereum.org/en/,” online. Accessed in November 3, 2021.
17. The Ethereum Foundation, “https://ethereum.org/en/developers/docs/consensus-mechanisms/,” online. Accessed in Janury 9, 2022.
18. The Ethereum Foundation, “https://ethereum.org/en/developers/docs/consensus-mechanisms/pow/,” online. Accessed in Janury 9, 2022.
19. The Ethereum Foundation, “https://remix-project.org/,” online. Accessed in November 3, 2021.
20. P. Merriam and J. Carver, “https://web3py.readthedocs.io/en/stable/,” online. Accessed in November 3, 2021.
21. R. Casellas, F. J. Vílchez, L. Rodríguez, R. Vilalta, J. M. Fàbrega, R. Martínez, L. Nadal, M. S. Moreolo, and R. Muñoz, “An ols controller for hybrid fixed / flexi grid disaggregated networks with open interfaces,” in *Optical Fiber Communication Conference (OFC) 2020,* (Optical Society of America, 2020), p. M3K.2.
22. The Ethereum Foundation, “https://ethereum.org/en/developers/docs/gas/,” online. Accessed in November 3, 2021.