Open Disaggregated Optical Network Control with Network Management as Code

Javier Errea[1,5,*,§], Huy Tran Quang[1,6,§], Dominique Verchere[1], Huu Trung Thieu[2], Andrea Mazzini[3], Lahcen Abnaou[4], Jelena Pesic[4], Marina Curtol[4], Abdelali El Imadi[4], Adlen Ksentini[5], Djamal Zeghlache[6]

[1] Nokia Bell Labs, Route de Villejust, 91620 Nozay, France - [2]Nokia Bell Labs, Murray Hill, US
[3]Nokia, Vimercate, Italy - [4]Nokia, Nozay, France - [5]EURECOM Research Institute, Biot, France -
[6]Telecom SudParis, Institut Polytechnique de Paris, France

[*]javier.errea-moreno@nokia.com

Abstract: Network Management as Code is presented to manage the infrastructure and deployment of control functions in a declarative manner by adopting GitOps in open disaggregated optical networks. We introduce an Automation Engine that analyzes topology changes to trigger the Continuous Delivery procedure of SDN applications on a micro-service architecture.

1. Overview

The transition from traditional vendor-specific to disaggregated optical transport networks leads to the expansion of equipment inventory. Thus, operators need to have an unified mechanism to manage the topology and device information since various vendors use different approaches to report their network elements. Besides, open interfaces such as the Open Networking Foundation Transport API (ONF TAPI) [1], OpenConfig Working Group [2] allow network control functions to migrate from monolithic software stack on purpose-built hardware to more fine-grained microservice components deployed in the cloud infrastructure [3]. Therefore, softwarization plays an important role in Optical Network Automation [4]. Nevertheless, there exists a requirement on the specific versions of data models implemented in the functions, which hinders the interoperability between them. Adaptive and efficient software practices are needed to cope with the constant evolution of Open Optical Software Defined Networking (OO-SDN) and Virtualization technologies. In addition, both automated and manual deployment procedures are being applied which may cause a potential version mismatch if they are not tracked properly.

To alleviate these problems, we adopt GitOps [5] as an operating model for managing applications and their underlying infrastructure. GitOps builds on the concept of Infrastructure-as-Code (IaC), which is the process of managing and configuring cloud infrastructure under a definition file. Any changes to the file will be monitored by an IaC tool (i.e., Terraform [6]). Consequently, the state of the infrastructure is kept synchronized with the one described in the file. In addition, GitOps incorporates the functionality of Git repositories, merge requests (MRs) and Continuous Delivery (CD) to further unify software deployment and infrastructure operations. Any change to infrastructure is committed to the git repository along with application changes. GitOps leverages Git [7] as a single source of truth for both infrastructure and applications.

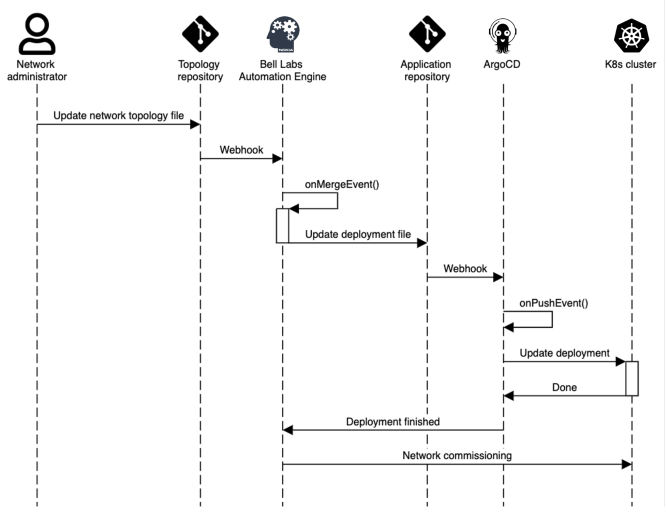
In this demonstration, we introduce Network Management as Code (NMaC) in open disaggregated optical networks by adopting GitOps techniques. The principles of NMaC are to automate i) optical topology management and ii) network control function deployment. Particularly, an Automation Engine (AE) is designed to analyze topology deviations and make decisions on the suitable control plane architecture. In combination with AE, Argo CD [8] acts as GitOps operator to form the NMaC tool, in order to trigger the CD procedure of SDN applications on a Kubernetes cluster. We validate this solution in the autonomous migration scenario from partially to fully disaggregated network. Thanks to the implementation of TAPI version 2.4, seamless multi-vendor network control is achieved.

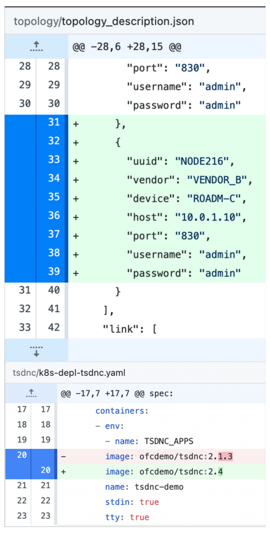
2. Innovation and OFC relevance

We showcase the complete autonomous management workflow of network infrastructure and control function deployment in open disaggregated optical networks using the Network Management as Code (NMaC) concept, as depicted in Fig. 1a. NMaC architecture consists of an Automation Engine (AE), a GitOps Continuous Delivery (CD) tool, a Docker registry, a Kubernetes cluster and two Git repositories. Topology and Application repositories represent the central source of truth of each NMaC principle. The network infrastructure is described in a topology definition file and stored in the Topology repository. On the other hand, deployment manifests are specified in the Application repository.

The AE acts as a reconciliation mechanism between the Topology and Application repositories. We configure the Topology repository with a webhook on merge events to notify AE of topology changes events automatically. When the network topology file is updated by a network administrator (see Fig. 1b), the AE analyzes the network

[§]These authors have equally contributed to this work.





topology/topology_description.json

```
@@ -28,6 +28,15 @@
          "port": "830",
          "username": "admin",
          "password": "admin"
+       },
+       {
+         "uuid": "NODE216",
+         "vendor": "VENDOR_B",
+         "device": "ROADM-C",
+         "host": "10.0.1.10",
+         "port": "830",
+         "username": "admin",
+         "password": "admin"
        }
      ],
      "link": [
```

tsdnc/k8s-depl-tsdnc.yaml

```
@@ -17,7 +17,7 @@ spec:
        containers:
        - env:
          - name: TSDNC_APPS
-         image: ofcdemo/tsdnc:2.1.3
+         image: ofcdemo/tsdnc:2.4
          name: tsdnc-demo
          stdin: true
          tty: true
```

Fig. 1. (a) NMaC Workflow. (b) Network Topology (top) and Application Deployment (bottom) file changes.

infrastructure and makes the decision to upgrade/downgrade the version of the SDN Controller (SDNC) application and to add/remove network control functions in the deployment file (see Fig. 1b). We use Argo CD as the GitOps CD tool to ensure the automatic deployment on the Kubernetes cluster. It synchronizes the desired state as declared in the repository with the actual state installed on the Kubernetes cluster. A webhook on push event is used to notify Argo CD when there is a new commit on the deployment manifest for faster reaction time. Then, it will automatically sync the deployment to match the desired state. In addition, Argo CD informs AE once the deployment is finished so that it can launch the network commissioning process.

We consider two Git operations for code updates in two repositories. In the Topology repository, the network administrator acts as a contributor and forks the shared project to their own one. They can submit the modified topology definition file and open a *Pull Request* (PR) to notify other maintainers of the change that they want to merge to the original repository. Involved contributors, such as technicians and operation engineers, can review and validate the PR. This operation provides a mechanism to safely verify the integrity of the network among the collaborators. Upon the approval, PR can be merged to the main repository and consequent actions are applied accordingly. On the other hand, the *Push* operation is applied for the Application repository since control functions deployment has been well studied by the AE and code approval procedure could be mitigated.

Additionally, we introduce for the first time a TAPI v2.4 implementation in the SDNC. TAPI v2.4 provides new features over previous releases (i.e., TAPI v2.1.3) such as equipment profiles, optical impairment parameters and event streaming. This enhancement enables the SDNC to manage directly Optical Impairment (OI) parameters in the topology, connectivity and path computation services without the need of intermediate Open Line System (OLS) controllers. Thus, a mapping between the OpenConfig based device configuration and TAPI v2.4 is implemented. On the Northbound, every control function interfaces with the SDNC using TAPI v2.4. The external OI-aware Path Computation Engine (PCE) application performs routing, modulation scheme and spectrum assignment on the fully disaggregated network based on the TAPI topology state. In addition, the database and Graphical User Interface (GUI) are upgraded to store and visualize the new TAPI v2.4 context. Also, the Kafka broker is used to stream TAPI event notifications between the SDNC and network applications.

3. Demo content and implementation

We propose the autonomous migration scenario (as depicted in Fig.2) from the partially to fully disaggregated network to demonstrate the applicability of NMaC architecture. The demonstration is performed live on our remote physical workbench of Nokia 1830 equipment. We consider two different OpenConfig versions implemented on devices to demonstrate a multi-vendor network scenario, notably Vendor-A and Vendor-B. In practice, this work is applicable to any OpenConfig-based solutions. In addition, control functions, such as GUI, OI-aware PCE, Kafka broker, SDNC and its database, are deployed on a Kubernetes cluster. The demo content is described as follows:

Scenario 1. Partially disaggregated network: Vendor-A OLS of two ROADMs (ROADM-A and ROADM-B) with a Vendor-A Transponder (TPR-A) and a Vendor-B Transponder (TPR-B) (Fig. 2a)

Step 1: Control Functions Deployment - Network devices and physical links are declared in the Topology Repository. AE examines the topology file and composes a deployment suite for this scenario. Argo CD is invoked and deploys

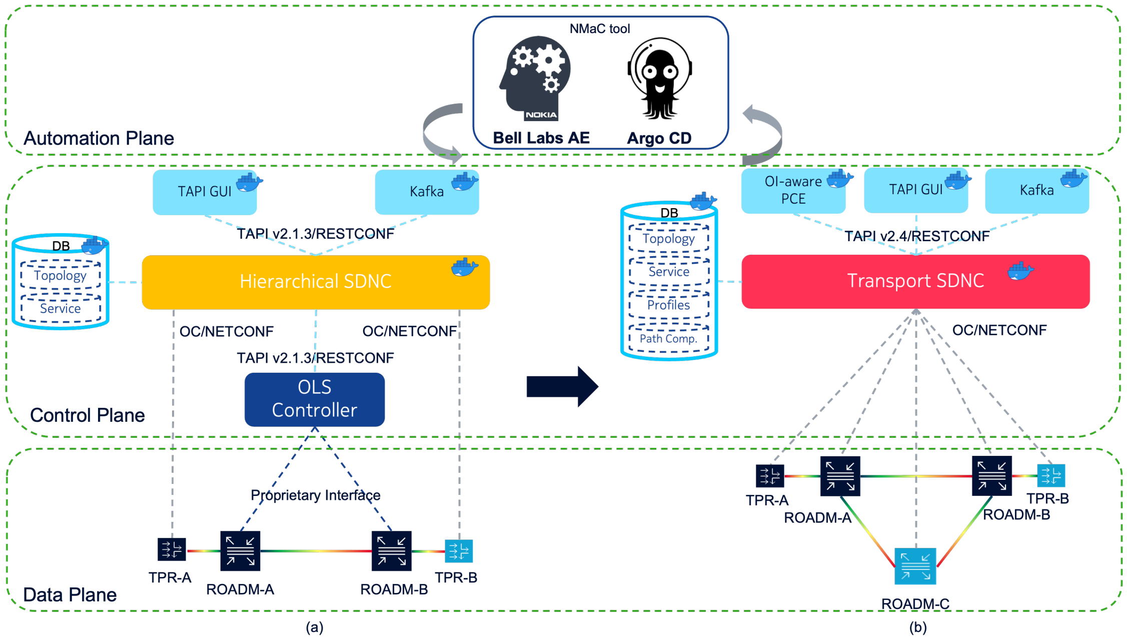


Fig. 2. Transition in Topology and Control plane from (a) Partially to (b) Fully Disaggregated Network managed by NMaC tool

the corresponding applications accordingly. A Hierarchical SDNC (H-SDNC) controls the transponders via OpenConfig/NETCONF and communicates with the OLS Controller via the TAPI v2.1.3 interface. Once completed, AE executes the network equipment commissioning to H-SDNC to populate the database with the TAPI context.
Step 2: TAPI Connectivity Service Creation - The H-SDNC exposes a TAPI v2.1.3 based interface on its NBI. Our TAPI GUI application discovers the TAPI context to visualize the TAPI topology. The OLS domain is abstracted as a single TAPI node. Via the dashboard, the TAPI connectivity service between TPR-A and TPR-B is requested with the DYNAMIC_RESTORATION protection scheme. H-SDNC configures both transponders and delegates the Network Media Channel (NMC) service establishment to the OLS controller, which communicates with the optical devices through a proprietary interface. Additionally, TAPI GUI subscribes to this connectivity service Kafka topic to receive state notifications during its life-cycle.

Scenario 2. Fully disaggregated network: adding ROADM-C from Vendor-B to the Vendor-A OLS (Fig. 2b)

Step 3: Migration from Partially to Fully Disaggregated Network - Upon merging a new pull request to add ROADM-C device information in the topology, the AE analyzes the changes and triggers the deployment of the corresponding applications. It swaps the H-SDNC with the Transport SDNC (T-SDNC), adds the OI-aware PCE application and removes the OLS controller application. Once the deployment has been finalized, Argo CD sends a confirmation to the AE in order to initiate the network commissioning process and update the Database information from TAPI v2.1.3 to TAPI v2.4. During this process, ROADM-A and ROADM-B management interface is switched to OpenConfig/NETCONF to be able to communicate with T-SDNC. Underlying services are not affected since the migration occurs on the control plane.
Step 4: Service Restoration - We audit the functionality of new components with a link failure event. One of the degree ports in the link between two Vendor-A ROADMs is shut down to simulate a fiber cut. The T-SDNC receives alarms, locates the source of failure and activates the dynamic restoration protection scheme at the NMC layer. It utilizes the OI-aware PCE application to compute a new path via the newly added ROADM-C.

References

1. ONF Transport API, [online] Available: https://github.com/OpenNetworkingFoundation/TAPI.
2. OpenConfig, [online] Available: https://github.com/openconfig/public.
3. Q. Pham-Van et. al. Demonstration of Container-Based Microservices SDN Control platform for Open Optical Networks. In *OFC 2019*.
4. G. Schembra et al. Guest Editors' Introduction: Special Section on Smart Management of Future Softwarized Networks. *IEEE Transactions on Network and Service Management*, 19(3):1942–1950, 2022.
5. Thomas Limoncelli. GitOps: A Path to More Self-service IT: IaC + PR = GitOps. *Queue*, 16:13–26, 06 2018.
6. Terraform, [online] Available: https://www.terraform.io/.
7. Scott Chacon and Ben Straub. *Pro git*. Apress, 2014.
8. ArgoCD, [online] Available: https://argoproj.github.io/cd/.