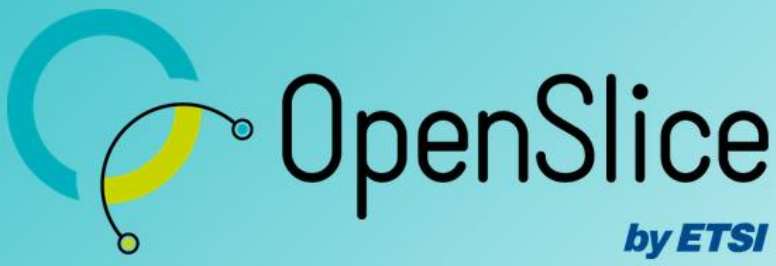
ETSI SDG OSL



OpenSlice 2024Q2

Release White Paper | 29 July 2024

This white paper provides an overview of the latest official release of OpenSlice, proudly brought to you by ETSI Software Development Group OpenSlice (SDG OSL). This marks ETSi SDG OSL first release under the ETSI umbrella, reflecting our commitment to excellence and innovation in the field of open-source Operations Support System (OSS) solutions.

"The latest OpenSlice 2024Q2 version is a manifest to our commitment to pave the way for modern telco-cloud requirements, seamless integration and reference implementations for 6G" - Christos Tranoris, Senior Research at UPATRAS and Chair of ETSI SDG OSL.

We want to keep the community's interest on par with our highest passion and expectation to revolutionize the way Network as a Service (NaaS) is delivered, and our latest release is a testament to our dedication! With this new release, we introduce significant changes aimed at enhancing user engagement and addressing the contemporary needs of both research and industry sectors on the matter.

Key Highlights of the Release 2024Q2

Here is a brief overview of some key highlights of this release

Cloud native support

OpenSlice can be deployed via docker compose and HELM Chart.

Native Kubernetes support

OpenSlice can now create and manage Custom Resources (CRs) and offer them as a Service. Also, it is able to deploy Services based on HELM Charts to target Kubernetes clusters.

Enhanced NFVO primitive support

OpenSlice can now automatically extract and display a list of available (VNF-level) primitives. It also offers a dedicated UI that enables seamless primitive execution.

Service characteristics' exposure revision

The transition of service characteristics between related services underwent a major revision to properly reflect and isolate customer and resource facing aspects.

Robustness and Performance enhancements

The latest release introduces several bugfixes, code testing extension, automated pipelines to validate the developed code and performance centric component enhancements.

Contents

Introduction

OpenSlice can be used in managing 5G network services from the user device to the core network and cloud as well as for Orchestrating cloud resources across private and public clouds for enterprise applications. OpenSlice is capable of supporting most of the features of an end-to-end (E2E) service orchestration framework while many of them will be more mature in future releases. The following figure displays the general usage of OpenSlice. The Figure 1 illustrates how OpenSlice supports the idea of an E2E network service orchestration framework by integrating multiple network components and layers, from user devices at the edge to radio, transport networks, core and public cloud services, ensuring seamless, secure, and efficient delivery of network services. Assuming that there are domain controllers for all the above domains OpenSlice can create the end-to-end service via the domain controllers by following the process of creating and deploying the end-to-end service by implementing transformations, and consuming APIs from various network entities. OpenSlice, offers user interfaces where users can interact with the system to expose, and manage service catalogs, services and resources that can be ordered by end users, following business logic and policies, exposed through the standardized TMFORUM APIs[1].

What we do

The ETSI Software Development Group for OpenSlice is developing an open source service based Operations Support System to deliver Network Slice as a Service following specifications from major SDOs including ETSI, TM Forum and GSMA

The group liaises with relevant standards bodies and projects working on network transformation such as the TM Forum, ETSI ZSM, ETSI NFV, OpenSourceMANO and TeraFlowSDN.

See more at: https://osl.etsi.org/

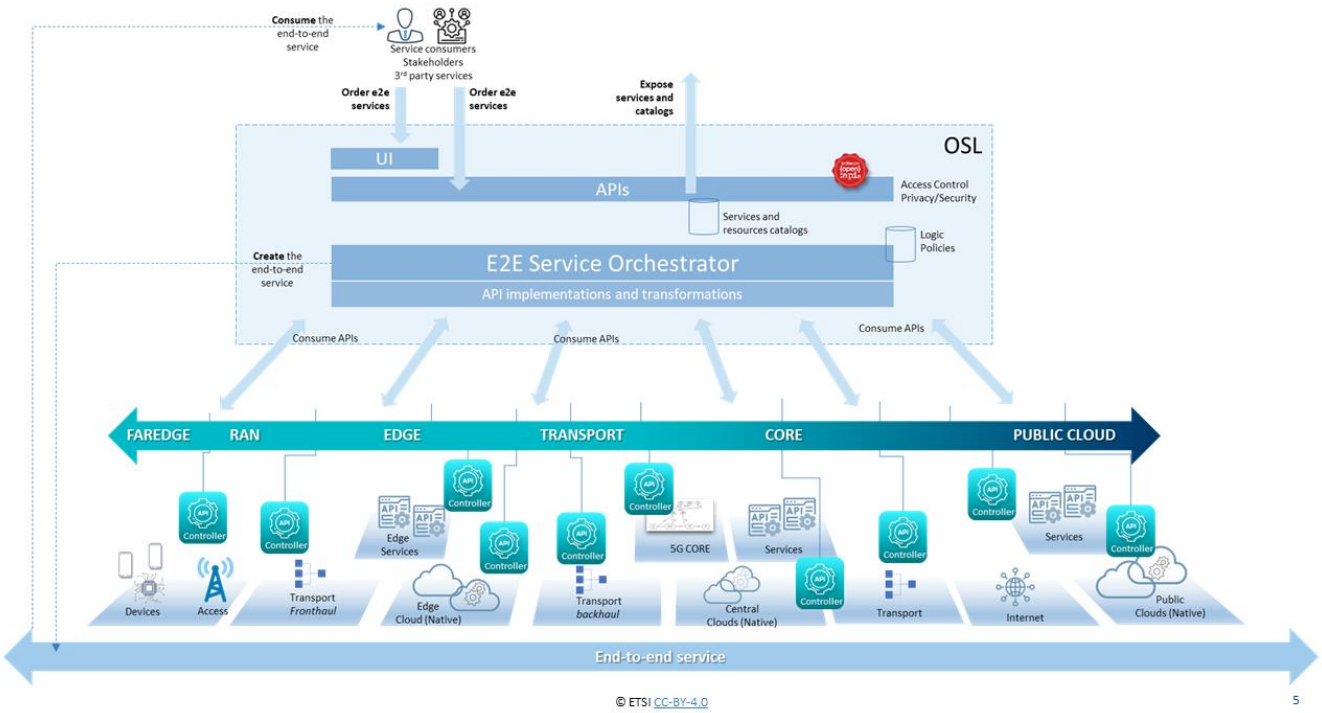


Figure 1 The general usage of OpenSlice

[1] OpenSlice Supported TMFORUM Exposed APIs: https://osl.etsi.org/documentation/latest/naas/exposed_apis/

An end-to-end (E2E) service orchestration framework

An end-to-end (E2E) service orchestration framework is designed to manage and automate the entire lifecycle of services across multiple domains and technologies. For delivering, especially, Network as a Service (NaaS) a comprehensive system is needed that automates and manages the entire lifecycle of network services, from provisioning to monitoring and decommissioning, while ensuring seamless integration, operation, and delivery of services from the initial request to the final delivery, spanning all involved components and layers. Such E2E frameworks enable users to consume network services on-demand, similar to how cloud computing services are consumed. Some key components and features of such frameworks are:

- Service Catalogs including predefined Network Services based on service templates for common network services like 5G core functions, 5G slices, VPNs, SD-WAN, firewalls, load balancers, etc. as well as custom Network services with Options for users to define their own network configurations.
- User Interface (UI) and APIs exposure, offering both a Self-Service Portal that allows users to request, configure, and manage network services as well as APIs for enabling programmatic access to network services for integration with other systems and automation scripts.
- Service Design and Creation tharough service templates based on predefined models for creating services.
- Automation and Workflow Management via Orchestration Engines, supporting Process Automation for automating repetitive tasks and processes, workflow management and orchestration for automating the provisioning, configuration, and management of network services while coordinating multiple workflows to ensure services are delivered efficiently, ensuring that services comply with predefined policies and standards.
- Standardized API exposure for seamless integration with different systems and services and APIs transformation support for converting data formats and protocols to ensure compatibility and information exhange between systems during workflows orchestration
- Service and Resource management and Orchestration while including the capability of multi-domain coordination in managing services/resources across different domains like cloud, 5G core, radios, transport network, and edge including dynamic allocation with adjusting resources based on demand and service requirements. To accomplish the above advanced technologies need to be exploited like, Containerized workloads, Network Function Virtualization (NFV) which uses virtualized network functions to provide services like routing, switching, and security and Software-Defined Networking (SDN) which Controls the network programmatically to dynamically manage traffic and resources.
- Monitoring and Analytics including Service Monitoring while continuously tracking the performance and health of services with capabilities to analyse data to optimize service delivery and predict issues. Real-Time Monitoring is also needed for tracking the performance and health of network services enabling analytics that provide insights for optimization and troubleshooting.
- Security and Access Control for ensuring only authorized users and systems can access network services. while implementing rules and policies to comply with regulatory requirements.

An E2E service orchestration workflow

In general an E2E service orchestration workflow includes the following phases:

- Service Request: Users or systems request a network service through the self-service portal or API. The request can specify details such as bandwidth, security features, geographic coverage, and duration.
- Service Orchestration: The orchestration engine evaluates the request, determines the necessary resources, and initiates the automated workflows.It interacts with the underlying components (e.g. 5G Core, Radios, Containerized controllers, NFV, SDN controllers) to provision and configure the required network functions and connectivity.
- Provisioning and Configuration: Services, network resources and network functions (VNFs) are instantiated and configured according to the service request during Service Orchestration through the orchestration engine. Other controllers manage their own domains, for example SDN controllers, manage the flow of data through the network to ensure optimal performance and adherence to policies, RAN controllers manage the RAN resoruces, Containerized controllers manage their workload, etc
- Service Delivery: The E2E network service is activated and made available to the user. Continuous monitoring ensures the service operates as expected, with automatic adjustments made as necessary.
- Lifecycle Management: The orchestration framework handles updates, scaling, and any necessary modifications throughout the service lifecycle.
- At the end of the service period, resources are decommissioned and reclaimed.

OpenSlice for Service Providers

OpenSlice is used by Service Providers to design Network Services, expose them in Service Catalogues and make them available for Service Orders. OpenSlice then can perform the E2E service orchestration workflow.

There are various portals offering UI friendly access to users acting as Service Providers:

- The Services portal allows Service Providers to design and expose services.
- The Resource portal allows users to access resource specifications and running resources in resource inventory.
- The NFV portal allows users to manage NFV artifacts and onboard them to a target MANO/NFV Orchestrator.
- The Testing portal allows Service Providers to manage test artifacts
- The Products portal allows Service Providers to expose services as products

OpenSlice for Service Consumers

OpenSlice allows Service Consumers to browse the available offered service specifications in a self-service manner. It also supports TMFORUM Northbound APIs regarding Service Catalog Management, Ordering, Resource, etc. There are various portals offering UI friendly access to users acting as Service Consumers:

- The Services portal allows Service Consumers to select and order predefined services.
- The Resource portal allows users to access running resources in resource inventory.
- The NFV portal allows users to self-manage NFV artifacts and onboard them to a target MANO/NFV Orchestrator.
- The Testing portal allows Service Consumers to manage test artifacts

- The Products portal allows Service Consumers to expose services as products
- 3rd party applications can use OpenSlice through TMForum Open APIs.

Network as a Service (NaaS)

This section describes some core concepts for Delivering Network as a Service in OpenSlice. There are many articles and reports on the subject like:

- TMF909 API Suite Specification for NaaS
- TMF926A Connectivity as a Service
- TMF931-Open Gateway Onboarding and Ordering Component Suite
- GSMA Open Gateway initiative

In general Network as a Service (NaaS) is a service model that allows users to consume network infrastructure and services, similar to how they would consume other cloud services like Software as a Service (SaaS) or Infrastructure as a Service (IaaS). NaaS abstracts the complexity of managing physical network infrastructure, providing users with virtualized network resources that can be dynamically allocated and managed through software.

OpenSlice and NaaS

OpenSlice makes extensive use of TMFORUM's models and APIs. Therefore if one is familiar with TMF APIs the terminology and ideas are the same.

To deliver NaaS we need to incorporate various APIs (see TMF909 API Suite Specification for NaaS). OpenSlice implements various TMF APIs to deliver NaaS and support the lifecycle functions required to manage the network capabilities exposed as Network as a Service and managed by operational domains.

Key Highlights of the Release 2024Q2

Cloud native support

OpenSlice can be deployed either via docker-compose in a single machine or via a HELM Chart in a Kubernetes cluster.

Installation via docker compose

Installation via docker compose, is intended for users who would like to evaluate OpenSlice or developers of OpenSlice. The method is performed via a dedicated deployment script.

Read more at: https://osl.etsi.org/documentation/latest/deploymentCompose/

Installation via a HELM Chart in a Kubernetes cluster.

Installation via a HELM chart, is intended for users who would like to install a more scalable version of OpenSlice.

Read more about the topic at: https://osl.etsi.org/documentation/latest/deploymentK8s/

Native Kubernetes support

OpenSlice since the beginning offered the exposure of Network Services via the NFV model, with extensive support of OpenSourceMANO (ETSI OSM). Since this release OpenSlice can now create and manage Custom Resources (CRs) and offer them as a Service. Also, it can deploy Services based on HELM Charts to target Kubernetes clusters. Therefore, OpenSlice is capable to:

- Create and manage Custom Resources (CRs) using installed CRDs on a target Kubernetes cluster.
- Facilitate complex orchestration scenarios by wrapping Kubernetes APIs as TMF APIs and models.
- Handles connectivity to a Kubernetes cluster and manages the lifecycle of CRDs
- Wraps the Kubernetes API, Receives and provides resources towards other OpenSlice services via the service bus

The approach allows OpenSlice to enable Loose Coupling and Orchestration via:

- Language Flexibility: Developers can write CRDs in any language and expose them via the Kubernetes APIs. OSL will reuse these CRDs, enhancing flexibility and integration capabilities.
- Familiar Deployment: Developers can create and deploy applications using familiar tools such as Helm charts, simplifying the process and reducing the learning curve.

Kubernetes is an orchestration system for automating software deployment, scaling, and management. One can interact though the Kubernetes API and it has a set of objects ready for use out of the box. Custom Resource Definitions (CRDs) is a way that allows to manage things other than Kubernetes itself and allows to create our own objects The use of CRDs makes the possibilities of Kubernetes management almost limitless. You can extend the base Kubernetes API with any object you like using CRDs.

By allowing the design and lifecycle management of services/resources that expose CRDs/CRs in a Kubernetes cluster via the TMF APIs, OSL can be used in many complex scenarios involving now resources from multiple domains.

OpenSlice capitalizes on the extensive Kubernetes ecosystem, particularly focusing on operators (CRDs) while key repositories and hubs such as artifacthub.io and Operatorhub.io can be utilized for finding and deploying operators. With this feature, OpenSlice can expose CRs in service catalogs, facilitating their deployment in complex scenarios. These scenarios may include service bundles that involve multiple systems, such as RAN controllers or other Kubernetes clusters, providing a robust and versatile deployment framework.

OpenSlice in general is responsible for exposing service specifications which are ready to be ordered and orchestrated, through TMForum Open APIs as defined in the OSL Service Spec Catalog. Usually for a service specification a corresponding (one or more) resource specification (resourceSpecificationReference) is registered in the OSL Resource Spec Catalog. The following image illustrates the approach.

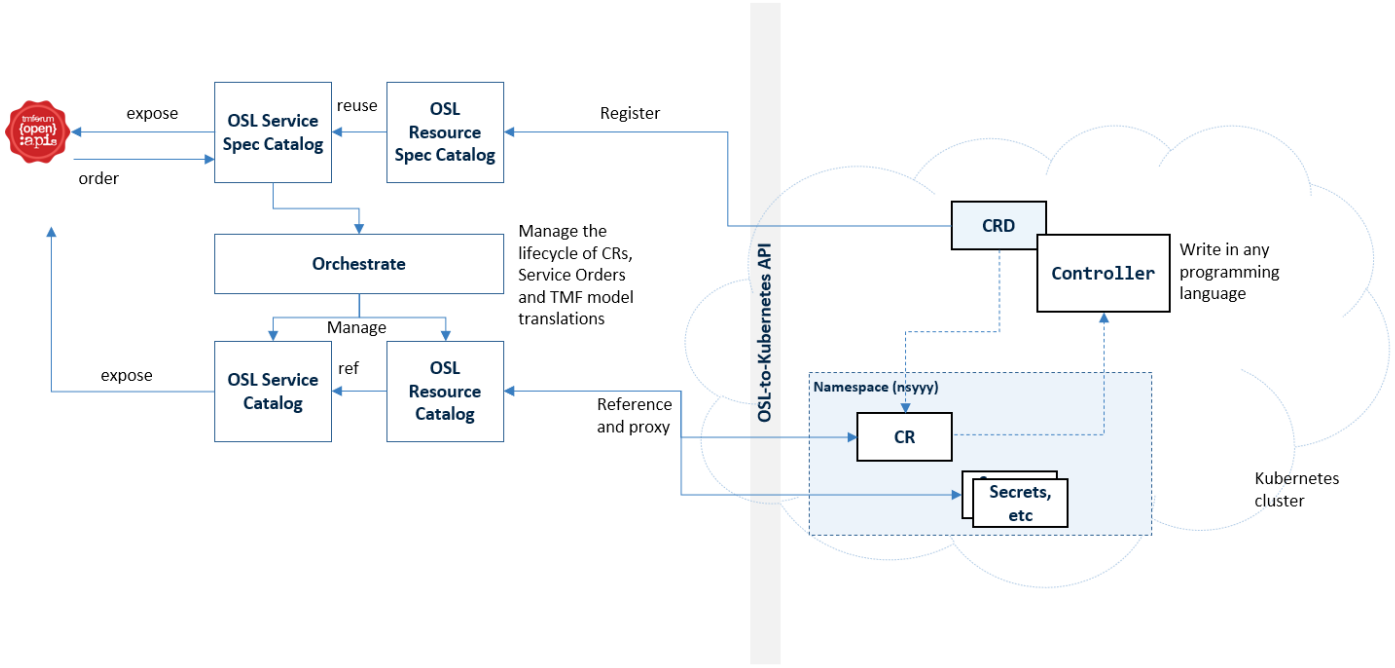


Figure 2 Exposing CRDs and managing CRs through OpenSlice

1. A CRD in a cluster will be mapped in TMF model as a Resource specification and therefore can be exposed as a service specification in a catalog
2. Service Orders can be created for this service specification.
3. OSOM creates a Resource in OSL Resource inventory and requests new Custom Resource (CR) in the target cluster
    - The resource is created in a specific namespace (for example the UUID of the Service Order)
    - A CR in a cluster will be mapped in TMF model as a Resource in the resource Inventory
    - Other related resources created by the CRD Controller within the namespace are automatically created in OSL Resource Inventory under the same Service Order

Read more about the topic at:

https://osl.etsi.org/documentation/latest/service_design/kubernetes/ExposingKubernetesResources/

Expose HELM charts as Service Specifications

Helm is a tool that automates the creation, packaging, configuration, and deployment of Kubernetes applications by combining your configuration files into a single reusable package

At the heart of Helm is the packaging format called charts. Each chart comprises one or more Kubernetes manifests and a given chart can have child charts and dependent charts, as well. Using Helm charts and offer them as a service has the following benefits:

- Reduces the complexity of deploying Microservices
- Enhances deployment speed
- Developers already know the technology
- There are many Helm charts and Helm repositories there that are ready to be used
- Enable loose coupling and more orchestration scenarios
- Developers create and deploy applications in things they already know (e.g. Helm charts)
- Use the TMF models as wrapper entities around Helm charts

OpenSlice can be used to expose HELM's as services in service catalogs and there are ready to be deployed in complex scenarios (service bundles) involving also other systems during the orchestration, for example:

- Include e.g. RAN controllers
- Pass values through life cycle rules from one service to another
- Manage multiple Helms in multiple clusters

The installation of HELM charts is based on OpenSlice latest feature on Kubernetes operators (CRD) support. For installing HELM charts we will use ArgoCD a well known Kubernetes-native continuous deployment (CD) tool. We will mainly use the CRD of **`Kind: Application`** that ArgoCD can manage

Before proceeding, install ArgoCD in your management cluster, by following ArgoCD instructions

As soon as you install ArgoCD,, OpenSlice is automatically aware for specific new Kinds. The one we will use is the **`Kind: Application`** that ArgoCD can manage under the **`apiGroup argoproj.io`**. Browse to Resource Specifications. You will see an entry like the following:

> ArgoCD is a Kubernetes-native continuous deployment (CD) tool. While just deploying Heml charts is just a scenario for ArgoCD , in future one can exploit it for many things. Despite some other tools like FluxCD, it provides also a UI which is useful for management and troubleshooting.

`Application@argoproj.io/v1alpha1@kubernetes@https://10.10.10.144:6443/`

This means that OpenSlice can use now this operator for managing HELM chart installations. (*Please note that the https://10.10.10.144:6443/ part of the Resource Specification's name will vary in different Kubernetes environments.*)

Example: Offer Jenkins as a Service via Openslice

In this example, we will use the Kind: Application of ArgoCD and create a ResourceFacingServiceSpecification (RFSS) for Jenkins. Eventually, we will offer Jenkins as a Service.

1. Go to Service Specifications
2. Create New Specification
3. Provide a Name, eg. jenkinsrfs

4. Go to Resource Specification Relationships
5. Assign **Application@argoproj.io/v1alpha1@kubernetes@https://10.10.10.144:6443/** as a related Resource Specification

Continue the example here:

https://osl.etsi.org/documentation/latest/service_design/examples/helmInstallation_aaS_Example_Jenkins/HELM_Installation_aaS_Jenkins_Example/

Enhanced NFVO primitive support

OpenSlice can now automatically extract and display a list of available (VNF-level) primitives. It also offers a dedicated UI that enables seamless primitive execution.

Service characteristics' exposure revision

The transition of service characteristics between related services underwent a major revision to properly reflect and isolate customer and resource facing aspects.

Robustness and Performance enhancements

The latest release introduces several bugfixes, code testing extension, automated pipelines to validate the developed code and performance centric component enhancements.

Examples

NFV-based service design and deployment: Open5GS through VNFs:
https://www.youtube.com/watch?v=h83vKsEOqvM

Offer Jenkins as a Service via Openslice:
https://osl.etsi.org/documentation/latest/service_design/examples/helmInstallation_aaS_Example_Jenkins/HELM_Installation_aaS_Jenkins_Example/

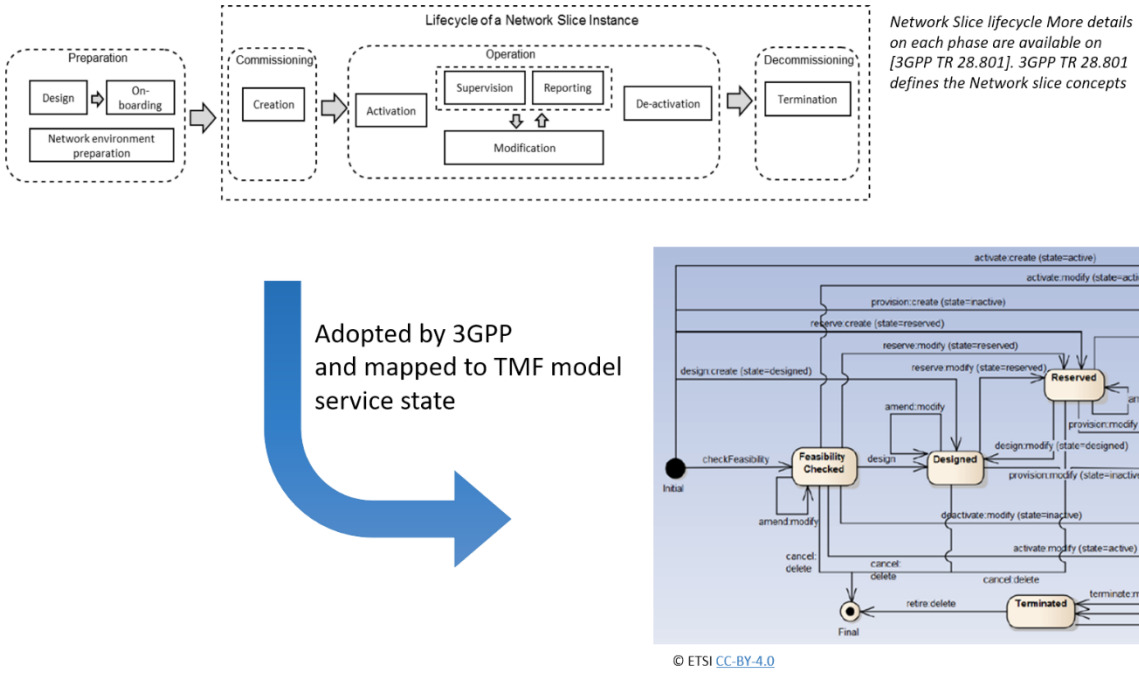
Exposing Kubernetes Operators as a Service : Offering "Calculator as a Service" through OpenSlice:
https://osl.etsi.org/documentation/latest/service_design/examples/ExposingCRDs_aaS_Example_Calculator/ExposingCRDs_aaS_Example_Calculator/

Dynamic orchestration

Service Designers create detailed service specifications, which are then managed and exposed in service catalogs. These services are integrated into OpenSlice E2E service orchestration framework to automate and optimize the delivery of network services. OpenSlice can be used to design service specifications for various services, even non-network related services.

Lifecycle Management: The orchestration framework handles the activation, termination and any necessary modifications throughout the service lifecycle.

In OpenSlice the Lifecycle of a service follows in general the concept of Network Slice lifecycle as defined by 3GPP.



© ETSI CC-BY-4.0

Figure 3: Service lifecycle in OpenSlice

OpenSlice adopted the LCM model by 3GPP and mapped to the TMF model service state. Next we discuss briefly the process and the relationships. The lifecycle of a service, particularly in the context of Network Service lifecycle encompasses several stages that ensure the service is effectively planned, deployed, managed, and eventually decommissioned. Read a detailed overview here: https://osl.etsi.org/documentation/latest/naas/lcm_intro/

Lifecycle Management Rules - LCM Rules

Defining complex conditions and actions during the lifecycle of a service and any necessary modifications throughout the service lifecycle. OpenSlice end-to-end (E2E) service orchestrator follows some predefined workflows to manage a service lifecycle (They are described in BPMN language and included in our orchestration engine). So in the system there are already predefined recipes, which in each process-step of the workflow some piece of code is executed. How is it possible to intervene in the workflow process and inject some user defined actions? The next image illustrates the idea

- OpenSlice workflows are predefined recipes.
- Practically each process executes code
- How a service designer can modify/affect OpenSlice workflows?

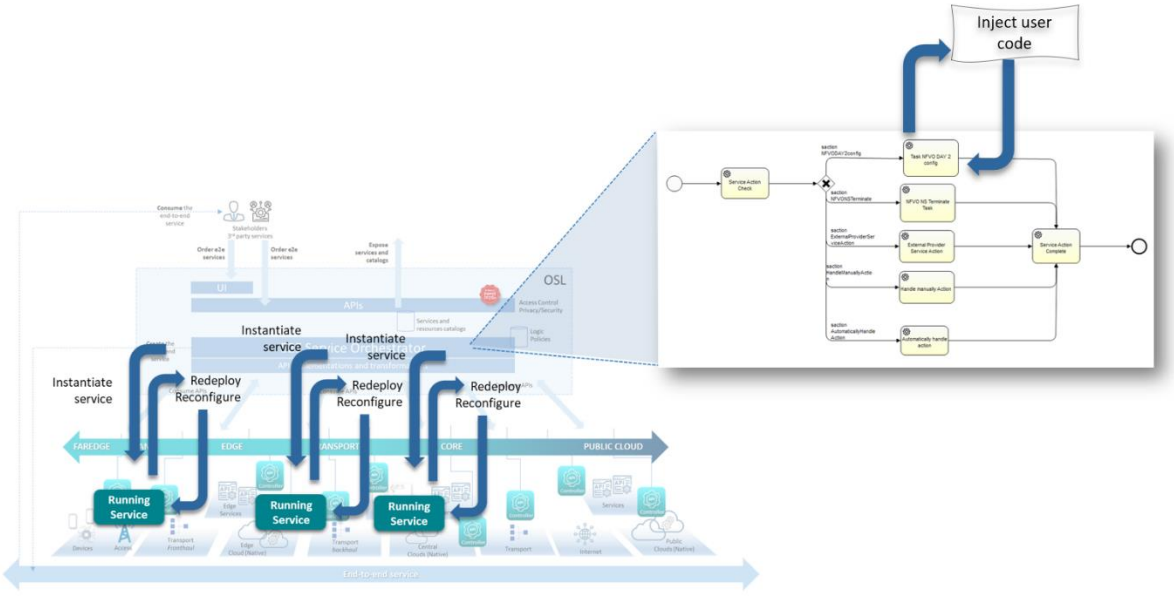


Figure 4 Affecting the orchestration process

How is it possible to intervene in the workflow process and affect it?

LCM Rules are used for defining complex conditions and actions during the lifecycle of a service. In Openslice there are the following types of rules defined:

- PRE_PROVISION
- CREATION
- AFTER_ACTIVATION
- SUPERVISION
- AFTER_DEACTIVATION

The following figure displays the different phases that the rules are performed, during the lifecycle of a Network Service Instance.

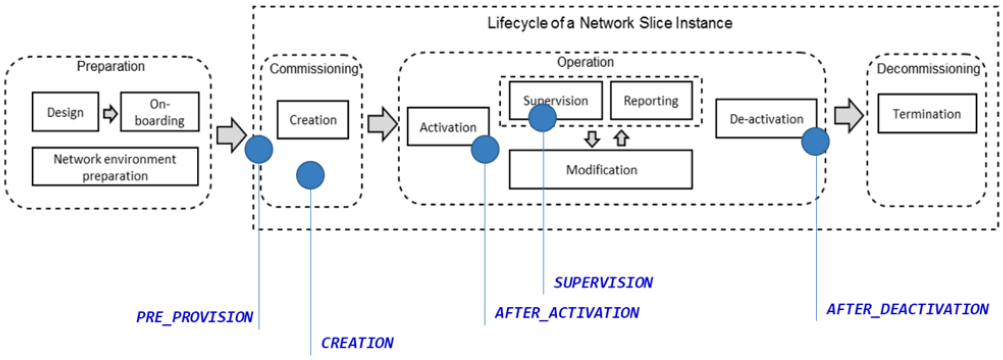


Figure 5 Phases that OpenSlice executes defined rules

- PRE_PROVISION rules: Run only once just before creating a service with a given priority.
- CREATION rules: Run while the referenced service dependencies of a service are created
- AFTER_ACTIVATION rules: Run only once just after a service get the ACTIVE state
- SUPERVISION rules: Run when a characteristic of a service is changed and the service is in the ACTIVE state
- AFTER_DEACTIVATION rules: Run only once just after a service get the INACTIVE/TERMINATED state

Read next here: https://osl.etsi.org/documentation/latest/service_design/lcmrules/intro/

OpenSlice 2024Q2 Release conclusion and next plans

ETSI SDG OSL 2024Q2 release of OpenSlice is the first release under the umbrella of ETSI. ETSI SDG OSL plans 2 releases per year, aiming to further support 5G/6G research and network services, while developing standards alignment and providing regular feedback to standardization activities.

Next release, will aim for more features, like: Monitoring integration, ready-to-install services in the pre-installed service catalog, improved Testing/Security in our CI/CD and many more.

Contact us

Information: https://osl.etsi.org

SDG Support: For ETSI SDG or OSL project related issues, contact SDG Support Team at SDGsupport@etsi.org

Slack: Join the User and Developer Communities on Slack

OSL TECH: Send you technical questions to OSL_TECH@list.etsi.org

OSL INFO: Stay tuned: subscribe the OSL_INFO mailing list.

OSL in the ETSI Portal: Check OSL page th ETSI Portal, register and contribute to OSL meetings.