LF NETWORKING

LFN Developer & Testing Forum

LF NETWORKING

LFN Developer & Testing Forum

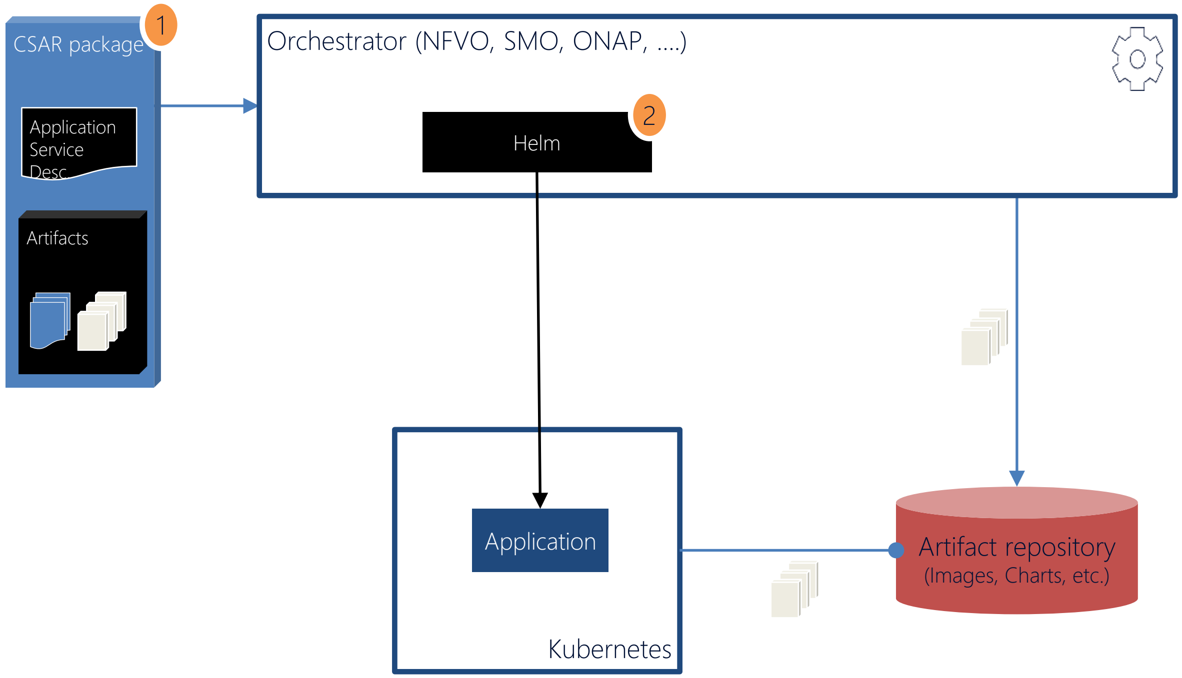
ONAP: Application Service Descriptor (ASD) for K8s NFs

Marian Darula, Byung-Woo Jun, Zu Qiang (Ericsson)
Thinh Nguyenphu, Joe Schumacher (Nokia)

Application Service Descriptor (ASD)

- Motivation
  - ASD is a common, simplified deployment descriptor for ONAP, O-RAN (NFs, xApps and rApps) aiming to:
    - Quickly leverage enhancements in Kubernetes while minimizing development and integration efforts
    - Avoid duplication of attributes/properties included in cloud native artifacts, e.g. Helm Charts
    - Descriptor format is not requiring a particular deployment tool (e.g. Helm)
    - Leverage established packaging standards (e.g., SOL004 with ASD as a top-level deployment artifact)
- Status
  - ONAP: The proposal is being reviewed at ONAP Modelling Subcommittee
    - ASD onboarding information model (IM)
    - ASD Resource Data Model
    - ASD Onboarding Packaging Format
    - Based on above proposal ASD and Application Onboarding and LCM Orchestration PoC is ongoing for ONAP Jakarta release
  - O-RAN: A technical proposal has been submitted to WG10 and WG6.

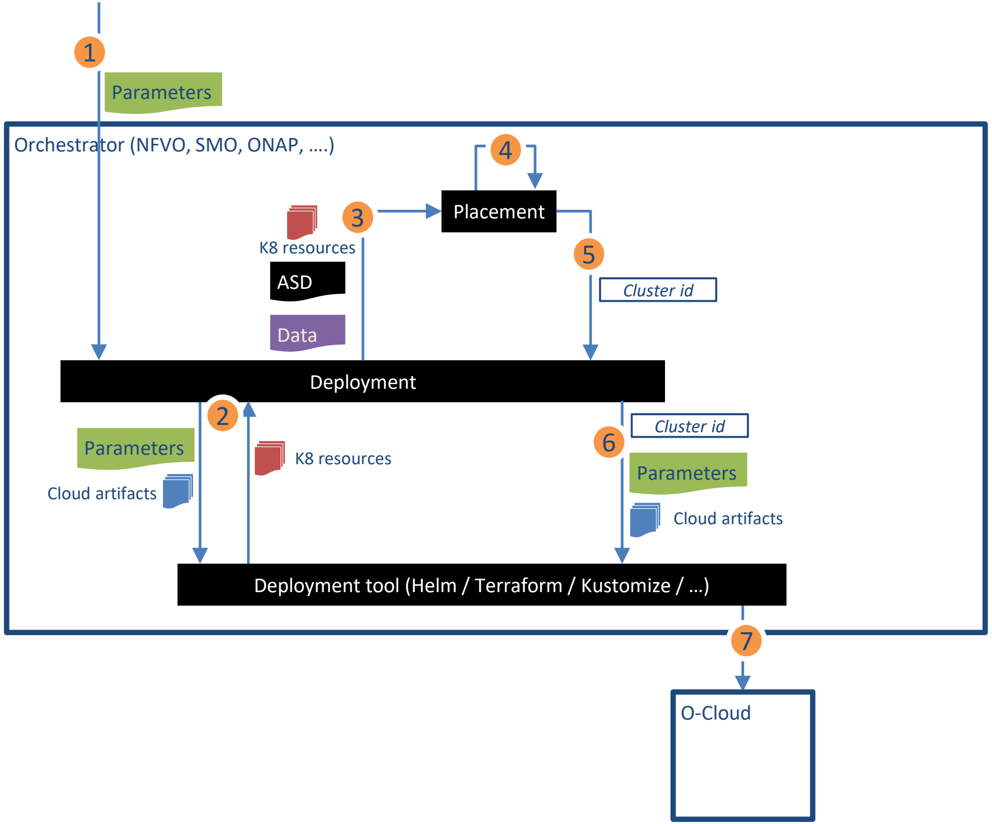
ASD Model and Packaging Proposal



The proposed solution:

1. Use CSAR packaging for bundling metadata and cloud-native artifacts in a single package. Describe the application using a lightweight Application Service Descriptor.
2. Choose Helm v3 as the initial cloud-native tool to embed in the orchestrator.

Example flow of application deployment



1. A deployment order is received, along with the required lifecycleParameters values
2. The cloud-native deployment tool is invoked with the received parameters to transform the cloud artifacts into K8S resource descriptions.
3. The K8S resource descriptions, ASD and any other relevant data is sent to the placement function
4. Placement decision is done based on input data
5. Inform deployment of placement
6. Request the cloud native deployment tool to deploy on the identified target cluster
7. Cloud native deployment tool deploys application in the chosen cluster using the K8S API.

Application Service Descriptor: Top Level

LF NETWORKING
LFN Developer & Testing Forum

Attribute	Qualifier	Cardinality	Content	Description
asdId	M	1	Identifier	Identifier of this ASD information element. This attribute shall be globally unique. The format will be defined in the data model specification phase.
asdSchemaVersion	M	1	Version	Specifies the version of the ASD's schema (if we modify an ASD field definition, add/remove field definitions, etc.)
asdProvider	M	1	String	Provider of the AS and of the ASD.
asdApplicationName	M	1	String	Name to identify the Application Service. Invariant for the AS lifetime.
asdApplicationVersion	M	1	Version	Specifies the version of the Application (so, if software, deplolymentItems, ASD values, … change, this changes)
asdApplicationInfoName	M	0..1	String	Human readable name for the Application service. Can change during the AS lifetime.
asdInfoDescription	M	0..1	String	Human readable description of the AS. Can change during the AS lifetime.
asdExtCpd	M	0..N	datatype.ExtCpd	Describes the externally exposed connection points of the application.
enhancedClusterCapabilities	M	0..1	datatype. enhancedClusterCapabilities	A list of expected capabilities of the target Kubernetes cluster to aid placement of the application service on a suitable cluster.
deploymentItems	M	1..N	DeploymentItem	Deployment artifacts

Described on next sldies

ASD: asdExtCpd

LF NETWORKING
LFN Developer & Testing Forum

Attribute	Qualifier	Cardinality	Content	Description
id	M	1	String	The identifier of this extCpdData
description	M	1	String	Describes the service exposed.
virtualLinkRequirement	M	1..N	String	Refers in an abstract way to the network or multiple networks that the ExtCpd shall be exposed on (ex: OAM, EndUser, backhaul, LI, etc). The intent is to enable a network operator to take decision on to which actual VPN to connect the extCpd to. Note 1.
networkInterfaceRealizationRequirements	M	0..1	datatype.networkInterfaceRealizationRequirements	Details container implementation specific requirements on the NetworkAttachmentDefinition. See Note 2 & 3.
inputParamMappings	M	0..1	Datatype.extCpd.ParamMappings	Information on what parameters that are required to be provided to the deployment tools for the asdExtCpd instance.
resourceMapping	M	0..1	String	Kubernetes API resource name for the resource manifest for the service, ingress or pod resource declaring the network interface. Enables, together with knowledge on namespace, the orchestrator to lookup the runtime data related to the extCpd.

Note 1: Corresponds more or less to a virtual_link requirement in ETSI NFV SOL001.
Note 2: Applies only for ExtCpds representing secondary network interfaces in a pod.
Note 3: Several ExtCpd may refer to same additional network interface requirements.

ASD:networkInterfaceRealizationRequirements

Attribute	Qualifier	Cardinality	Content	Description
trunkMode	M	0..1	”false” | ”true”	If not present or set to”false”, means that this interface shall connect to single network. If set to ”true” then the network interface shall be a trunk interface (connects to multiple VLANS).
ipam	M	0..1	"infraProvided", "orchestrated", "userManaged"	The default value ("infraProvided") means that the CNI specifies how IPAM is done and assigns the IP address to the pod interface.
interfaceType	M	0..1	"kernel.netdev", "direct.userdriver", "direct.kerneldriver", "direct.bond", "userspace	This attribute is applicable for passthrough and memif interfaces. Value default value is ”kernel.netdev”.
interfaceOptions	M	0..N	"virtio", "memif"	Alternative vNIC configurations the network interface is verified to work with.
interfaceRedundancy	M	0..1	"infraProvided", "activePassiveBond", "activeActiveBond", "activePassiveL3", "activeActiveL3",	”infraProvided” means that the application sees one vNIC but that the infrastruture provides redundant access to the network via both switch planes. ”Left” and ”right” indicates a vNIC connected non-redundantly to the network via one specific (left or right) switchplane. All other attributes indicates a mated vNIC pair in the Pod, one connecting to the network via left switchplane and the other connecting to the network via the right switchplane, and with application using them together as a redundant network interface using a particular redundancy method that need to be accomodated in the node infrastructure.
nicOptions	M	0..N	"examples": ["i710", "mlx-cx5v"]	nics a direct user space driver the application is verified to work with. Allowed values from ETSI registry.

ASD:datatype.ExtCpd.ParamMappings

LF NETWORKING
LFN Developer & Testing Forum

Attribute	Qualifier	Cardinality	Content	Description
loadbalancerIP	M	0..1	String	When present, this attribute specifies the name of the deployment artifact input parameter through which the orchestrator can configure the loadbalancerIP parameter of the K8s service or ingress controller that the ExtCpd represents.
externalIPs	M	0..N	String	When present, this attribute specifies the name of the deployment artifact input parameter through which the orchestrator can configure the extermalIPs parameter of the K8s service or ingress controller, or the pod network interface annotation, that the ExtCpd represents. The param name and provided IP address(es) value will be passed to the deployment tool when deploying the DeploymentArtifacts. Note 1
nadName	M	0..N	String Note 2, Note 3	These attributes specifies, for an ExtCpd respesenting a secondary network interface, the name(s) of the network attachment definitions (NADs) the orchestrator has created as base for the network interface the ExtCpd represents. It is expected that the NADs themselves have been created prior to the deployment of the deployment artifacts.
nadNamespace	M	0..1	String	Specifies, for an asdExtCpd respresenting a secondary network interface, the namespace where the NADs are located. Attribute may be omitted if the namespace is same as the application namespace.

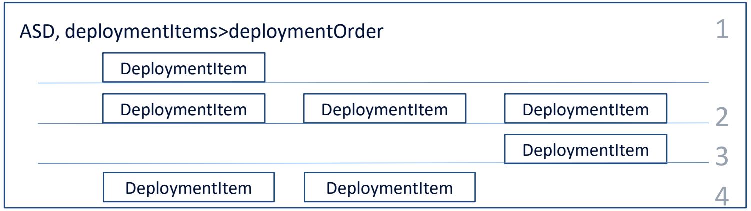
Note 1: When the asdExt Cpd represent a networkRedundant/mated-pair of sriov interfaces, there are references to 2 or 3 related NADs needed to be passed, while for other interface types only one NAD reference is needed to be passed.
Note 2: The format of the Content strings is specific for each different orchestration templating technology used (Helm, Teraform, etc.). Currently only a format for use with Helm charts is suggested: "*helmchartname:[subchartname.]0..N[parentparamname.] 0..Nparametername*". Whether the optional parts of the format are present depends on how the parameter is declared in the helm chart.
Note 3: A direct attached (passthrough) network interface, such as an sriov interface, attaches to a network via only one of the two switch planes in the infrastructure.

ASD: enhancedClusterCapabilities

Attribute	Qualifier	Cardinality	Content	Description
minKernelVersion	M	1	String	Describes the minimal required Kernel version, e.g. 4.15.0. Coded as displayed by linux command uname –r
requiredKernelModules	M	0..N	String	Required kernel modules are coded as listed by linux lsmod command, e.g. ip6_tables, cryptd, nf_nat etc.
conflictingKernelModules	M	0..N	String	Kernel modules, which must not be present in the target environment. The kernel modules are coded as listed by linux lsmod command, e.g. ip6_tables, cryptd, nf_nat etc. Example: Linux kernel SCTP module, which may conflict with use of proprietary user space SCTP stack provided by the application.
clusterLabels	M	0..N	String	This attribute allows to associate arbitrary labels to clusters. These can indicate special infrastructure capabilities (e.g., NW acceleration, GPU compute, etc.). The intent of these labels is to serve as a set of values that can help in application placement decisions. clusterLabels follow the Kubernetes label key-value-nomenclature (https://kubernetes.io/docs/concepts/overview/working-with-objects/labels/). It is recommended that labels follow a standardised meaning e.g. for node features (https://kubernetes-sigs.github.io/node-feature-discovery/v0.9/get-started/features.html#table-of-contents). Example: ClusterLabels - feature.node.kubernetes.io/cpu-cpuid.AESNI: true
requiredPlugin	M	0..N	Structure (inlined)	A list of the names and versions of the required K8s plugin (e.g. multus v3.8)
>requiredPluginName	M	0..1	String	The names of the required K8s plugin (e.g. multus)
>requiredPluginVersion	M	0..1	String	The version of the required plugin (e.g. 3.8)

ASD: deploymentItems

Attribute	Qualifier	Cardinality	Content	Description
deploymentItemId	M	1	Identifier	The identifier of this deployment item
artifactType	M	1	String	Specifies the artifact type. One of following values can be chosen: "helm_chart", "helmfile", "crd", "terraform"
artifactId	M	1	String	Reference to a DeploymentArtifact. It can refer to URI or file path.
deploymentOrder	M	0..1	Integer	Specifies the deployment stage that the DeploymentArtifact belongs to. A lower value specifies that the DeploymentArtifact belongs to an earlier deployment stage, i.e. needs to be installed prior to DeploymentArtifact with higher deploymentOrder values. If not specified, the deployment of the DeploymentArtifact can be done in arbitrary order and decided bu the orchestrator.
lifecycleParameters	M	0..N	String	The list of parameters that can be overridden at deployment time (e.g., the list of parameters in the values.yaml which can be overridden at deployment time)



In order to support complex applications that require multiple artifacts (like Helm charts) to be installed in a particular order, the orchestrator must support an easy method of chaining these artifacts – including dependency relationships.

As shown, items are given a deployment order. Items with the same order are deployed in parallel; items with different orders are deployed in

ASD Resource Data Model

- The initial ASD Resource Data Model proposal is based on TOSCA language specification
  - TOSCA parser available in ONAP
  - Use minimal set of TOSCA language capabilities defining ASD schema
  - Defining TOSCA node types of ASD
  - Data model resembling “plain yaml”
  - To be used in the ongoing PoC, the final names for node types can be selected as a part of standardization.

ASD node_templates example


```
…
node_templates:
  applicationServiceDescriptor:
      type: tosca.nodes.asd
      description: "Sample Application"
      properties:
        descripter_id: fdsa-3211-sdfsd-wqeuy
        descripter_invariant_id: ddsx-2234-wers-1234
        version: 1.0
        schema_version: 2.0
        provider: MyCompany
        application_name: SampleApp
        application_version: 2.3
        application_info_name: Sample Application
        extCpds:
          - id: 1
            description: webpage-service
            virtual_link_requirement: endUser
          - id: 2
            description: transactionAPI
            virtual_link_requirement: backhaul
        enhanced_cluster_capabilities: [ ... ]
      artifacts: #these are the deployment items:
        sampleapp-db:
          type: tosca.artifacts.asd.deploymentItem
          file: "sampleapp-db-operator-helm.tgz" # or a URI
          properties:
            artifact_type: "helm_chart"
            itemId: 1
            deployment_order: 1
            lifecycle_parameters:
              - ".Values.db.fullBackupInterval"
              - ".Values.db.walConsolidationInterval"
```

ASD TOSCA service template

```
tosca.nodes.asd:
    derived_from: tosca.nodes.Root
    description: "The ASD node type"
    version: 0.1
    properties:
      descriptor_id:
          type: string # UUID
          required: true
          description: Identifier of this ASD.
          It is in UUID format as specified in RFC 4122
      descriptor_invariant_id:
          type: string # UUID
          required: true
          description: >
              Identifier of this descriptor in a version independent manner.
              This attribute is invariant across versions of ASD.
              It is in UUID format as specified in RFC 4122
…
```

(other attributes as per ASD Resource Data Model)

```
…
      extCpds :
          type: list
          required: false
          entry_schema:
               type: tosca.datatype.asd.extCpdData
          description: >
                   Describes the externally exposed connection points of the application
                   service described by this ASD
      enhanced_cluster_capabilities :
          type: tosca.datatype.asd.enhancedClusterCapabilities
          required: false
          description: >
                 A list of  expected capabilities of the target Kubernetes cluster to aid
                 placement of the application service on a suitable cluster.
…
```

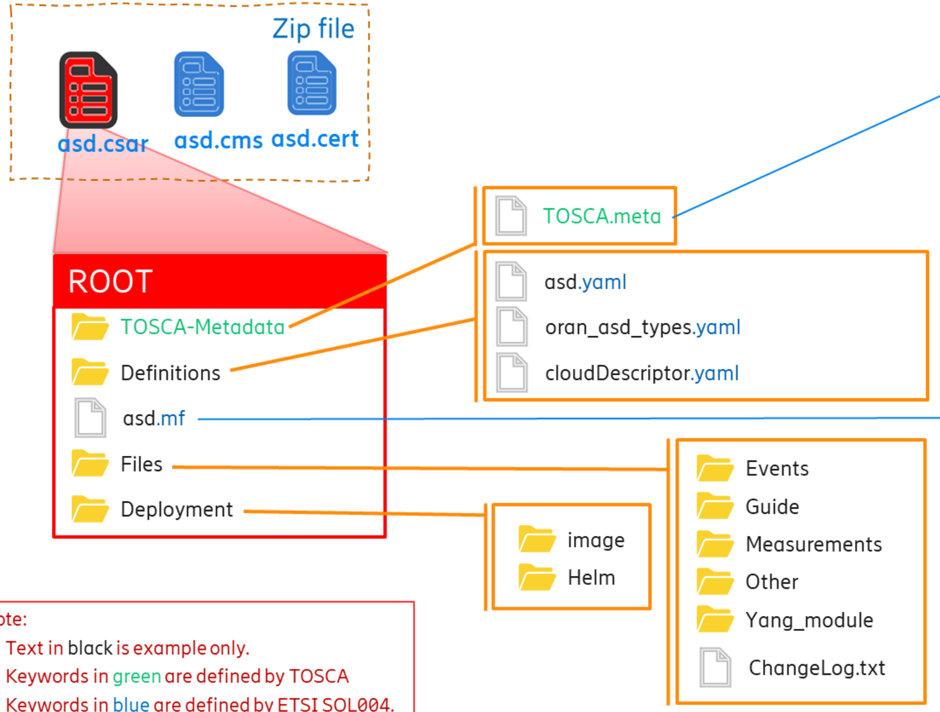
ASD type definition

ASD Onboarding Packaging Format

In order to facilitate compatibility with ETSI, ONAP and other telco standards, the CSAR (NFV SOL 004ed421) packaging format is used with following details:

- The structure and format of an ASD package shall conform to the TOSCA Simple Profile YAML v1.1 Specification of the CSAR format. The zip file format shall conform to Document Container Format File
- CSAR format with TOSCA-Metadata directory, specified in ETSI NFV SOL004ed431 section 4.1.2, with the differences that the following TOSCA.meta file keynames extensions are optional:
  - ETSI-Entry-Change-Log
  - ETSI-Entry-Tests
  - ETSI-Entry-Licenses
- Non-MANO artifact sets, specified in ETSI NFV SOL004ed431 section 4.3.7
- Registered non-MANO artifact keywords can be reused, to avoid duplication
- Package and artifacts security, specified in ETSI NFV SOL004 ed431 section 5 and 4.3.6
- Package manifest file, specified in ETSI NFV SOL004ed431 section 4.3.2, with new manifest metadata proposed in the wiki page
- Additional non-mano-artifact keywords for 5G use cases.

An example of an ASD onboarding package



TOSCA-Meta-File-Version: 1.0
CSAR-Version: 1.0
Created-By: Ericsson (Zu Qiang 2021-10-21)
Entry-Definitions: Definitions/asd.yaml

ETSI-Entry-Manifest: asd.mf

```
metadata:
application_name: vCU
application_provider: Ericsson
release_date_time: 2021-10-21T11:30:00+05:00
entry_definition_type: asd

Source: Definitions/asd.yaml
Signature: Definitions/asd.sig.cms
Certificate: Definitions/asd.cert
Source: Definitions/app.yaml
Signature: Definitions/app.sig.cms
Certificate: Definitions/app.cert
Source: Definitions/cloudDescriptor.yaml
Signature: Definitions/cloudDescriptor.sig.cms
Certificate: Definitions/cloudDescriptor.cert

non_mano_artifact_sets:
onap_ves_events:
        source: Files/Events/VES_registration.yaml
onap_pm_dictionary:
        source: Files/Measurements/PM_Dictionary.yaml
onap_yang_module:
        source: Files/Yang_module/Yang_module.yaml
onap_ansible_playbooks:
        source: Files/Playbooks/playbook.yml
onap_others:
        source: Files/Other/installation_guide.txt
        source: Files/Other/review_log.txt
onap_vendor_artifacts:
        vendor_name: Ericsson
        artifact_type: [rApp_name1]
        source:/artifacts/[rapp_name1]/rapp.zip
```

This is one of the possibilities to introduce ASD type, since ASD metadata keywords are not defined in any STO yet.

Note:
- Text in black is example only.
- Keywords in green are defined by TOSCA
- Keywords in blue are defined by ETSI SOL004.
- Keywords in purple are defined by ONAP
- Keywords in red are new

ASD Link to network service modelling

- ASD has one K8S cluster scope
- Link to network level modelling
- One option is to use ETSI MANO NSD
  - To avoid any impacts on ETSI NSD standarization and current ONAP implementation
  - A new node type is proposed: asdInNsd
  - The approach can be used for early PoC
- ASD allows to integrate with other options for network service modelling, the solutions are under discussion.

NSD & ASD

LF NETWORKING

Details can be found here

NSD template

```
tosca_definitions_version: tosca_simple_yaml_1_3
imports: nsd_types.yaml

node_types:
 tosca.example_NS:
  derived_from: tosca.nodes.nfv.NS
  properties:
   descriptor_id:
    type: string
    constraints: [ equal: b1bb0ce7-ebca-4fa7-95ed-4840d70a1177 ]
    default: b1bb0ce7-ebca-4fa7-95ed-4840d70a1177
...

topology_template:

 node_templates:
  my_service:
   type: tosca.example_NS
   properties:
    descriptor_id: b1bb0ce7-ebca-4fa7-95ed-4840d70a1177
    designer: MyCompany
    name: ExampleService
    version: '1.0'
    invariant_id: 1111-2222-aaaa-bbbb
    flavour_id: simple
   interfaces:
    Nslcm:
     operations:
      instantiate:
       implementation: instantiate.workflow.yaml
      terminate:
       implementation: terminate.workflow.yaml


  asd_1:
   type: tosca.nodes.asdInNsd
   properties:
    descriptor_id: fdsa-xdsfg-sdfsd-wqeuy
    descriptor_invariant_id: fdsa-xdsfg
    vnf_profile:
     min_number_of_instances: 0
     max_number_of_instances: 2
   requirements:
    - virtual_link_1: NsVirtualLink_1
    - virtual_link_2: NsVirtualLink_2
...
```

descriptor_id must be defined in the NSD, and exposed as a substitution filter in the ASD

The NSD must associate the virtual link requirements to the correct NS Virtual Link nodes for the networks that each extCpd shall connect to

ASD type

```
tosca_definitions_version: tosca_simple_yaml_1_3

node_types:

 tosca.nodes.asdInNsd:
  derived_from: tosca.nodes.nfv.VNF
  properties:  # all these properties are irrelevant for the ASD, disabled here:
   flavour_id:
    type: string
    constraints: [ equal: "simple" ]
    default: "simple"
   flavour_description:
    type: string
    default: ""
    constraints: [ equal: "" ]
   vnfm_info:
    type: list
    entry_schema:
     type: string
     constraints: [ equal: "" ]
    default: [ "" ]
  requirements:
   - virtual_link_1:
     capability: tosca.capabilities.nfv.VirtualLinkable
     relationship: tosca.relationships.nfv.VirtualLinksTo
     occurrences: [ 0, 1 ]
   - virtual_link_2:
     capability: tosca.capabilities.nfv.VirtualLinkable
     relationship: tosca.relationships.nfv.VirtualLinksTo
     occurrences: [ 0, 1 ]
   - ...
```

ETSI properties not used for the ASD will have default fixed values

The generic ASD node type will contain a fixed set of virtual link requirements, to be used for the extCpds

ASD template

```
tosca_definitions_version: tosca_simple_yaml_1_3
imports:
 - asd_types.yaml

topology_template:

 node_templates:
  applicationServiceDescriptor:
   type: tosca.nodes.asd
   version: 1.0
   description: "Sample Application to Illustrate ASD Usage"
   properties:
    descriptor_id: fdsa-xdsfg-sdfsd-wqeuy
    descriptor_invariant_id: fdsa-xdsfg
    version: 1.0
    schema_version: 1.0
    function_description: "Sample Application"
    provider: MyCompany
    application_name: SampleApp
    application_version: â€œ2.3â€
    application_info_name: â€œSample Applicationâ€
    ext_cpds:
     - id: webpage-service
      virtual_link_requirement: endUser
     - id: transactionAPI
      virtual_link_requirement: backhaul
    enhanced_cluster_capabilities: [ o-ran.o-cloud.hw.gpgpu ]
   artifacts: #these are the deployment items:
...
```

The ASD will not include tosca requirements for each extCpd virtual link. Instead in the ASD_in_NSD the virtual_link_1 will provide the connectivity requirement for the first CP in the extCpds list etc.

Application Service Descriptor - Summary

- Application Service Descriptor (ASD) provides simplified way of modelling and packaging of NFs
  - It’s an alternative to complex ETSI MANO based approach.
  - Relies on cloud native modeling tools (e.g. helm),complemented by slim descriptor layer providing information which cannot be conveyed via native modeling tools (e.g. networking related information)
  - Not repeating information from the native tools.
  - Utilizing established standards where applicable (e.g. TOSCA for the schema definition, the CSAR structure based on SOL004ed431 format).
- PoC ongoing to proof the concept in ONAP environment *LINK*