K8S Application Service Descriptor and Packaging

(REQ & Modeling Subcommittees)

Nokia: Thinh Nguyenphu, Timo Perala,
Ericsson: Marian Darula, Byung-Woo Jun

2021-10-21

Contents

- Part 1: Overviews, status, and plans
- Part 2: Application Service Descriptor (ASD) and Packaging
- Part 3: ONAP PoC ASD

ONAP
OPEN NETWORK AUTOMATION PLATFORM

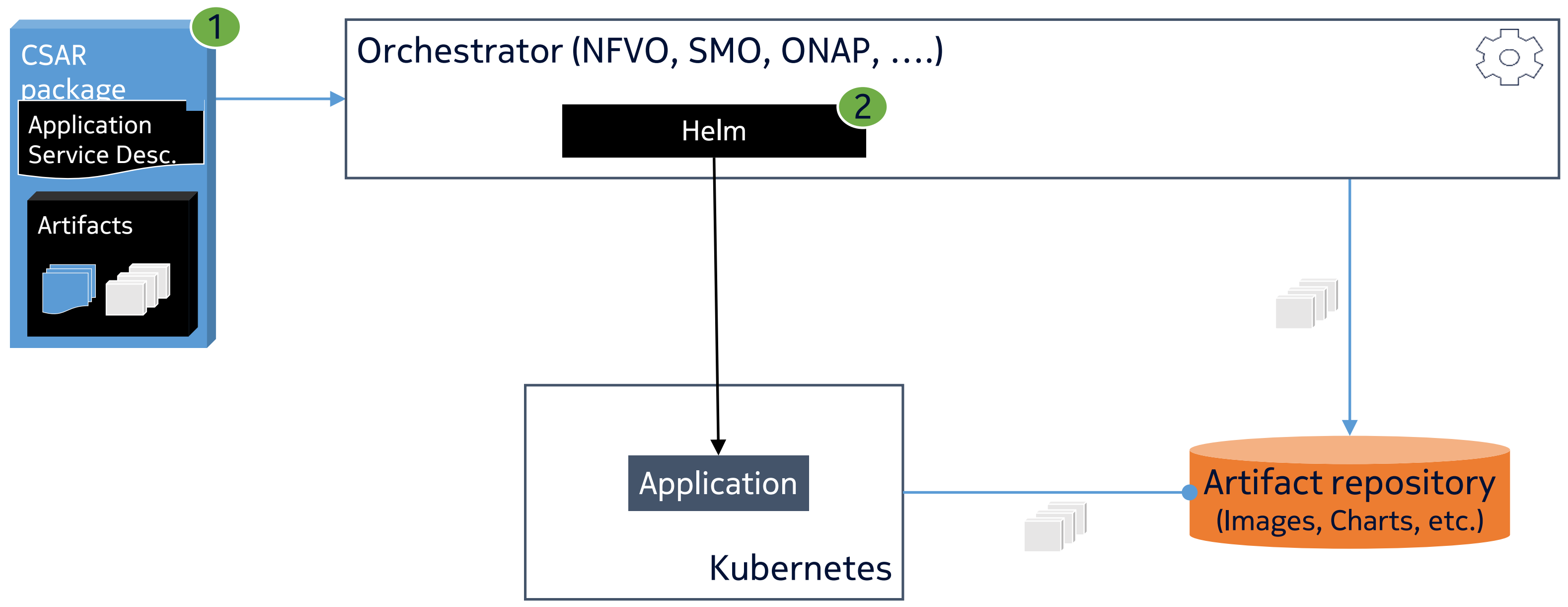
Part 1: Overview and status

ASD & packaging summary

- 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 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)
- ONAP: Work is in progress in Orchestration Scenarios, https://wiki.onap.org/display/DW/Orchestration+Scenarios
- Detail proposal with example: https://wiki.onap.org/display/DW/Application+Service+Descriptor+%28ASD%29+and+packaging+Proposals+for+CNF
- Status: Completed technical proposal with detail ASD information model, examples, and PoC plan
  - Jakarta release - functional requirements
  - REQ ticket in jira: https://jira.onap.org/browse/REQ-993
- Plans:
  - Jakarta release: ASD information model specification, ASD Orchestration PoC
  - ONAP Modeling Sub: Planning to present to ONAP Modeling Sub and requesting to start the work. Target: Resource modeling team
  - O-RAN: A technical proposal has been submitted to WG10 and WG6. WG10 has started the discussion on Aug4, meanwhile WG6 is scheduled on August 31. Proposal recommending for November Train targets:
    - Incorporate ASD into WG10 OAM Architecture Document v06--Common App LCM section
    - Incorporate ASD into WG10 IM/DM Specification v2.0
    - Incorporate ASD into the O-RAN-WG1.Package-Definition.0-v00.01
    - Incorporate ASD into WG6 User Case document and possibility into O2 GAAP document

Part 2: ASD and Package:

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, ETSI-compatible Application Service Descriptor.
2. Choose Helm v3 as the primary cloud-native tool to embed in the orchestrator.

Application Service Descriptor

Application Service Descriptor, top level

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

Description proposal attributes

asdExtCpd, simple configuration example

- Use a small subset of the attributes of the full asdExtCpd proposal based on working assumption:
  - Leave out the possibility for a secondary network interface to specify trunk interfaces and passthrough/sriov interfaces and assume IPAM to be CNI managed if inputParamMappings do not include ipAddress mapping.
  - All primary network interfaces to be assigned IP address by the orchestrator.
  - The CNI configuration of all NetworkAttachmentDefinitions (NADs) created by the Orchestrator to use an IPAM plugin.

Attribute	Qualifier	Cardinality	Content	Description
description	M	1	String	Describes the service exposed.
virtualLinkRequirement	M	1	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.
inputParamsMappings	M	1	datatype.extCpd.ParamMappings	Information for the orchestrator on what parameters need be provisioned, and the mapping of the information to the orchestration templates used by the CNF (helm charts, terraform config files, etc.).

NOTE 1: Corresponds more or less to a virtual_link requirement in ETSI NFV SOL001.

Description proposal attributes

asdExtCpd, Attributes of the datatype.ExtCpd.ParamMappings

Attribute	Qualifier	Cardinality	Content	Description
ipAddressParameter	M	0..1	String[Note1]	When present, it indicates the name of the parameter that will hold the IP address attributed to this interface. This parameter and its value will be passed to the deployment tool when deploying all the *DeploymentArtifacts.*
nadName	M	0..1	String[Note1]	When present, these attributes indicate~~s~~ for a secondary network interface the parameters that will hold the name of the NetworkAttachmentDefinition used by the network interface, respectively the name of the Namespace the NetworkAttachmentDefinition resource resides in. These parameters and their corresponding values will be passed to the deployment tool when deploying the *DeploymentArtifacts* in order to configure the network interface with a customized NetworkAttachmentDefinition.
nadNamespace	M	0..1	String[Note1]	

Note1: 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, and it is as follows: "*helmchartname:[subchartname.]*$^{0..N}$*[parentparamname.]*$^{0..N}$*parametername*".
Whether the optional parts are present depends on how the parameter is declared in the helm chart.

Description proposal attributes

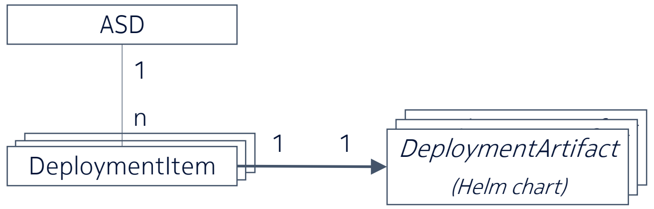
enhancedClusterCapabilities

Attribute	Qualifier	Cardinality	Content	Description
Id	M	1	String	Asd local unique name for the enhanceClusterCapabilities instance
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..1	List of strings	Required kernel modules are coded as listed by linux lsmod command, e.g. ip6_tables, cryptd, nf_nat etc.
conflictingKernelModules	M	0..1	List of strings	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 would conflict with use of proprietary user space SCTP stack provided by the application.
requiredCRDs	M	0..1	List of strings	List the required CRDs and their versions in the target environment. The list shall include those CRDs which are not delivered with the application. Example: Redis CRD, version 5.0.
clusterLabels	M	0..1	List of strings	This attribute allows to associate arbitrary labels to clusters. These can indicate special infrastructure capabilities (e.g., NW acceleration, GPGPU compute, etc.). The intent of these labels is to serve as a set of values that can help in application placement decisions. This can be specified with the attribute -m: Mandatory, means deployment is not attempted if such support is not available in the target system -p: As preference - it means orchestrator will try to select a system with specific requirements, but if not found it will attempt deployment in a system not having such HW.
secondaryInterfacePlugin	M	0..1	String	The plug-in name / revision of the operator to handle secondary interface (e.g. Multus-CNI, v3.8)

Description proposal attributes

deploymentItem

As an alternative to "parent" or "umbrella" charts, we propose a means to structure and sequence multiple Helm charts that is easily expressed inside the ASD, using the DeploymentItem construct



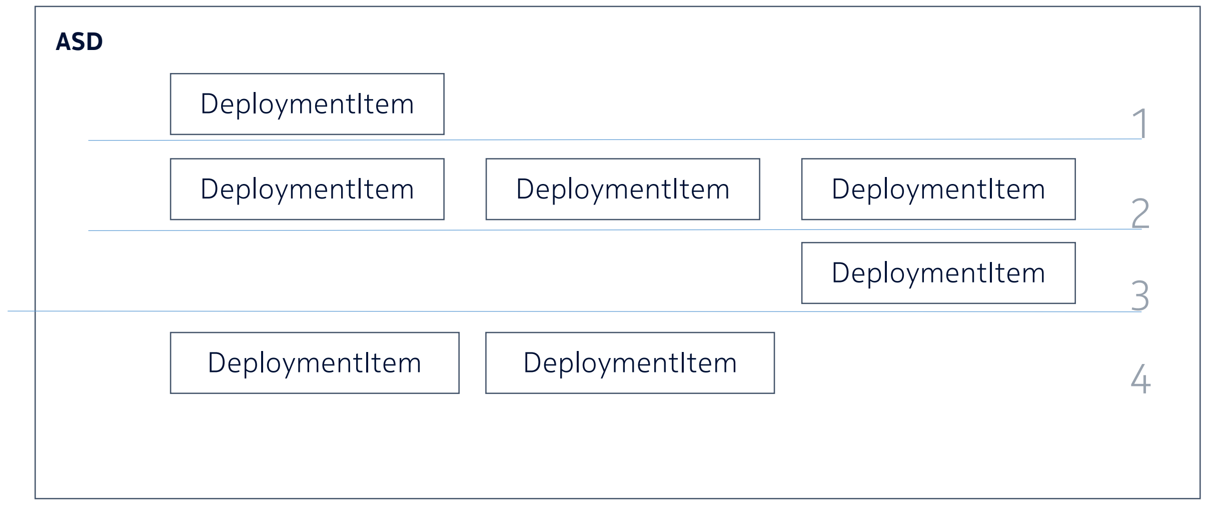
Attribute	Qualifier	Cardinality	Content	Description
deploymentItemId	M	1	Identifier	The identifier of this deployment item
artifactType	M	1	String/enum	Specify artifact type. e.g. Helm chart, helmfile, CRD etc.
artifactId	M	1	Identifier (reference to)	Reference to a Deployment artifact
deploymentOrder	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.
lifecycleParameters	M	0..N	List of strings	List of parameters that can be overridden at deployment time (e.g. values for values.yaml in the chart this item references)

Description proposal attributes

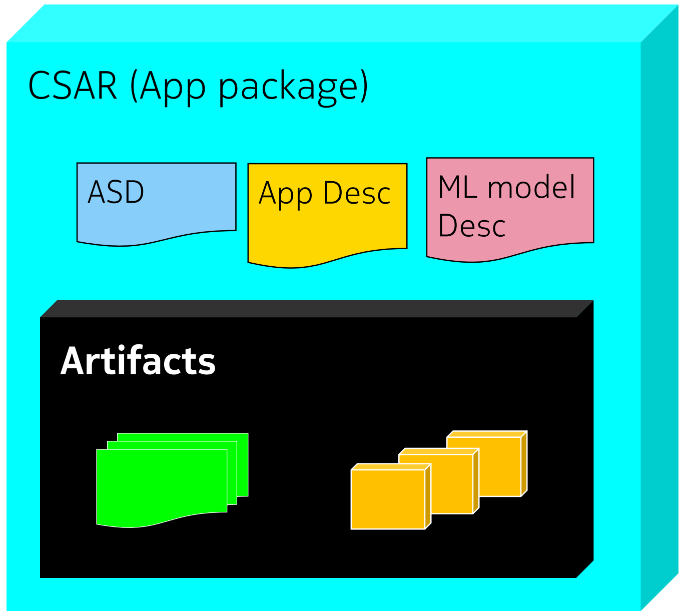
deploymenOrder

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 sequence.



Leveraging ETSI-compliant packaging



The proposal for packaging is to continue to rely on CSAR.

Recommend to standardize:

a) The Packaging of containerized applications, the ASD itself, and the usage of ASD as a top-level artifact for a CSAR.

b) The use of Helm as the cloud-native artifact for describing (pieces of) the application

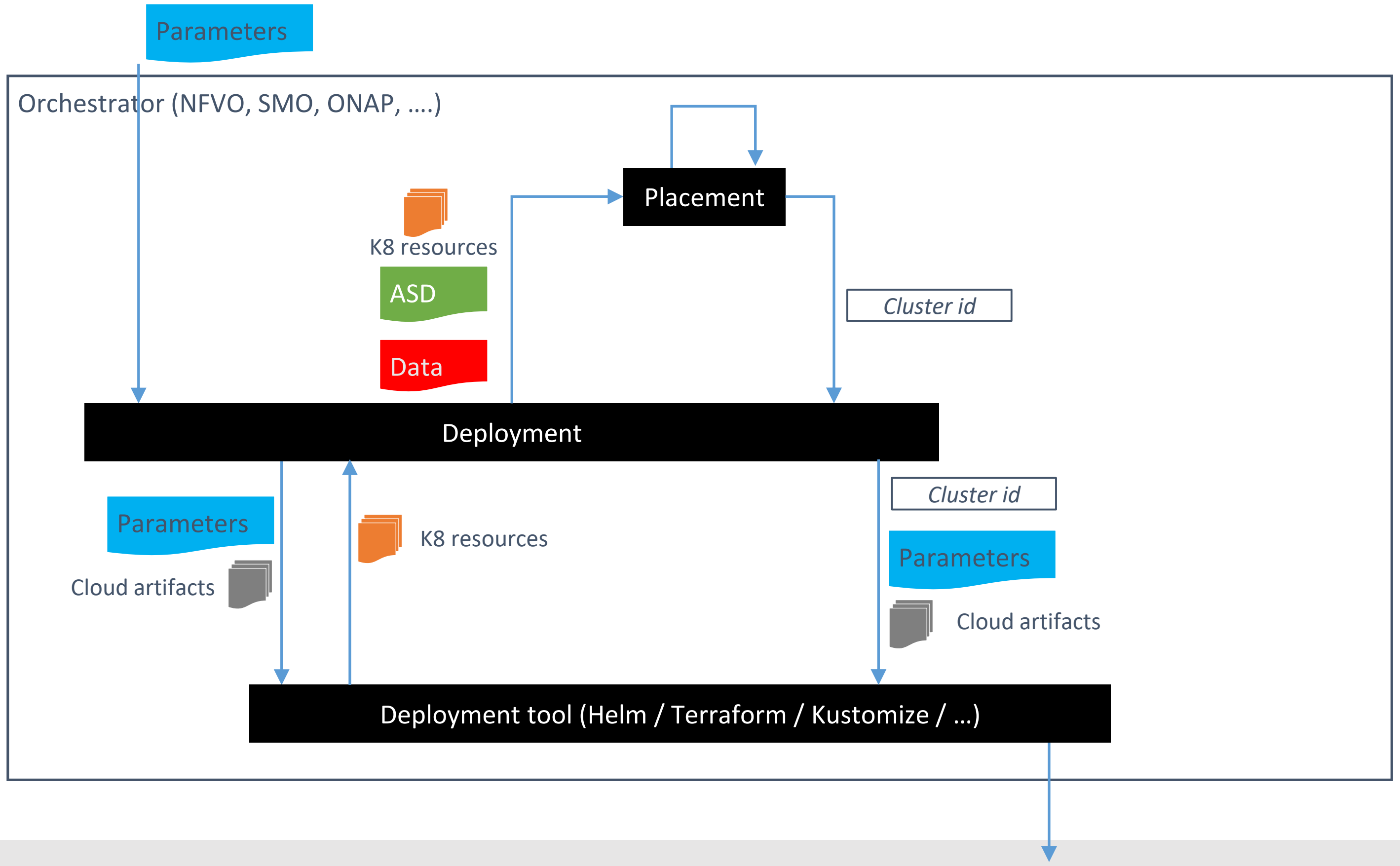
c) Adopt SOL004, adding the possibility to include an ASD instead of a VNFD as a top-level artifact

SMO Orchestration Capabilities

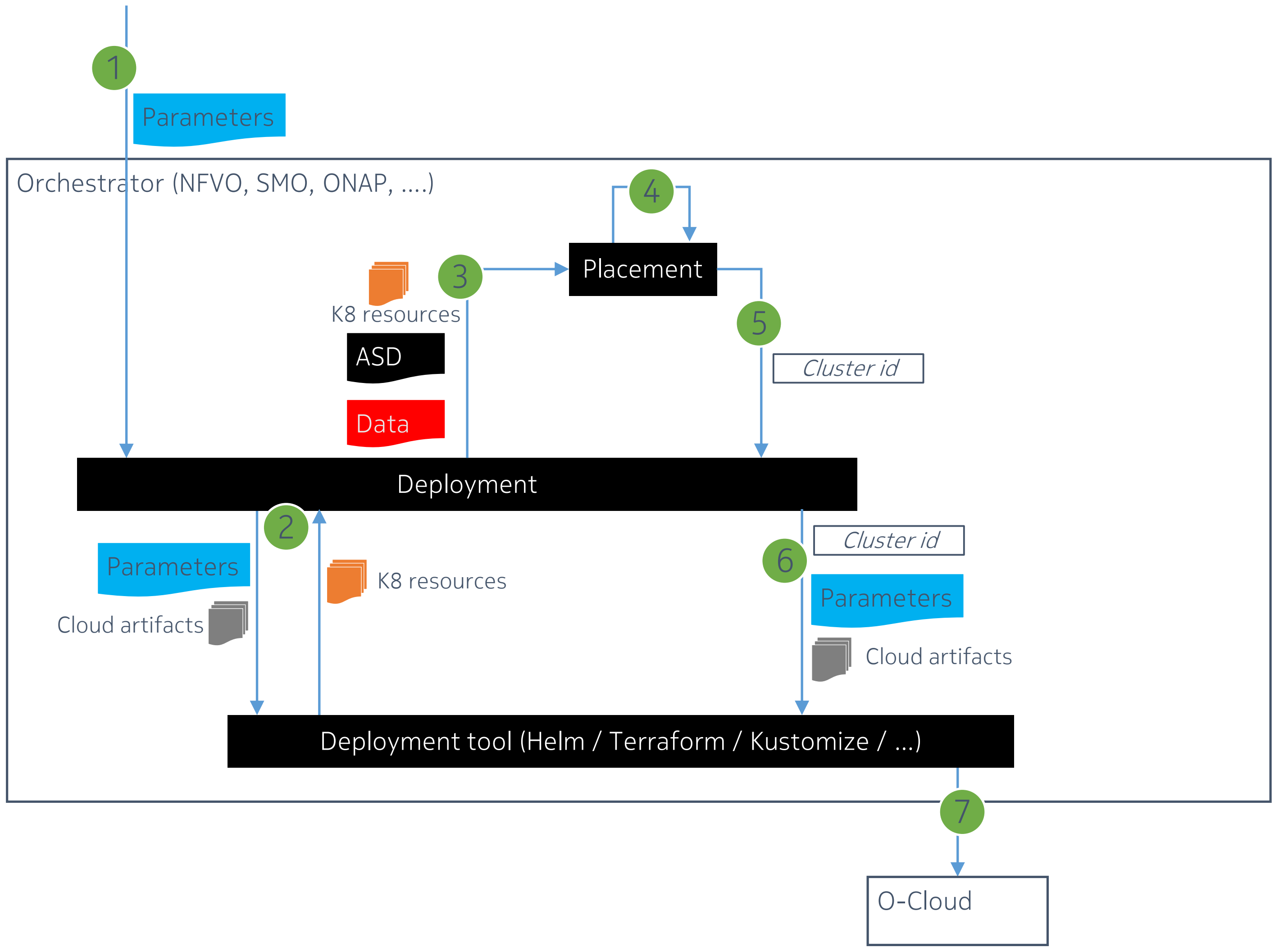
Non-Normative example of how O-RAN SMO capabilities might be structured

Functional block description:

- Deployment: this function is in charge of coordinating the deployment of an application through the DMS interface. It receives deployment requests, interprets deployment package descriptors, delegates placement and interacts with cloud deployment tools.
- Deployment tool: one or more commonly used cloud packaging / deployment tools, like Helm, Terraform, Kustomize, Helmfile…
- Placement: this function is responsible for selecting an appropriate cluster in the available O-Cloud resource pools. The placement criteria is up to each implementor but has to respect the restrictions indicated in the deployment descriptors.



Example flow of application deployment

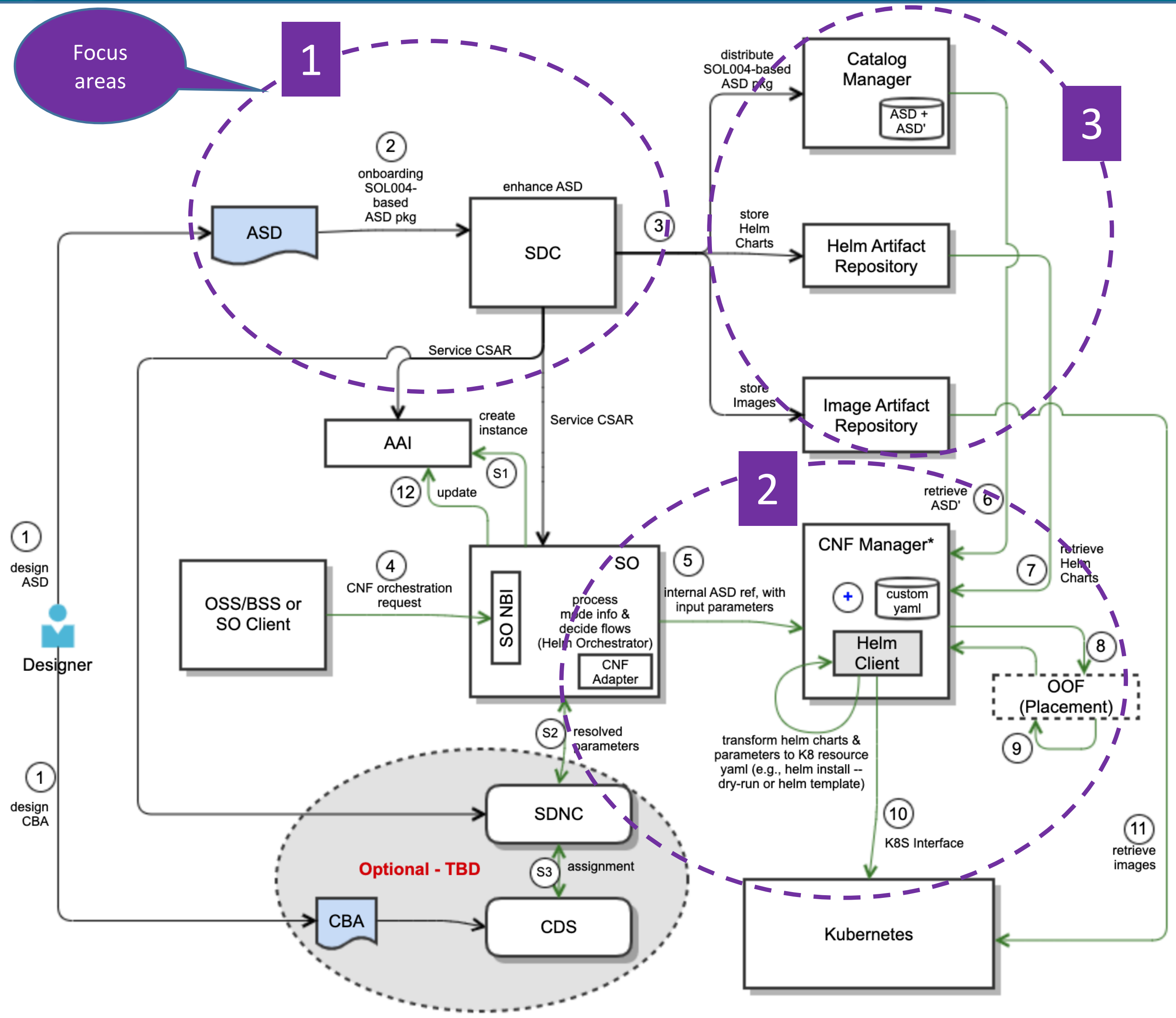


1. A deployment order is received, along with the required lifecycleParameters values
2. The cloud-native deployment tool is invoked with the received parameters to transform the cloud artifacts into K8S resource descriptions.
3. The K8S resource descriptions, ASD and any other relevant data is sent 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.

Part 3: ONAP ASD PoC

ASD Onboarding, Distribution and Orchestration PoC

- Target: Jakarta Release (Design is in progress)



#	Actor	Action
1	Designer	•Design ASD and/or CBA
2	Designer / SDC	•Onboard SOL004-based ASD package (could include CBA) •Design Resource & Compose Resources into Service •Enhance ASD (ASD')
3	SDC	•Distribute Service CSAR to ONAP runtime components (SO, AAI, SDNC, etc.) •Distribute ASD + ASD' to Catalog Manager •Distribute Helm Charts to Helm Repository (centralized) •Distribute Images to Image Repository (centralized)
4	SO Client	•Start CNF service creation
5	SO	•S1: create instance to AAI •S2: resolve parameters •S3: CDS returns assignment •Process model & decide flows (in this case, Helm Orchestrator) •Delegate Resource-level orchestration to Helm Orchestrator •Pass ASD' or its reference with input (resolved) parameters
6	CNF Manager	•Retrieve ASD' (and/or ASD : TBD) from Catalog Manager
7	CNF Manager	•Retrieve Helm Charts from Helm Repository
	CNF Manager - internal	•Transform ASD cloud artifacts with parameters to K8S resource description (e.g., helm template or helm install --dry-run) •Get additional data as needed
8	CNF Manager	•Get placement information by passing K8 resources + ASD' + additional data to the Placement
9	Placement	•make a placement decision based on input data & return it
10	CNF Manager	•Send CNF creation request (with cluster id, parameter, cloud artifacts) to K8S (e.g., helm install ...)
11	k8S	•Retrieve images from Image Repository and process for CNF
12	SO	•Update instance to AAI

backup

Description proposal attributes

asdExtCpd

Attribute	Qualifier	Cardinality	Content	Description
description	M	1	String	Describes the service exposed.
virtualLinkRequirement	M	1	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.
interfaceOrder	M	0..1	Integer, greater or equal to zero	Mandatory attribute for a secondary network interface (not applicable for a primary network interface). Defines the order in which an the additional/secondary network interface declaration appears in the pod manifest. Note that an SRIOV mated vNIC pair shall be modelled by a single asdExtCpd, and its order value set to the lower value of the two vNIC's order numbers; the two vNICs are expected to appear in consecutive order on the compute instance, and be attached to same network(s). See Note 2
networkInterfaceRequirements	M	0..1	NetworkInterfaceRequirements	Details container implementation specific requirements on the NetworkAttachmentDefinition ~~to~~ . See Note 2 & 3.
inputParamMappings	M	0..1	extCpd.ParamMappings	Information on what parameters that are required to be provided to the deployment tools for the asdExtCpd instance.
resourceMapping	M	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.

Description proposal attributes

asdExtCpd, NetworkInterfaceRequirements IE

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	**”cniManaged” | ”userManaged” | ”inBand”**	The default value (”cniManaged”) means that the CNI specifies how IPAM is done and assigns the IP address to the pod interface. Value ”user” indicates that IPAM is done via the application inside the pod. "inBand" indicates that the application expects to receive the interface configuration through protocols/procedures over the interface itself, such as DHCP, DHCPv6, SLAAC.
interfaceType	M	0..1	**”kernel” | ”userspace” | “sriov"**	This attribute describes the type of network interface the container implementation expects. *NOTE: For further study if the attribute need to be extended to be able to specify that an application expects a specific type of kernel interface, such as virtual-netdev or eth-netdev, a specific userspace interface like memif or virtio-user, or a pci-device based sriov interface.*
interfaceOption	M	0..N	e.g. "speed=1G|10G|25G|100G" "nic-type=virtio|i710|mlx-cx5|"	Applicable to Pod network interfaces that directly connect to a physical NIC. The value is a list of verified options for physical NIC caracteristics.
networkRedundancy	M	0..1	**"infraProvided“ | "none“ | "matedPair"**	Default value is "infra-provided”, which means that the infrastructure is expected to provide network redundancy for the pod interface. Value "none" means that the application has no requirement on network redundancy. Value ”matedPair” means that the Pod asks for a mated pair of non-redundant left/right network attachments (typically SRIOV) and handles redundancy on application level. The same set of networks shall be configured on both interfaces.
switchPlane	M	0..1	**”Left” | ”Right”**	Used (only) in conjunction with redundancyMethod "none" (see the NetworkInterfaceRequirements), when the application requires two independent virtual links that for redundancy reasons have to reside on different switch planes (left or right). See Note 2.
redundancyMethod	M	0..1	**”activePassiveBond” | ”activeActiveBond” | ”activePassiveL3” | ”activeActiveL3”**	Used (only) when specifying networkRedundancy "matedPair". "*activeActiveBond*“: The bonded left/right links must be part of a multi-chassis LAG in active-active mode. | "*activePassiveBond*": Interfaces bonded in active-passive mode in the application with move of bond MAC address. No specific requirements on DC fabric. | "*activePassiveL3*": Move of application IP address. | "*activeActiveL3*": Anycast/ECMP.

Best practice when operating with deploymentItems, artifactType="helm"

Deployment specific values, lifecycleParameters, use of values.schema.json, customer parameters

- In case of helm deployment item, in addition to / instead of lifecycleParameters use values.schema.json helm v3 capability to define and validate content of values.yaml
- Include values.schema.json as part of package to automatically validate the values.yaml before sending to K8S for deployment.
- Customer parameters
  - All parameters that are for internal use as well as external use shall be present in values.schema.json file with indication of what parameters are to be set by service designer/the customer. values.schema.json is provided by vendor and it's immutable.
  - In values.schema.json, tag **"value-use": "external"** indicates parameters which can be/ intended to be modified by the service designer / customer.
  - Similarly, tag with **"value-use": "internal",** indicates parameters which are set by vendor and shall not be modified by customer.

```
....

"timezone": {

        "description": "Timezone setting. Timezone values
are the ones defined in the IANA Timezone Database.",

        "value-use": "external",

        "type": "string",

        "default": "UTC"

}

.....
```

Orchestration Scenarios

- Meeting Information:
  - Weekly meeting: Mondays at 1200 UTC, 5AM PDT, 8AM EDT, 1300 CEST, 5:30 PM India, 8PM China.
  - https://zoom.us/j/722438866?pwd=d3VTTFJiK3Bkeml2RlRqVkdhSlhKQT09
  - One tap mobile: +16699006833,,722438866# US (San Jose) +16465588656,,722438866# US (New York)

ONAP

OPEN NETWORK AUTOMATION PLATFORM

NOKIA