Design-Time Data Model: 01. Quick Start

ONAP DT DM by example..

Work in progress..

ONAP Data Model Normatives

```
###################################
### ONAP Data Model Normatives ###
###################################

data_types:

interface_types:
  onap.interfaces.node.lifecycle.Standard:
    derived_from: tosca.interfaces.node.lifecycle.Standard
    description: the ONAP resource lifecycle interface, in case it extends the standard TOSCA's
    # here come the extensions

  onap.interfaces.node.lifecycle.VNF:
    # VNFs may need an extended lifecycle interface

  onap.interfaces.node.lifecycle.Service:
    # Services may need an extended lifecycle interface

capability_types:
  onap.capabilities.Compute:
    # a derivation of a TOSCA normative capability type

  onap.capabilities.Storage:
    # a derivation of a TOSCA normative capability type

policy_types:
  onap.policies.scaling.Fixed:
    # ....

  onap.policies.scaling.Variable:
    # ....

  onap.policies.placement.Affinity:
    # ....

  onap.policies.placement.AntiAffinity:
    # ....

  onap.policies.naming.NumSequence:
    # ....


node_types:
  #TODO: provide a description of the metadata for the node templates
  onap.nodes.Resource:
    description: |
      a base of the ONAP hierarchy of resources
    derived_from: tosca.nodes.Root
    requirements:
      - host:
          description: |
            An ONAP resource may be hosted by a TOSCA container.
            In a VDU, this requirement is of the onap.capabilities.Compute type
          capability: tosca.capabilities.Container
          occurrences: [0, 1]
          relationship: onap.relationship.HostedOn

onap.nodes.Function:
  derived_from: onap.nodes.Resource
  description: |
```

```
    a virtual function (VNF, VDU, VFC)
  properties:
    # all IM properties
    # all ECOMP VNF properties
  requirements:
  # 1. MANY requirements of type onap.capabilities.Compute (a summary of
  #    the Compute requirements of all inner Container and VDU nodes
  #    exposed through the substitution mapping)
  # 2. MANY requirements of type onap.capabilities.Linkable (a summary of
  #    the Linkable requirements of all ExtCPs inside the topology
  #    exposed through the substitution mapping)
  # 3. MANY OTHER application-level reqs&caps


  onap.nodes.VDU:
    derived_from: onap.nodes.Resource
    description: |
      represents a virtualization container at the infrastructure level;
      contains the software image,
      declares [required] hardware capabilities
    capabilities:
      host:
        type: tosca.capabilities.Container
        occurrences: [0, UNBOUNDED]
      compute:
        type: onap.capabilities.Compute
        occurrences: [0, UNBOUNDED]
      storage:
        type: onap.capabilities.Storage
        occurrences: [0, UNBOUNDED]
```

Sample VNF

```
###################################
### Sample VNF                  ###
###################################

node_types:
  com.vendorxxx.SampleVNF:
    derived_from: onap.nodes.VNF
    description: a concrete VNF provided by a vendor
    properties:
      num_of_instances_inside:
        type: integer
    requirements:
      - compute_1:
          type: onap.capabilities.Compute
      - storage_1:
          type: onap.capabilities.Storage
    capabilities:
      the_important_capability:
        #...

topology_template:
  inputs:
    num_resource_instances:
      description: how many resource instances to create
      type: integer

  node_templates:
    vl_1:

    cp_1:

    internal_valuable_resource_1:
      type: com.vendorxxx.ResourceType
      artifacts:
        image: ResourceDockerFile
```

```
    capabilities:
      valuable_capability: #....
    requirements:
      container:
        node: vdu_1
        capability: container
  vdu_1:
    type: onap.nodes.VDU
    artifacts:
      image: myImageFile.ovf
    capabilities:
      container:
      compute:
        num_cpus: 2
        mem_size: 1GB

policies:
  scale_the_value:
    type: onap.policies.scaling.Fixed:
    properties:
      quantity: {get_input: num_resource_instances}
    targets: [internal_valuable_resource_1]

  separate_hosts:
    type: onap.policies.placement.AntiAffinity
    targets: [internal_valuable_resource_1]
    properties:
      distance: PhysicalHost

  same_office:
    type: onap.policies.placement.Affinity
    targets: [internal_valuable_resource_1]
    properties:
      scope: DataCenter

substitution_mappings:
  node_type: com.vendorxxx.SampleVNF
  properties:
    num_of_instances_inside:
      mapping: [SELF, num_resource_instances]  # a mapping to an input
  capabilities:
    the_important_capability:
      mapping: [internal_valuable_resource_1, valuable_capability]
    compute_1:
      mapping: [vdu_1, compute]
    storage_1:
      mapping: [vdu_1, storage]
```

Service using the Vendor VNF

```
#######################################
### A Sample Service using the Sample VNF ###
#######################################

topology_template:
  node_templates:
    vnf_1:
      type: com.vendorxxx.SampleVNF
      properties:
        num_of_instances_inside: 13
      capabilities:  # infrastructure requirements, to be satisfied by the orchestrator
        - compute_1:
            capabilities: onap.capabilities.Compute
        - storage_1:
            capabilities: onap.capabilities.Storage
```

Points to emphasize:

1. Scaling expressed through policies
2. Affinity/anti-affinity expressed through policies
3. VDU node type to model a virtual container (VM, Docker container):
    a. States quantified requirements for infrastructure:
        i. Generic requirements: computational power, storage volumes,
        ii. Requirements for specific hardware: Intel's, AMD, etc.
    b. Includes a software image used to initialize the container - as a TOSCA artifact
    c. In order to specify hardware/infrastructure requirements for a resource, the designer creates a VDU node in the topology template and then creates a relationship between the resource node and the VDU node using the tosca.capabilities.Container capability-requirement pair; multiple resources may share a VDU
    d. The requirements for hardware/infrastructure specified by the VDU nodes across the model will be satisfied on instantiation by the orchestrator.
4. The VNFD element of the Information Model is modelled through a combination of the following TOSCA constructs:
    a. A TOSCA node type – the "interface" part of the definition: derived from the onap.nodes.VNF node type (and, consequently, from the basic onap.nodes.Resource); exposes the important properties, capabilities, requirements
    b. A TOSCA topology template – the "implementation" part of the definition: the internal topology of component resources and policies
    c. A TOSCA substitution mapping construct that wires the interface to the implementation: interface properties are mapped to the implementation topology inputs, interface capabilities and requirements – to those of the component resource nodes
5. An occurrence of the VNF in a higher-level topology (a service or a "higher" VNF) is modelled as a TOSCA node template of the VNFD "interface" node type, with its properties populated and requirements and capabilities involved into relationships with the neighbor nodes.

See also: ECOMP SDC Metadata Overview, Affinity and AntiAffinity, Splitting VDU: VFC + Container, Scaling