TOSCA Enhancements to Support Multiple Node Template Instances

March 23, 2018

# Introduction

TOSCA service templates specify a set of nodes that need to be created at service deployment time. Some service templates may include multiple nodes that perform the same role. For example, a template that models an SD-WAN service might contain multiple VPN Site nodes, one for each location that accesses the SD-WAN. Rather than having to create a separate service template for each possible number of VPN sites, it would be preferable to have a single service template that allows the number of VPN sites to be specified at deployment time.

This document proposes TOSCA extensions in support of this functionality.

# SD-WAN Example

The following figure shows an example SD-WAN deployment to three sites:

VPN

San Jose

Austin

Boston

The SD-WAN service was deployed using a service template that contains VPN and Site nodes as shown in the following figure:

The following code snippet shows a TOSCA service template from which this service could have been deployed:

|  |
| --- |
| tosca\_definitions\_version: tosca\_simple\_yaml\_1\_2description: Template for deploying SD-WAN with three sites.topology\_template: inputs: location1: type: Location location2: type: Location location3: type: Location node\_templates: sdwan: type: VPN site1: type: VPNSite properties: location: { get\_input: location1 } requirements:* vpn: sdwan

 site2: type: VPNSite properties: location: { get\_input: location2 } requirements:* vpn: sdwan

 site3: type: VPNSite properties: location: { get\_input: location3 } requirements:* vpn: sdwan
 |

Unfortunately, this template can only be used to deploy an SD-WAN with three sites. To deploy a different number of sites, additional service templates must be created, one for each number of possible SD-WAN sites. This leads to template proliferation, which is undesirable. The next section explores alternatives.

# Specifying Number of Occurrences

To avoid the need for multiple service templates, TOSCA needs to be extended with a mechanism that allows all VPN Site nodes to be created from the same Site node template in the topology, and allow the number of sites to be specified at deployment time. Specifically, this functionality must:

* Allow service template designers to specify that multiple node instances can be created from a single node template
* Allow service template designers to constrain how many node instances can be created from a single node template
* Allow users to specify at deployment time the exact number of instances that need to be created from the single node template.

The following extensions to the TOSCA node template definition grammar could be considered to provide this functionality:

| Keyname | Required | Type | Constraints | Description |
| --- | --- | --- | --- | --- |
| occurrences | no | range of integer | when not specified, defaults to [1,1] | The optional minimum and maximum number of instances that can be created from this node template. If not specified, only one single instance can be created. |
| instance\_count | no | integer | when not specified, defaults to the lower bound of the range specified by the ‘occurrences’ keyname | The requested number of runtime instance of the node template. |

These extensions enable the creation of a simplified SD-WAN service template that contains only one single VPN Site node as shown in the following figure:

The following code snippet shows the updated TOSCA service template that uses the new grammar:

|  |
| --- |
| tosca\_definitions\_version: tosca\_simple\_yaml\_1\_2description: Template for deploying SD-WAN with a variable number of sites.topology\_template: inputs: numberOfSites: type: integer  node\_templates: sdwan: type: VPN site: type: VPNSite occurrences: [1, UNBOUNDED] instance\_count: { get\_input: numberOfSites } requirements:* vpn: sdwan
 |

# Specifying Inputs

The service template in the previous section conveniently ignores the location property of the Site node. As shown earlier, the location property is expected to be provided as an input value. If Site node templates can be instantiated multiple times, then it follows that multiple input values are required to initialize the location property for each of the Site node instances.

To allow specific input values to be matched with specific node template instances, a new reserved keyword is introduced:

| Keyword | Valid Contexts | Description |
| --- | --- | --- |
| INDEX | Node Template | A TOSCA orchestrator will interpret this keyword as the runtime index in the list of node instances created from a single Node Template. |

The following service template shows how the INDEX keyword is used to retrieve specific values from a list of input values in a service template:

|  |
| --- |
| tosca\_definitions\_version: tosca\_simple\_yaml\_1\_2description: Template for deploying SD-WAN with a variable number of sites.topology\_template: inputs: numberOfSites: type: integer locations: type: list entry\_schema: Location  node\_templates: sdwan: type: VPN site: type: VPNSite occurrences: [1, UNBOUNDED] instance\_count: { get\_input: numberOfSites } properties: location: { get\_input: [ locations, INDEX ] } requirements:* vpn: sdwan
 |

*This approach requires that inputs are provided as lists, even if only one instance will be created. Should we allow a single value as well as a list of values?*

*Do nodes keep the same index for their entire lifecycle? What if one instance is deleted at runtime. Do all other nodes keep their index? What if a new node instance is created at runtime? Does it get the next available index at the end of the list, or does it try to fill any holes in the list?*

# Cardinality of Relationships

When multiple node instances can be created from the same node template, the service template designer must also have control over the cardinality of relationships established to and from those nodes. Specific mechanisms depend on the type of the relationship.

## Many-to-One Relationships

In the SD-WAN service template above, each of the site node instances has a relationship to a VPN node that can only be instantiated once. This is an example of a *many-to-one* relationship, which can be readily supported using existing relationship syntax. Of course, care must be taken to ensure that the capability in the single node that terminates the relationships allows multiple occurrences.

## One-to-Many Relationships

*One-to-many* relationships are less common, but they can just as easily be accommodated using existing TOSCA grammar, as long as the requirement in the single node accommodates multiple occurrences.

## Matched Pairs

For some services, instances of different node templates that have a relationship between them must remain matched up in pairs. For example, let’s extend the SD-WAN service above with a third node template that represents a virtual PE router that must be used at each site to establish VPN connections over an underlay. Let’s assume that Site nodes establish a HostedOn relationship to the vPE nodes. The extended service topology is shown in the following figure:

In this example, the intent is for each site node to remain paired with its own vPE node for that site. To accomplish this, the service template designer must first make sure that the number of vPE instances is the same as the number of Site instance by using the same input value for the count keynames in both the vpe and site node templates. Next, a simple TOSCA extension must be introduced to allow each site node instance to identify its own target vPE node instance in its host requirement. The following TOSCA template shows how the node INDEX can be used for this purpose:

|  |
| --- |
| tosca\_definitions\_version: tosca\_simple\_yaml\_1\_2description: Template for deploying SD-WAN with a variable number of sites.topology\_template: inputs: numberOfSites: type: integer locations: type: list entry\_schema: Location  node\_templates: sdwan: type: VPN vpe: type: vPE occurrences: [1, UNBOUNDED] instance\_count: { get\_input: numberOfSites } site: type: VPNSite occurrences: [1, UNBOUNDED] instance\_count: { get\_input: numberOfSites } properties: location: { get\_input: [ locations, INDEX ] } requirements:* vpn: sdwan
* vpe: [ vpe, INDEX ]
 |

## Many-to-Many Relationships

We may also need to accommodate scenarios where a node template that can have multiple instances has a relationship to another node template that can also have multiple instances, but the number of instances for each node differs. Those scenarios do not support *matched pair* relationships. Additional mechanisms may be required to specify how the cardinality of the relationship is affected. The intended cardinality depends on the following questions:

* Are relationships established from each instance of one node to all instances of the other node in full-mesh mode?
* Are relationships established from each instance of one node to only one instance of the other node? If so, which one?
* Can relationships be established from each instance of one node to a subset of the instances of the other node? If so, which ones?

These scenarios need to be explored further and will likely require additional extensions to the TOSCA grammar.

# Propagating Properties

If node template A supports multiple occurrences, and node template B uses a

{ get\_property, [A, <property\_of\_A> ] }

then it should be possible to specify which specific instance of A will be used in the corresponding property value of B.

# Conclusion

A number of extensions to the TOSCA grammar are required to support the creation of multiple node instances from the same node templates. More exploration is required to identify language requirements to uniquely identify relationships between such node instances.