White Paper

Deploying Virtual Network Functions: The Complementary Roles of TOSCA & NETCONF/YANG

Prepared by

Caroline Chappell
Principal Analyst, Cloud & NFV, Heavy Reading
www.heavyreading.com

on behalf of

www.cisco.com www.alcatel-lucent.com

February 2015

Executive Summary

The agile and automated management of virtualized network functions (VNFs) throughout their lifecycles is a key goal for network functions virtualization (NFV). This white paper focuses on the first phase of the VNF lifecycle: the agile deployment and fulfillment of VNFs in the cloud. It explores the complementary roles played by cloud deployment templates, such as Topology and Orchestration Specification for Cloud Applications (TOSCA), and the network data modeling language, YANG, and its associated configuration application programming interface (API), NETCONF.

Heavy Reading research consistently confirms that service agility is the top driver for NFV adoption, closely followed by operational cost reduction. Network operators want to be able to launch network services more rapidly and with much lower cost than they can today.

If the network functions that services depend upon are virtualized, they can be deployed very quickly, potentially anywhere in the network, without the delays associated with physical hardware procurement and installation. When VNFs are spun up on top of cloud infrastructure, they take advantage of the instant availability of virtual resources.

In addition, VNFs can be instantiated using cloud application lifecycle management techniques that automate deployment, reducing operational cost and time-to-launch. NFV taps into the agile application lifecycle management opportunity that cloud delivers to achieve operators' service agility goals.

However, VNFs need further configuration at runtime to fulfill customer-specific services. A cloud deployment template can help to turn up a new VNF instance so that it is operationally ready, but such a template does not have an API that can configure the VNF dynamically, on demand. This is where runtime configuration approaches are needed to complete the fulfillment process. NETCONF/YANG is an example of one such approach that delivers a programmatic configuration experience that can be harnessed to support the automated lifecycle management goals of NFV.

Section II describes the principles behind cloud application deployment templates and explains why TOSCA is becoming a leading candidate deployment template specification.

Section III explores why TOSCA is not sufficient in the virtualized network and analyzes how specifications such as TOSCA and runtime interfaces such as NETCONF/YANG will coexist.

Application Deployment in the Cloud

In the cloud, an application's installation and startup requirements – its basic "operations" – are modeled in advance by the application developer using a method known as "DevOps." DevOps emphasizes the interdependence of an application's development and its IT operations. This is in contrast to traditional development models, where applications are "thrown over the wall" to operations staff who work out how and where physically to run them. Since the operational environment is predefined in the cloud – virtual resources are already specified and ready to be launched – it is easier for application developers to take responsibility for operationalizing their own applications.

A developer captures an application's operational requirements in an accompanying deployment template. The template is essentially a file describing all the artefacts needed to get the application up and running, and to scale and heal the application once it has been deployed. The template is "read" by a cloud management system, which orchestrates the implementation of all the artefacts it contains and sets up the virtual network connectivity between them. Orchestration is automated for speed and error-free deployment. The template approach ensures that cloud application deployment is repeatable and consistent on top of cloud infrastructure.

Cloud application deployment templates that are attempting to establish themselves as standards in the cloud environment include: TOSCA, from the Organization for the Advancement of Structured Information Standards (OASIS) standards body; Heat Orchestration Templates (HOT), under development by the OpenStack community; CloudFormation templates launched by Amazon Web Services; and Juju Charms from Canonical.

This paper focuses on TOSCA, but the principles of using a TOSCA template for the deployment phase of the VNF lifecycle equally apply to other types of cloud application deployment template. TOSCA templates can also be used for later lifecycle operations such as scaling, healing and software update.

There is growing market interest in TOSCA, since it currently supports more advanced concepts than Heat and can be used for application deployment across multiple types of cloud platforms, including, but not confined to, OpenStack-based clouds. HOT is just one of the template types to which TOSCA templates can be mapped.

This situation could, of course, change in the future, and the use of TOSCA templates does not yet guarantee perfect interoperability across cloud platforms and orchestration and management systems.

In an NFV context, VNFs also need associated deployment templates, called VNF descriptors (VNFDs) by the European Telecommunications Standards Institute (ETSI) NFV Industry Specification Group (ISG). These templates can also be used to describe network services that consist of two or more VNFs, with an optional "forwarding graph" that describes the network connectivity between them. In this case, the deployment template is referred to as a network service descriptor (NSD) in ETSI parlance.

The VNF Manager interprets the VNFD (deployment template), triggering the Virtualized Infrastructure Manager (VIM) to carry out virtual resource allocation actions in the NFV Infrastructure (NFVI). Deployment templates can be used to bring a VNF to a state of operational readiness, as defined by TM Forum standards.

Fulfilling VNFs

This paper discusses the separate but complementary roles of:

- **TOSCA** as a file-based template language for deploying VNFs on cloud infrastructure.
- **NETCONF/YANG** for subsequent fulfillment. NETCONF provides a runtime API both for configuring VNFs after they have been installed, bringing VNFs to a state of operational readiness, and while they are running in the cloud, fulfilling the service requirements of a particular customer(s).

Cloud Deployment Template Limits in the Network Environment

TOSCA operationalizes the deployment of VNFs and triggers their initial configuration. It provides a minimal set of operational functionality that needs to be preserved across application deployments in different cloud environments, for example:

- Install/uninstall/reinstall an application instance
- Start an application instance, or restart if it crashes or needs to be upgraded with new features/configurations
- Carry out initial "Day Zero" (operational readiness) configuration of the application instance

In the IT cloud world, applications are rarely reconfigured once they have been deployed. If IT application owners want a different configuration, they will create a new deployment template, "kill" the application instance with the unwanted configuration, and replace (reinstall) it, via the new template, with an instance with the new configuration.

In the network, VNFs may need further configuration at runtime, *after* they have reached the operational readiness stage. An operator may want to configure a VNF to fulfill new customer-specific requirements in addition to configurations specified in the original TOSCA template – for example, configure new customer-specific rules in a firewall.

A TOSCA template can install a virtual router, but it cannot subsequently create/modify/delete configuration on demand on the same router. TOSCA is not itself an API for the ongoing configuration of VNFs dynamically once they are executing. Instead, a TOSCA template specifies one or more configuration APIs that can be used during deployment for initial activation of a VNF.

TOSCA Template Components

TOSCA defines a service template, where a "service" for TOSCA is an application or application component, such as a database server, firewall or virtual router. In a networking context, the term "service" has very different connotations. It implies the customer-specific configuration of a network function, as we explain later on in this paper. Examples would be an individual Layer 3 VPN or customer-specific rules configured in the firewall.

The TOSCA template consists of two parts:

- **A topology template**, which in an NFV context describes the structure of the VNF (or network service) to be deployed and provides links to the artefacts needed to deploy it. Artefacts may include application images, application packages and package managers, VM image references and scripts. TOSCA uses XML today, but is moving to the encoding language, YAML, to model the software topology of a VNF – that is, the VNF's software structure and relationships between its building blocks, or "nodes." Different types of application nodes can be defined, for example, Web server or database server. YAML provides a compact, flexible and human-readable representation of the application instance that needs to be deployed. On the other hand, it does not have a strict schema.
- **Plans,** or workflows that describe the process of deploying a VNF in a complex scenario, where the VNF has many nodes with non-obvious dependencies. The relationships and sequence in which nodes in simple applications need to be deployed can typically be deduced from the topology template, but where this is not possible, the TOSCA template allows plans to be created in any workflow or deployment language, e.g., BPEL, Chef, Puppet.

The TOSCA service template and its constituent parts are encapsulated physically in a CSAR (Cloud Service ARchive) zipped file, Because TOSCA is a file structure, the CSAR can contain any artefact – a key advantage of TOSCA. However, there is no guarantee that the scope of these artefacts, the language they are written in, their packaging or any other aspect will be standardized across applications. Until appropriate tooling is built to handle the variety of artefact formats that will crop up in a CSAR file, the interoperability of TOSCA templates across vendor clouds and orchestration systems is limited.

Fulfilling VNFs Requires More Than TOSCA

A TOSCA template is a file-based description of VNF-related "things" and a set of execution guidelines for deploying and operating them as a single entity (VNF) on cloud infrastructure. Each TOSCA-defined node has interfaces through which it can be manipulated by an initial configuration application or script, which is referenced in the TOSCA template.

However, each VNF instance needs to be further configured at runtime to fulfill the specific needs of a customer and service. This is where the ETSI NFV architecture needs to be complemented by OSS functions for service activation. Runtime configuration is beyond the scope of a TOSCA template.

This paper focuses on NETCONF/YANG as a means of programmatically configuring a VNF instance once it is onboarded to the cloud. But there can be many runtime configuration actions to configure specific parameters or data on each VNF. These actions may require one or more management and/or data models and their associated protocols to interact with the VNF. Depending on the runtime configuration action required and what the VNF supports, the appropriate protocol/interface(s), such as NETCONF/YANG, TM Forum/3GPP/ITU-T/IETF-defined management or other interface standards, vendor-proprietary interfaces, Diameter, etc., needs to be applied to the VNF.

The Interplay Between TOSCA & NETCONF/YANG

The relationship between the OSS and the ETSI NFV management and orchestration (MANO) functions in supporting the service fulfillment process has yet to be fully

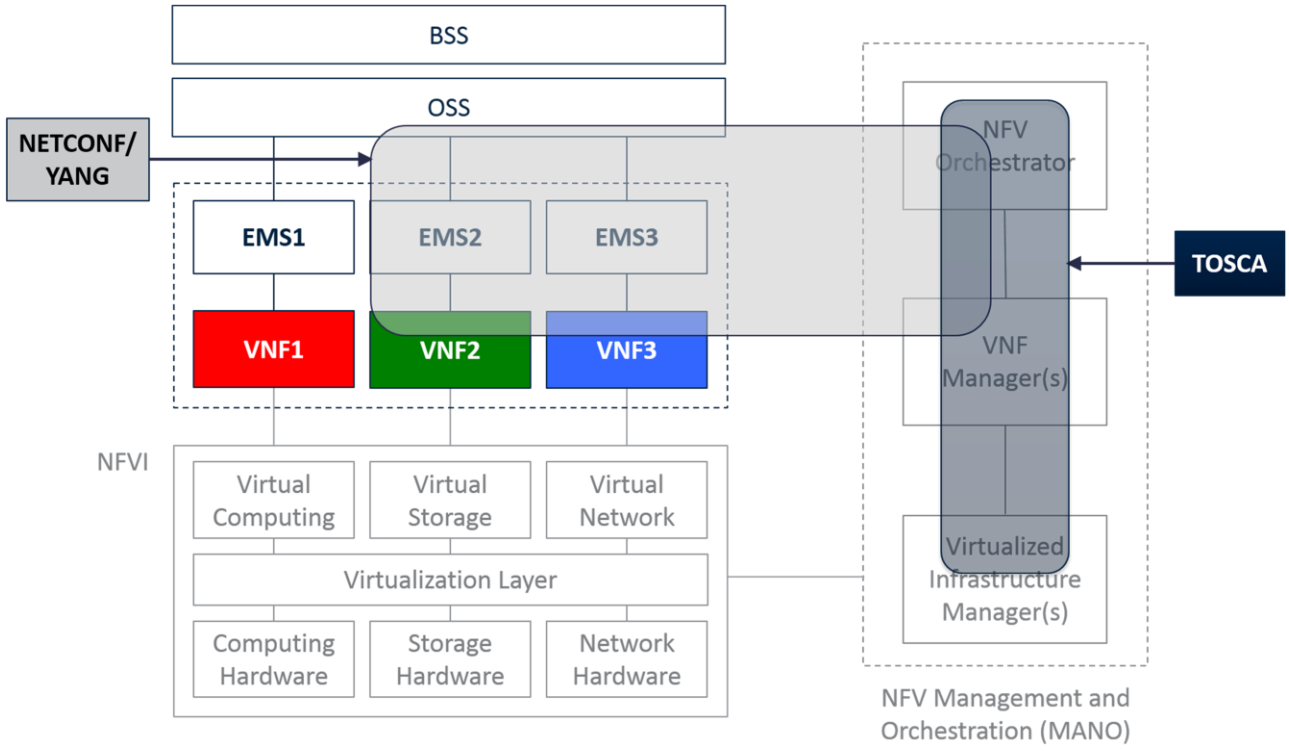
worked out. OSS functions will complement the MANO functions, NFV Orchestrator (NFV-O) and VNF Manager (VNFM), where needed to achieve service fulfillment.

Service fulfillment must be understood with reference to the different NFV MANO and OSS interpretations of the term "service lifecycle," which we recap here:

- In an NFV MANO context, the service lifecycle refers to maintaining VNFs in a healthy state and stitching them together to create "network services." For example, NFV MANO is responsible for spinning up new instances of a load balancer and firewall and connecting them together. This achieves an "operational readiness" state and is the domain of a deployment template, such as TOSCA.
- In the OSS view of the world, the term "service lifecycle" refers to the continuous recurrent reconfigurations of the VNFs due to service activation requests using an approach such as NETCONF/YANG.

Figure 1 shows where TOSCA and NETCONF/YANG play in the ETSI NFV Reference Architecture.

Figure 1: TOSCA & NETCONF/YANG Applicability



Source: Tail-F Systems

In a service fulfillment scenario, where new instances of VNFs need to be created, NFV MANO will be responsible for unpacking TOSCA templates to understand what resources the VNFs will need. Working together, the different orchestration layers of the NFV MANO stack will use the VNFs' deployment templates to create the required

infrastructure resources, install the VNFs' associated images on them and instantiate appropriate monitoring parameters.

A TOSCA deployment template may trigger NETCONF/YANG configuration during initial instantiation of a VNF. Both TOSCA and NETCONF/YANG may therefore be involved in this particular step of the NFV MANO service lifecycle management process.

Once the VNF instances are installed, the OSS takes over, working with individual VNFs or their corresponding element management systems (EMSs) to configure the VNF at runtime, for example, using NETCONF/YANG. An advantage of NETCONF/YANG is that the same approach can be used for runtime configuration of both virtual and physical network functions.

The details of the interplay between NFV MANO and OSS are likely to be worked out on an operator-by-operator basis, depending on each operators' vendor selections and process preferences.

The important point is that VNF deployment and configuration, both initially and at runtime, are carried out programmatically, using automated tooling. Automation is critical to achieving the low-cost, agile fulfillment model that operators are hoping for from NFV. TOSCA and NETCONF/YANG are complementary here because both contribute to an automated, "DevOps" approach to VNF lifecycle management.

Are Both NETCONF/YANG & TOSCA Necessary in an NFV Context?

There is currently some market confusion, compounded by the draft ETSI NFV MANO document, over cloud application deployment templates and YANG models. The YANG data modeling language can be used both to describe a deployable instance of a VNF and to configure that instance at runtime.

Some industry players believe that cloud deployment templates like TOSCA, which focus only on deployment and not on runtime configuration, are therefore unnecessary in an NFV context. Others believe that cloud deployment templates are sufficient, and that there is no need for NETCONF/YANG.

It is worth remembering that API-based data modeling languages such as YANG are designed for writing machine-readable schemas for network devices and resource-facing services (RFS, in TM Forum parlance, including NFV "network services," which are a virtual type of RFS). This is an essential requirement for programmatic runtime configuration.

Template languages focus on writing templates in a format that is easy for humans to read. TOSCA already provides a fit-for-purpose directory structure for deploying both IT applications and VNFs, so the same skillsets can cost-effectively be used to deploy either type of cloud workload. Using YANG models for initial VNF deployment would require the re-education of IT operations staff used to easy-to-read cloud deployment templates, which may very well be overkill.

NETCONF/YANG and cloud deployment template standards, of which TOSCA is one example, serve different purposes, and both contribute to the successful deployment and lifecycle management of VNFs.

Conclusion

In summary, TOSCA and NETCONF/YANG play complementary roles in supporting the deployment and runtime fulfillment of VNFs using an agile, automated "DevOps" approach. TOSCA is useful for its application-centric deployment model, which enables VNF instances to be reliably deployed over and over with the same initial set of configurations, whenever they are needed. A data model-driven configuration API such as NETCONF injects dynamism and flexibility into VNF fulfillment, enabling operators to carry short-lived, service-specific reconfigurations of VNFs at runtime.

Both TOSCA and NETCONF/YANG contribute to deployment agility by providing programmatic ways of deploying and then configuring applications on a cloud infrastructure, so that they are available for consumption quickly without incurring high levels of operational cost. Over the next few years, we expect standard VNF deployment patterns, and therefore VNF deployment templates, to emerge, making it easier to launch VNFs across different cloud platforms using a variety of vendor NFV MANO tools. We also expect growing numbers of network function owners to specify YANG data models for their products, supporting the programmatic configuration of network functions regardless of whether they are running on the cloud or embedded in physical appliances.

Operators that grasp early the differences between TOSCA and YANG – and where and how they complement each other – will be well-placed to optimize VNF deployment in support of NFV service agility goals.

Figure 2: A Comparison of TOSCA & YANG

	TOSCA	NETCONF/YANG
Organization	OASIS	IETF
Roots	IT, applications	Network services/devices
Purpose	Application-centric, lifecycle management of applications and their dependent artefacts in a data center. Design-time focus on what needs to be preserved and implemented across deployments in different environments, e.g., scaling, healing, upgrade.	Configuration of running network devices and network services. Runtime focus on what can be configured in a running deployment.
"Slogan"	Make them operationally ready and able to run well	Configure them
Example	Start CRM-system, SQL DB, IPTV server, network service	Provision a new IPTV service for customer ACME
Operations	Install, Start, Configure, Scale, Heal, Upgrade, Stop, Uninstall	Operations to configure things after Start and before Stop
Lifecycle Phase	Deployment and at runtime for specific functions (update, scaling, healing)	Runtime (service fulfilment)
Technologies	Topology template: application structure; Plans: process models to create and terminate (and manage) applications; Bundled in a physical CSAR file (zipped directory); Data model: YAML	Protocol: NETCONF; Data model: YANG

Source: Cisco/Tail-f Systems

About Cisco

Cisco (Nasdaq: CSCO) is the worldwide leader in IT that helps companies seize the opportunities of tomorrow by proving that amazing things can happen when you connect the previously unconnected. For ongoing news, please go to newsroom.cisco.com. Cisco and the Cisco logo are trademarks or registered trademarks of Cisco and/or its affiliates in the U.S. and other countries. A listing of Cisco's trademarks can be found at www.cisco.com/go/trademarks. Third-party trademarks mentioned are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company.

About Alcatel-Lucent

Alcatel-Lucent (Nasdaq: ALU) is the leading IP networking, ultra-broadband access and cloud technology specialist. We are dedicated to making global communications more innovative, sustainable and accessible for people, businesses and governments worldwide. Our mission is to invent and deliver trusted networks to help our customers unleash their value. Every success has its network. For more information, visit Alcatel-Lucent's website, read the latest posts on the Alcatel-Lucent blog and follow the company on Twitter: @alcatel_lucent.