A Day in the Life of a VNF

11 Nov 2015

Bryan Sullivan and Dan Druta, AT&T
Seeing the Forest

- OPNFV’s first two releases have been oriented on CI/CD and the NFVI
- It’s time to bring the bigger picture into focus for end-users
  - What can I do with the OPNFV platform and how do I do it, e.g. how does the OPNFV platform enable an end-to-end VNF / service lifecycle
- As NFVO/VNFM-scope projects are developing (e.g. SFC), we need to focus them on support of an overall lifecycle and on synergy with other projects at the same level
- With a common set of goals for the VNF / service lifecycle, we can use the maturing ETSI NFV MANO standards to shape the forest of functions growing as upstream projects
VNF and Service Lifecycle: Setting and Delivering on Expectations

VNF Provider
- Develop
  - Design
  - Develop
  - Test
- Deliver
  - Package
  - Validate
  - Accept and catalogue (onboard)
- Deploy
  - Combine
  - Assemble
  - Configure (software)
- Use
  - Service design
  - Configure (service)
  - Instantiate
- Manage
  - Monitor
  - Update
  - Upgrade
- Retire
  - Migrate Users

VNF Provider/Service provider (procurement)

Service provider (Service design)

Service provider (Service Delivery)

Service provider (Service Assurance)

User (Service Customization)

11 Nov 2015
OPNFV Summit
VNF and Service Lifecycle Modeling

• A completely modeled, declarative approach to setting VNF/service lifecycle expectations is a good goal

• We have some developing tools for this, e.g. TOSCA, Heat Templates

• Translating those tools into action is the role of the NFVO/VNFM

• Until we achieve a pure model-driven lifecycle, API-driven procedural approaches will have to fill the modeling gaps
Functional Model View

End to End Network Service

Logical Service Abstraction

Software Instances

Virtual Resources

Hardware Resources

Required Metadata (Descriptors)

Network Service
VNFs
Virtual Links
VNF Forwarding Graph

VNF
Connection Points
Virtual Links
Application config params
Scaling policies

VNF
VM
Resource Requirements
Affinity requirements for VDU/VNFC
Config param (infra)

VNF
Physical
resource requirements
Hardware constraints

Execution reference points
Other reference points
Main NFV reference points

Endpoints

5

11 Nov 2015
OPNFV Summit
# Mapping the Lifecycle to Upstream Projects: a starting point

<table>
<thead>
<tr>
<th>Lifecycle Stage</th>
<th>Action</th>
<th>VNFM/NFVO-Related Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Develop</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Deliver</td>
<td>Onboard</td>
<td>Heat-translator</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Tacker</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Murano</td>
</tr>
<tr>
<td>Deploy</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Use</td>
<td>Configure</td>
<td>Heat, Senlin</td>
</tr>
<tr>
<td></td>
<td>Schedule</td>
<td>Nova-scheduler</td>
</tr>
<tr>
<td></td>
<td>Start/Stop</td>
<td>Tacker</td>
</tr>
<tr>
<td></td>
<td>Suspend/Resume</td>
<td>Tacker</td>
</tr>
<tr>
<td>Manage</td>
<td>Monitor</td>
<td>Monasca, Congress</td>
</tr>
<tr>
<td></td>
<td>Scale</td>
<td>Heat, Senlin</td>
</tr>
<tr>
<td></td>
<td>Migrate</td>
<td>Nova</td>
</tr>
<tr>
<td></td>
<td>Upgrade</td>
<td></td>
</tr>
<tr>
<td>Retire</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
An integration view: current and in-development relationships
Challenges in mapping MANO functions to upstream

• OpenStack perceived as VIM
• OpenStack projects and functions are continually shifting
• OpenStack is pushing up into the MANO stack, but not under an overall strategy
• Other projects are competing to provide the same functions within their domain
Conclusions

• MANO architecture does not map well to OpenStack functionality. Not a surprise here.

• Existing OpenStack projects may provide the functionality in order to fulfill most of the MANO requirements for management and orchestration especially for VNF Lifecycle

• Some project functionality overlaps and due diligent selection of existing modules is going to be important

• In OpenStack, the management and orchestration functions are more likely going to be distributed across several components based on several factors:
  – Managed entities (VMs, VNFs, Services)
  – Functional role
  – Available code base

• The plug-in architecture will allow for vendor/service provider specific capabilities while maintaining consistency across platform
Next Steps

- Focused OPNFV discussions on VNF and service lifecycle
- User guides and tests demonstrating current lifecycle management features supported in the OPNFV Brahmaputra release
- Consider OPNFV ‘C’ release inclusion of NFVO/VNFM support goal

- The main project in the OpenStack Orchestration program
- Implements an orchestration engine to launch multiple composite cloud applications based on templates in the form of text files that can be treated like code
- Widely used and with broad industry support
Aim: To develop a fully functional ETSI MANO / IFA based general purpose NFV Orchestrator + VNF Manager for OpenStack

**NFVO**
- Templated end-to-end Network Service deployment using decomposed VNFs
- VNF placement policy – ensure efficient placement of VNFs
- VNFs connected using a SFC - described in a VNF Forwarding Graph Descriptor
- VIM Resource Checks and Resource Allocation
- Ability to orchestrate VNFs across Multiple VIMs

**VNFM**
- VNF Catalog
- Basic life-cycle of VNF (define/start/stop/undefine)
- Performance and Health monitoring of deployed VNFs
- Auto Healing VNFs based on Policy
- Facilitate initial configuration of VNF
Tacker: Status and relationships to other projects

- Integrated with
  - HEAT – VNF Creation
  - Monasca / Ceilometer - VNF Monitoring

- Developing
  - VNF state-machine
  - Basic auto-scaling

- Plans
  - Integrate with ODL SFC: using SDNC plugin to push config for specific VNFs
  - VNF Catalog – to leverage Murano

- Tosca-Heat Translator split in two libraries (parser, generator), parser to become a dependent library
Murano ([https://wiki.openstack.org/wiki/Murano](https://wiki.openstack.org/wiki/Murano))

- App developers can publish applications and services, including deployment rules and requirements, suggested configuration, output parameters and billing rules
- Tracking billing and usage information
- End-users/tenants can find and self-provision third-party applications and services, integrate them into their environment, and track usage information and costs
Murano (https://wiki.openstack.org/wiki/Murano)

- Service Admin UI
  - Create, update, list, delete Services
  - Manage Billing rules
  - Manage ACLs
  - Track usage
  - Track billing information

- User self-service UI
  - List available services
  - Provision a service and integrate it into environment
  - Track usage
  - Track billing information

- Murano Metadata
- Murano API
- RBAC
- Environment Control
- Billing

- Application Catalog

- Heat Orchestration
- Taskflow\Mistral
- OpenStack IaaS

- Orchestration Level

- Ceilometer

VM
Monasca ([https://wiki.openstack.org/wiki/Monasca](https://wiki.openstack.org/wiki/Monasca))

- Multi-tenant, highly scalable, performant, fault-tolerant monitoring-as-a-service solution
- Exposes REST API for high-speed metrics processing and querying
- Streaming alarm and notification engines
- Integrates with:
  - Ceilometer
  - Horizon
“Policy as a Service” enabling VNF/service-specific and independent policies for state/event conditions across OpenStack services, and any other platform for which a data source driver is implemented

Can be used to detect and respond to conditions of an arbitrary nature, given that those conditions can be represented by a table join on data obtained from data sources

- Provides a generic clustering service for an OpenStack cloud, capable of managing objects exposed by Nova, Heat, Cinder etc.
  - A generic clustering/collection service for managing groups of homogeneous cloud objects on OpenStack.
  - A set of APIs for managing cluster membership, e.g. add/remove nodes.
  - A plugin-based object profile management enabling the creation and management of any object pools.
  - A plugin-based policy enforcement framework featuring flexible policy customization for cluster management.
  - An asynchronous execution engine for ensuring the state consistency of clusters and nodes.
  - An open design for action execution that can be extended to accommodate complex application deployment.
Key ETSI NFV IFA focus areas

- Network Service Lifecycle Management
- Virtual Network Function Lifecycle Management
- Resource Management
- Performance Management
- Fault Management

Drafts available in the public area: https://docbox.etsi.org/ISG/NFV/open/Drafts/
NFV IFA Work (phase 2)

OSS/BSS

EMS 1  EMS 1  EMS 1
VNF 1  VNF 1  VNF 1

NFV Orchestrator (NFVO)

Os-Ma-nfvo
IFA013
IFA012

VE-Vnfm
IFA008

VNF Manager(s)

Or-Vnfm
IFA007

Vi-Vnfm
IFA006

Virtualized Infrastructure Manager(s)

Or-Vi
IFA005

Virtualization Layer

Virtual Compute  Virtual Storage  Virtual Network

Hardware Resources

Compute Hardware  Storage Hardware  Network Hardware

NFVI
Network Service onboarding: Os-Ma-nfvo (IFA013, IFA012)

**NSD management**
Operations:
- Onboard NSD
- Disable NSD
- Enable NSD
- Query NSD
- Delete NSD

**NS Lifecycle Change Notification**
Operations:
- Notify

**NS Lifecycle Management**
Operations:
- Instantiate NS
- Terminate NS
- Scale NS
- Update NS
- Create VNFFG
- Delete VNFFG
- Query VNFFG
- Update VNFFG
- Create VL
- Delete VL
- Update VL
- Query VL
VNF Lifecycle Management

- OSS/BSS
  - Os-Ma-nfvo

NFV Orchestrator (NFVO)
  - VNF Package Management
  - VNF Lifecycle granting Management
  - VNF Lifecycle management
  - VNF change management

- EM
  - Ve-Vnfm
  - IFA008

VNF Manager(s)
  - VNF Package Management
  - VNF Lifecycle granting Management
  - VNF Lifecycle management
  - VNF change management
VNF Lifecycle Management (LcM)

- VNF LcM interface is exposed by VNFM
- VNF LcM interface can be re-exposed by NFVO
- VNF instantiation:
  - Resource Allocation
  - VNF Configuration
- MANO offers two options for resource management
  - Performed by NFVO (effectively acting as a Resource Orchestrator). VNFM requests resources to the VNFO and VNFO forwards the request to the VIM
  - Performed by the VNFM. VNFM ask VNFO for granting request and then directly requests resources to the VIM
VNF Instantiation Flow (NFVO initiated)

1. Instantiate VNF

2. Forward instantiation Request

3. Request Resource Allocation

4. Request Resource Allocation

5. Configure VNF Deployment Parameters

6. Notification of VNF Instantiation

7. Configure VNF app parameters

OSS/BSS

EMS

VNF

NFV Orchestrator (NFVO)

VNF Manager(s)

Virtualized Infrastructure Manager(s)
Resource Management

- Resource management models for VIM:
  - Reservation
  - On-demand
  - Quota based

<table>
<thead>
<tr>
<th>Lifecycle Stage</th>
<th>Operations related to</th>
<th>Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>VNF</td>
<td>NFVO</td>
<td>Or-Vi</td>
</tr>
<tr>
<td></td>
<td>Initial allocation, update and release of resources for VNF instantiation, scaling and termination</td>
<td></td>
</tr>
<tr>
<td>VNF</td>
<td>VNFM</td>
<td>Vi-Vnfm</td>
</tr>
<tr>
<td></td>
<td>Initial allocation, update and release of resources for VNF instantiation, scaling and termination</td>
<td></td>
</tr>
<tr>
<td>NS</td>
<td>NFVO</td>
<td>Or-Vi</td>
</tr>
<tr>
<td></td>
<td>Initial Allocation, updates and release for NS instantiation, scaling and termination</td>
<td></td>
</tr>
</tbody>
</table>