\Orchestrating a brighter world NEC

2nd Tokyo OpenStack Meetup at VMware

NFV related features in OpenStack

Ryota Mibu

July 9th, 2015

Intro

Ryota Mibu

NEC Engineer

OpenStack Developer (ATC)

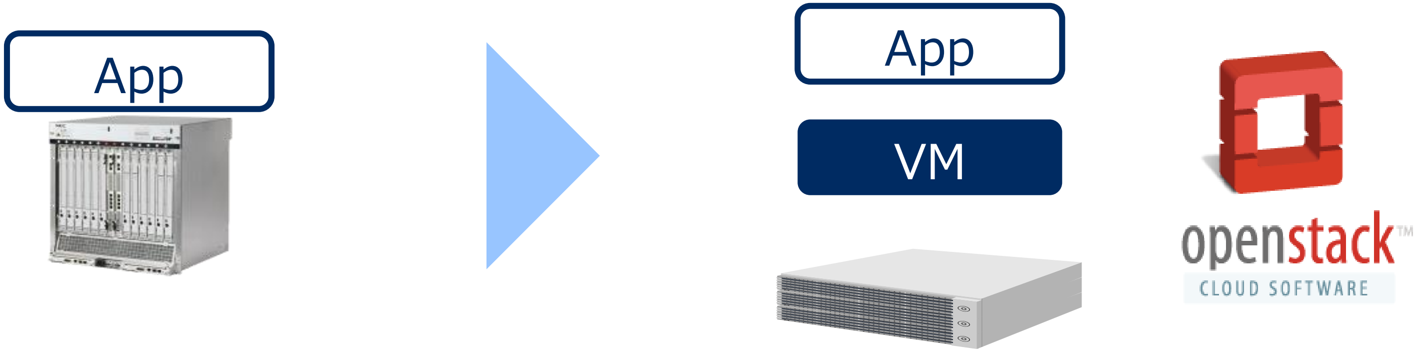
Project Lead of Doctor
(OPNFV Fault Management project)

What is NFV?

Network Functions Virtualization

Virtualizing network equipment to software application

Adopting cloud technologies into platform which hosts these applications



App

App

VM

Why Care About NFV?

NFV is a new trend in Telco industry as well as OpenStack ecosystem

NFV has various use cases and requirements that lead enhancement of OpenStack as shared platform

"Carrier Grade"

NFV related key words in OpenStack

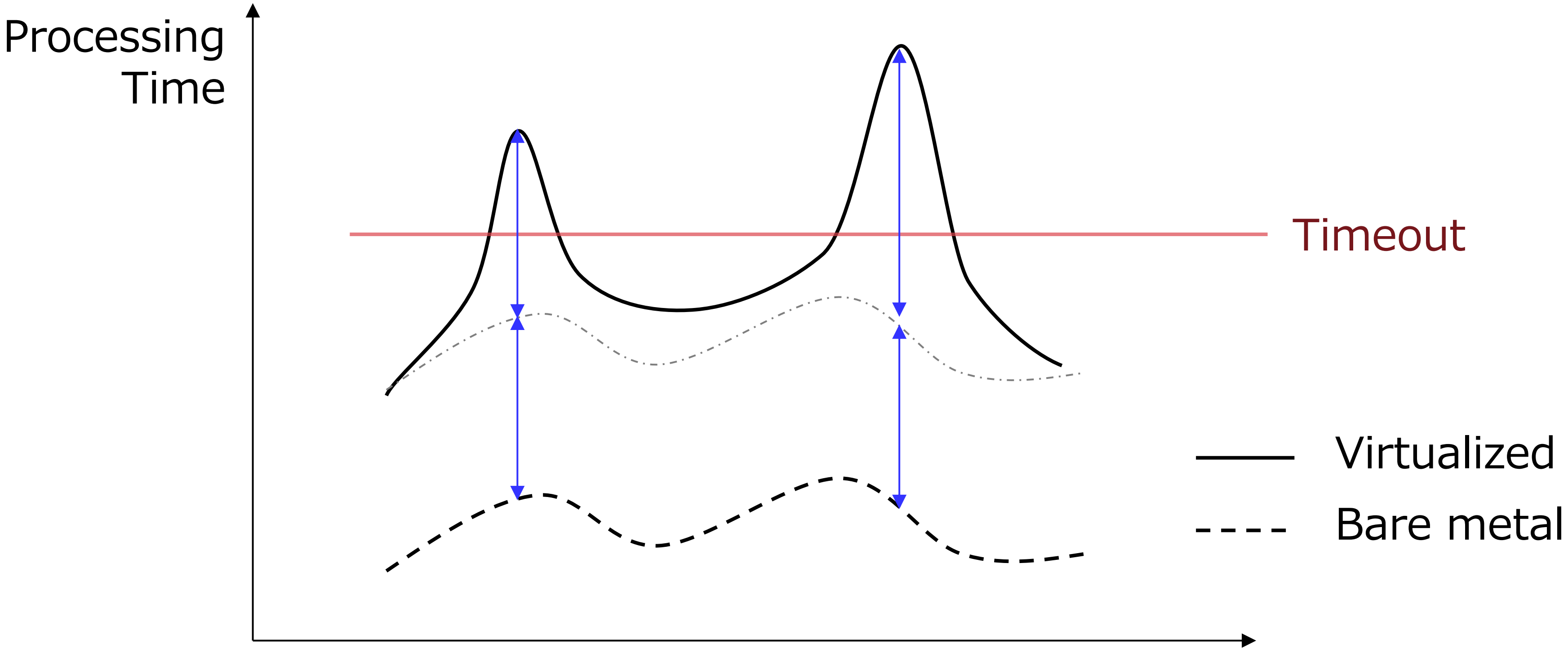
High Performance
Predictable Performance
High Availability
External Monitoring Tools
Multi-site / Cascading
Policy Management
CPU Pinning
NUMA Aware Scheduling ←
I/O Based NUMA Scheduling
Large/Huge Page
Schedule VM in evacuation
Solver Scheduler
Diskless VM

OVF Support
Service Function Chaining
Traffic Steering
Services Insertion
Service VM
SR-IOV Networking Support
Two vNICs One Network
Unaddressed Interfaces
VLAN Trunking Network ←
Flow-based Security Groups
VHOSTUSER (Snabb, DPDK)
Event Alarm ←
Mark Host Down

https://wiki.openstack.org/wiki/TelcoWorkingGroup https://wiki.opnfv.org/community/openstack

Nova: NUMA aware CPU Scheduling

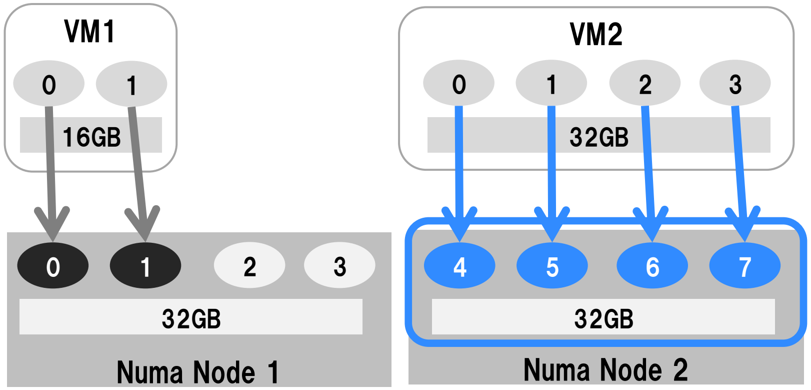
Virtualization may cause delay and spike in processing time by interruption, cache pollution and etc.



Nova: NUMA aware CPU Scheduling

End User can specify how vCPUs map to CPUs and whether he needs dedicated CPU

Note: Available from Juno release



Nova: NUMA aware CPU Scheduling

[User Step 1] Specify vCPU topologies in Flavor or Image Property

hw:cpu_sockets=NN - preferred number of sockets to expose to the guest

hw:cpu_cores=NN - preferred number of cores to expose to the guest

hw:cpu_threads=NN - preferred number of threads to expose to the guest

hw:cpu_max_sockets=NN - maximum number of sockets to expose to the guest

hw:cpu_max_cores=NN - maximum number of cores to expose to the guest

hw:cpu_max_threads=NN - maximum number of threads to expose to the guest

https://github.com/openstack/nova-specs/blob/master/specs/juno/implemented/virt-driver-numa-placement.rst

Nova: NUMA aware CPU Scheduling

[User Step 2] Set mapping policies in Flavor or Image Property

hw:numa_nodes=NN - numa of NUMA nodes to expose to the guest.

hw:numa_mempolicy=preferred|strict - memory allocation policy

hw:numa_cpus.0=<cpu-list> - mapping of vCPUS N-M to NUMA node 0

hw:numa_cpus.1=<cpu-list> - mapping of vCPUS N-M to NUMA node 1

hw:numa_mem.0=<ram-size> - mapping N GB of RAM to NUMA node 0

hw:numa_mem.1=<ram-size> - mapping N GB of RAM to NUMA node 1

hw:cpu_policy=shared|dedicated

hw:cpu_threads_policy=avoid|separate|isolate|prefer

avoid: the scheduler will not place the guest on a host which has hyperthreads.

separate: if the host has threads, each vCPU will be placed on a different core. ie no two vCPUs will be placed on thread siblings

isolate: if the host has threads, each vCPU will be placed on a different core and no vCPUs from other guests will be able to be placed on the same core. ie one thread sibling is guaranteed to always be unused

prefer: if the host has threads, vCPU will be placed on the same core, so they are thread siblings.

https://github.com/openstack/nova-specs/blob/master/specs/kilo/implemented/virt-driver-cpu-pinning.rst

Nova: NUMA aware CPU Scheduling

[Nova 1] Nova get CPU info of each host

nova/db/sqlalchemy/models.py

```
class ComputeNode(BASE, NovaBase):
    """Represents a running compute service on a host."""
…
    # Note(masumotok): Expected Strings example:
    #
    # '{"arch":"x86_64",
    #   "model":"Nehalem",
    #   "topology":{"sockets":1, "threads":2, "cores":3},
    #   "features":["tdtscp", "xtpr"]}‘
    # Points are "json translatable" and it must have all dictionary keys
    # above, since it is copied from <cpu> tag of getCapabilities()
    # (See libvirt.virtConnection).
    cpu_info = Column(MediumText(), nullable=False)
```

Nova: NUMA aware CPU Scheduling

[Nova 2] Nova find host available for the request

nova/scheduler/filters/numa_topology_filter.py

nova/virt/hardware.py

```
def numa_fit_instance_to_host(
        host_topology, instance_topology, limits=None,
        pci_requests=None, pci_stats=None):
    """Fit the instance topology onto the host topology given the limits
…
    Given a host and instance topology and optionally limits - this method
    will attempt to fit instance cells onto all permutations of host cells
    by calling the _numa_fit_instance_cell method, and return a new
    InstanceNUMATopology with it's cell ids set to host cell id's of
    the first successful permutation, or None.
    """
```

Nova: NUMA aware CPU Scheduling

[Nova 3] Nova find the best assignment in the scheduled host

nova/virt/hardware.py

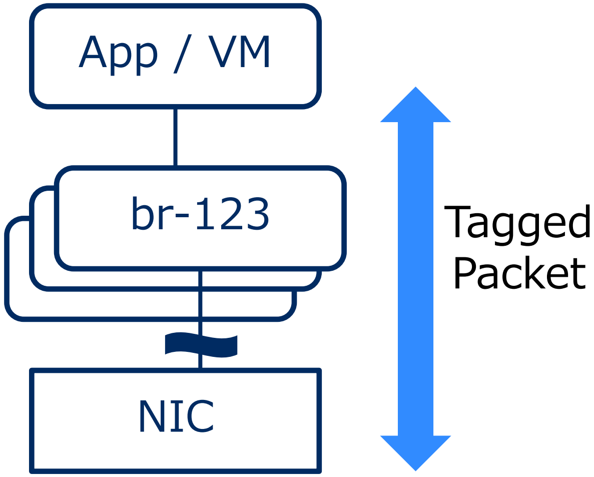
```
def get_best_cpu_topology(flavor, image_meta, allow_threads=True,
                          numa_topology=None):
    """Get best CPU topology according to settings
…
    Look at the properties set in the flavor extra specs and
    the image metadata and build up a list of all possible
    valid CPU topologies that can be used in the guest. Then
    return the best topology to use

    :returns: a nova.objects.VirtCPUTopology instance for best topology
    """
```

Neutron: VLAN Trunking Network

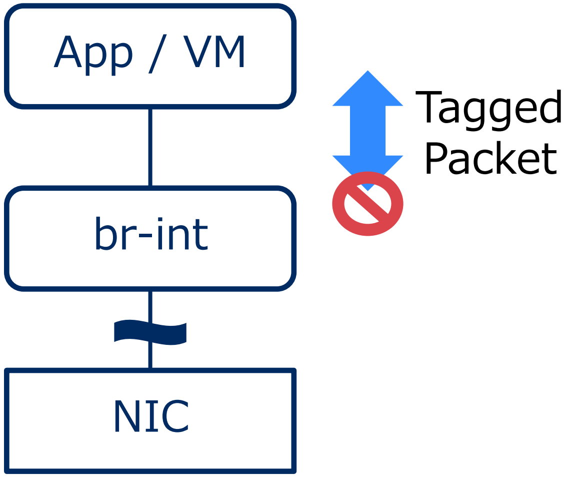
Telco Application uses tag VLAN and requires VLAN transparent Network.

This won’t change shortly due to deployment with physical servers or router.

ML2 + linuxbridge agent



ML2 + OVS agent



Neutron: VLAN Trunking Network

[User] User can request VLAN transparent network by passing a ‘vlan-transparent’ boolean property on the net-create request.

[Neutron]

If vlan-transparent==true, then the plugin is a VLAN aware plugin and (regardless of the request) has created a network capable of passing VLAN tagged packets.

http://specs.openstack.org/openstack/neutron-specs/specs/kilo/nfv-vlan-trunks.html

Ceilometer: Event Alarm

How can you find VM faults as a tenant user?

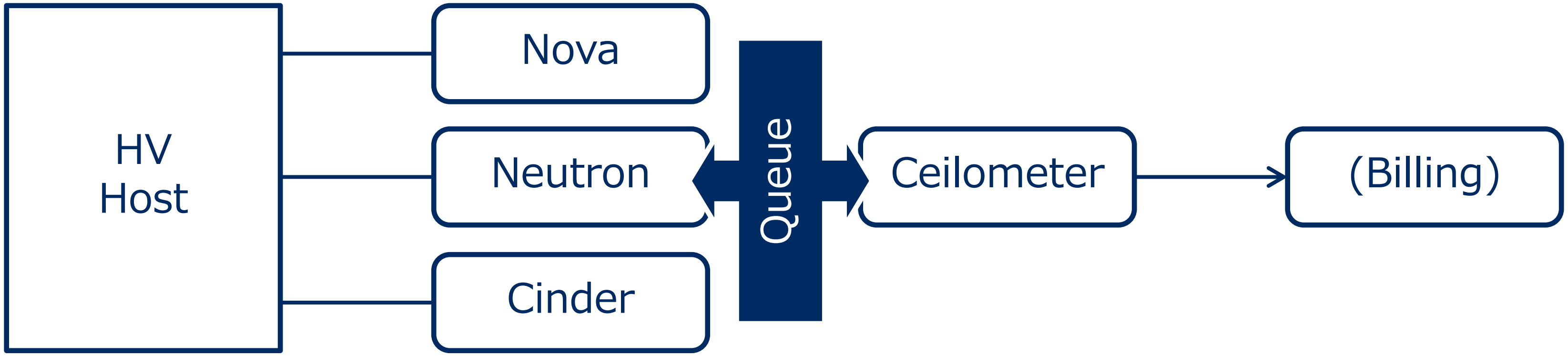
Keep-a-live check to each VM

Polling VM state to Nova API

Set alarm on metering service (e.g. CPU runtime)

Ceilometer: Event Alarm

Nova knows your VM has error...



Ceilometer: Event Alarm

[User] User can set alarm type=‘event’ via alarm API

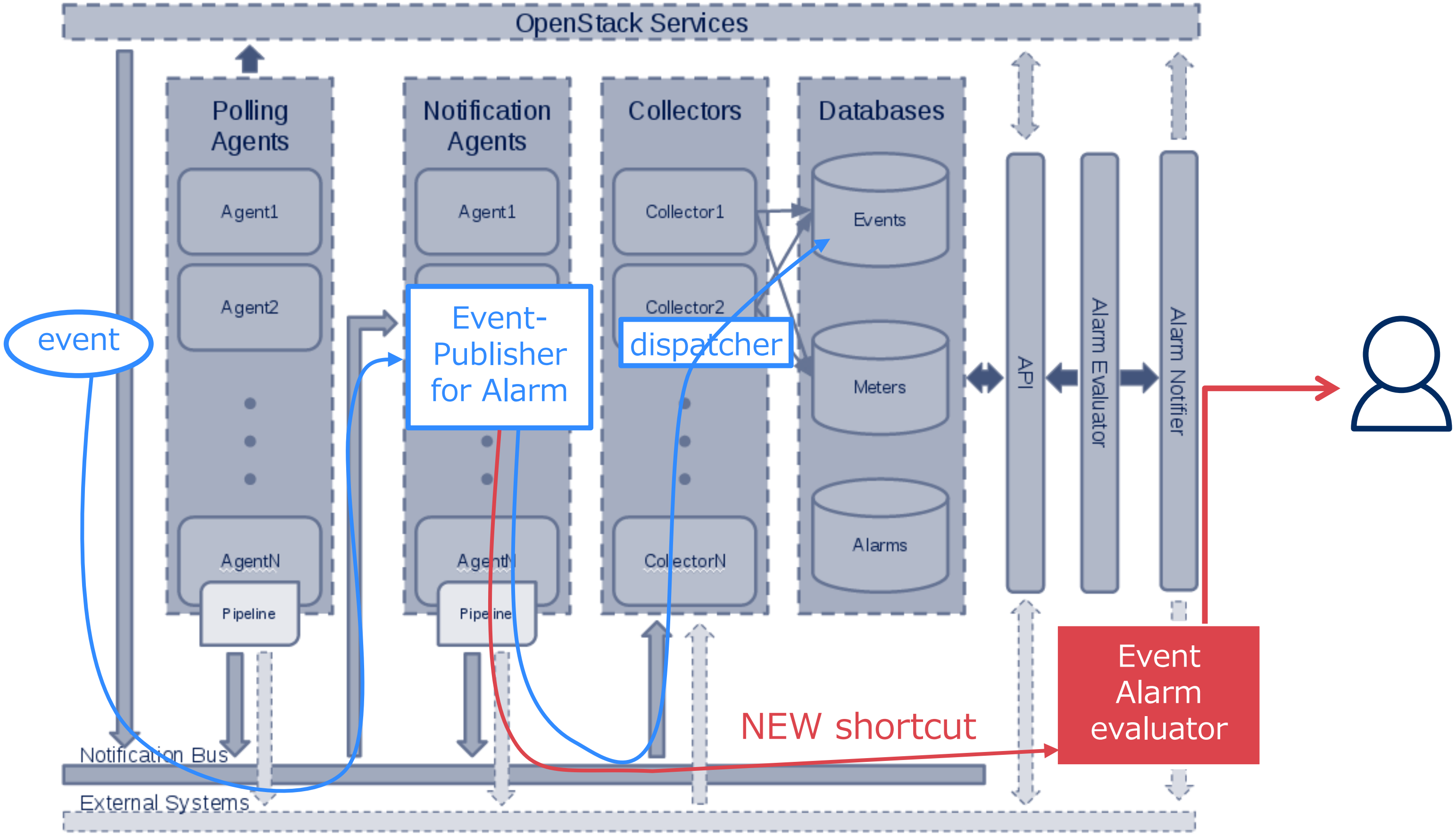
ceilometer-specs/specs/liberty/event-alarm-evaluator.rst

```
Sample data of Notification-type alarm::
…
    "name": "InstanceStatusAlarm",
    "event_rule": {
        "event_type": "compute.instance.update",
        "query" : [
…
            {
                "field" : "traits.state",
                "type" : "string",
                "value" : "error",
                "op" : "eq",
            },
```

Ceilometer: Event Alarm

# Under Development

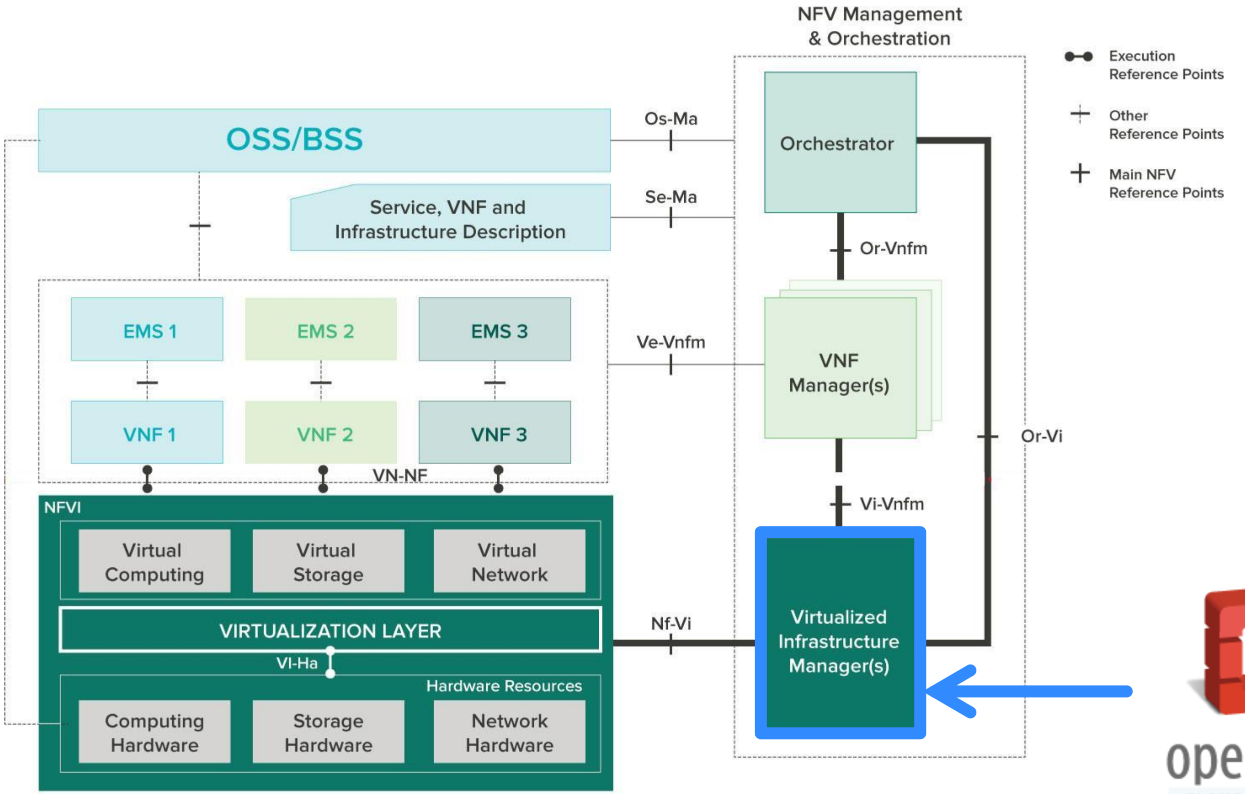
Ceilometer captures event and evaluate it on the fly



http://docs.openstack.org/developer/ceilometer/architecture.html

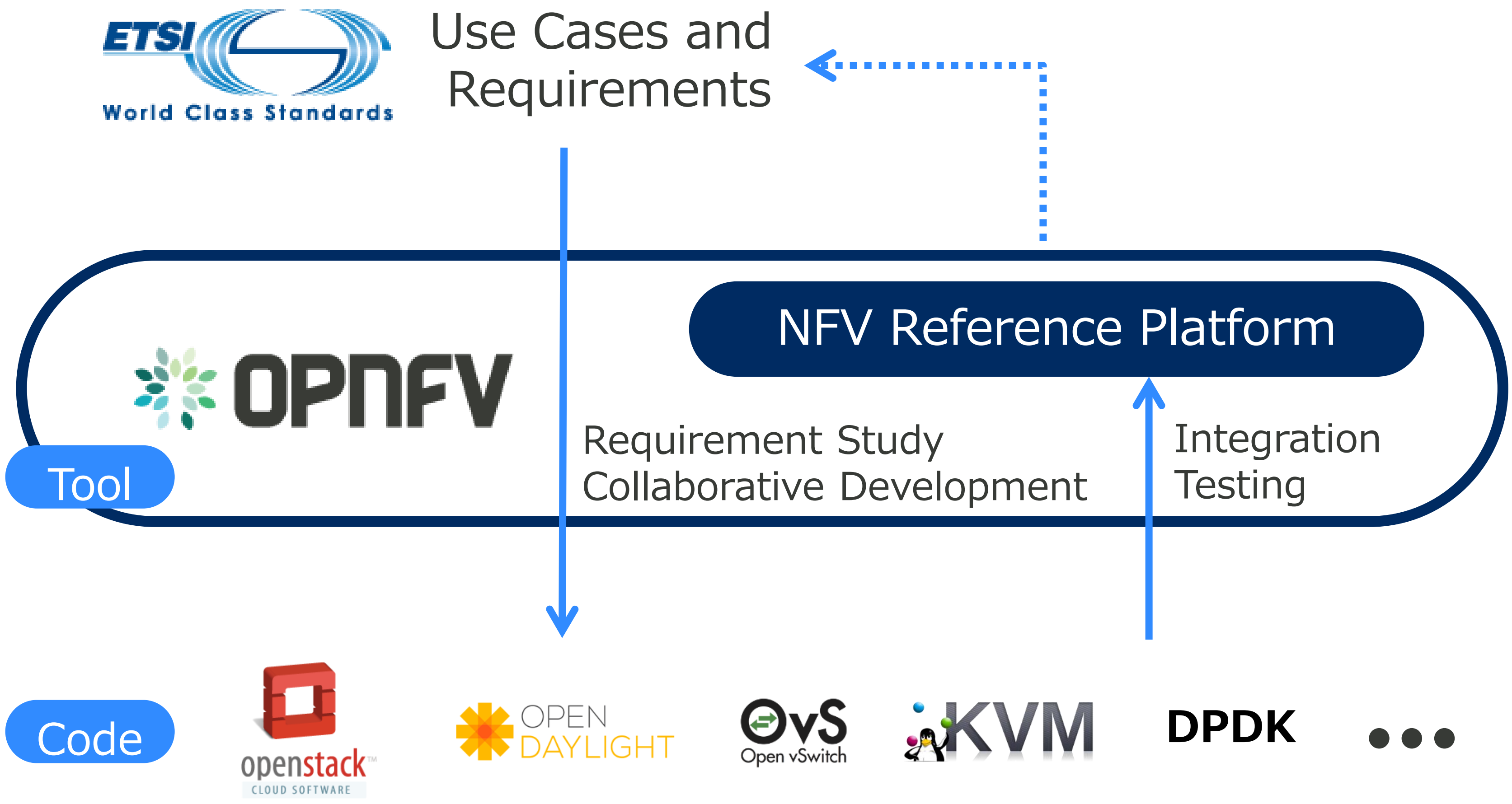
OpenStack Role in NFV System

OpenStack only covers part of NFV system that means we need discussion where each feature should be implemented in.



https://www.opnfv.org/software/technical-overview

OPNFV



Use Cases and Requirements

Tool

NFV Reference Platform

Requirement Study
Collaborative Development

Integration
Testing

Code

DPDK

•••

Note: Various enhancement along with the direction of each OSS,
NOT creating NFV exclusive features