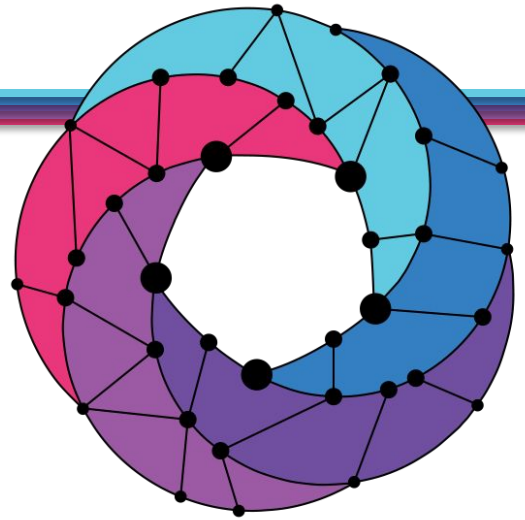


Open Source
MANO

OSM#9 Hackfest

# Placement optimization for our Network Services

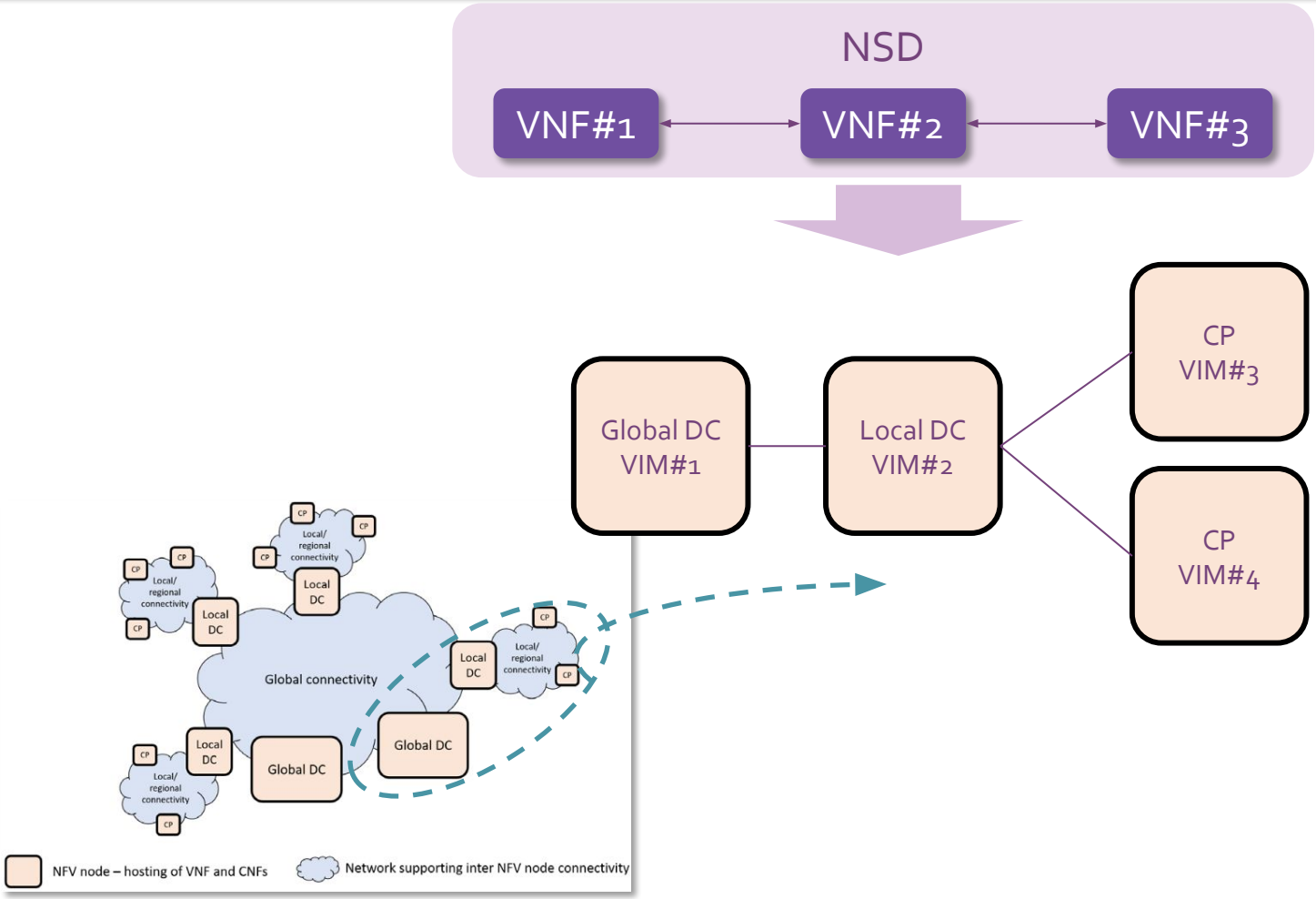
Lars-Göran Magnusson (Arctos Labs)

Open Source MANO

# Introduction to Placement Optimization

Note: Placement is targeted for next OSM Release

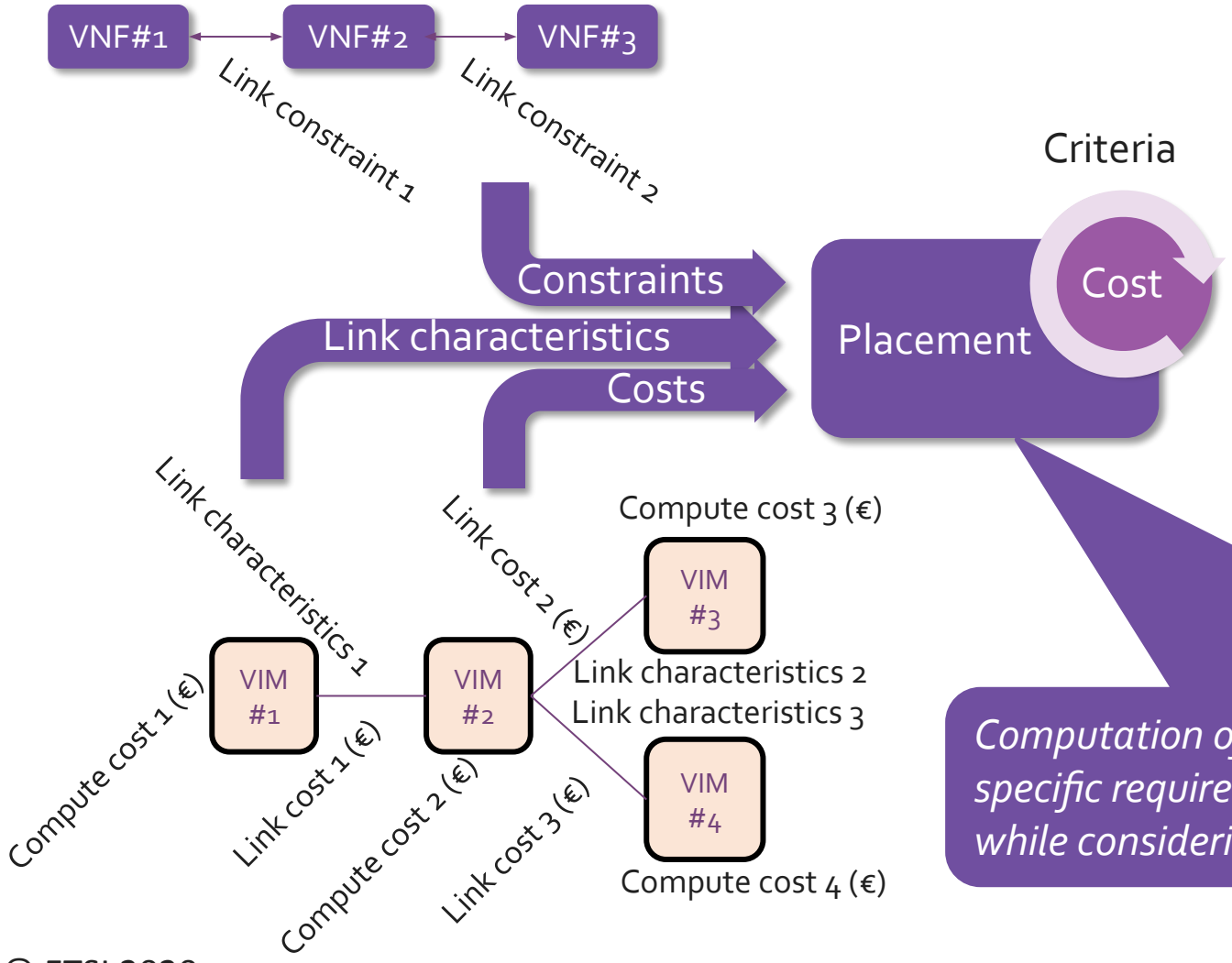

# What do we mean by Placement Optimization?




**Business Service Basic Architecture,**
**from *OSM Deployment and Integration WP, Feb 2020***

- Placement in context of OSM is the process of deciding **which VNF goes into which VIM**
- Optimal is subject to:
  - Cost of compute in VIMs
  - Cost of links for NS interworking
  - Constraints in NS interworking (Latency, Jitter) – if there are any
- Placement feature makes this process **Automatic & Optimal**

# The Optimization Process




- Placement function
  - Will consider all VIM's available to the user
  - Will make sure constraints are met – if there are any
  - Will optimize Cost (the Criteria)
- I.e. select the option that fulfills constraints at the lowest possible cost
  - Modeled as a constraints optimization problem

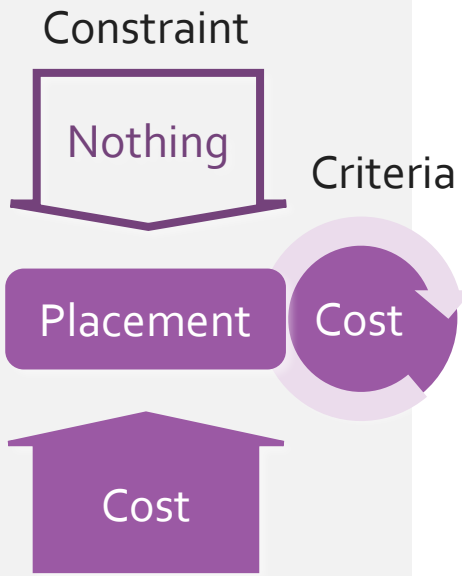
*Computation of optimal placement of VNFs over VIMs by matching NS specific requirements to infrastructure availability and run-time metrics, while considering cost of compute/network.*

# Placement optimization examples

Open Source MANO

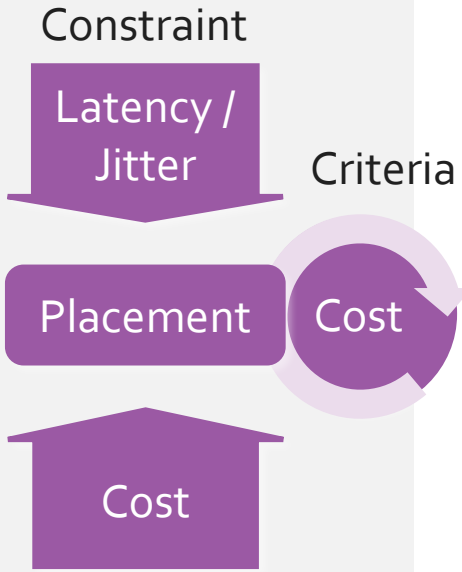
## 1

**Cost** optimization **only**



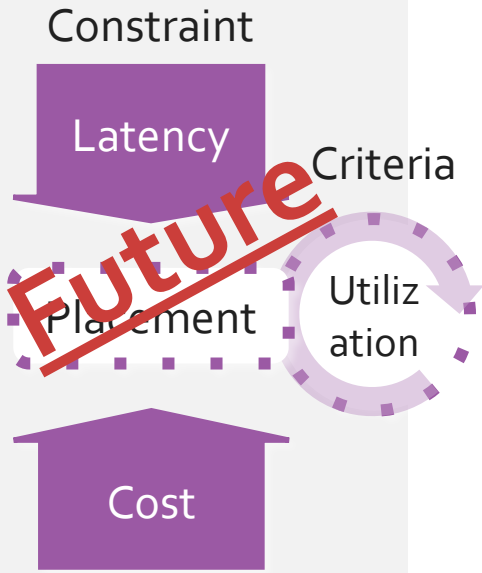

## 2

**Cost** optimization **with Latency** constraint



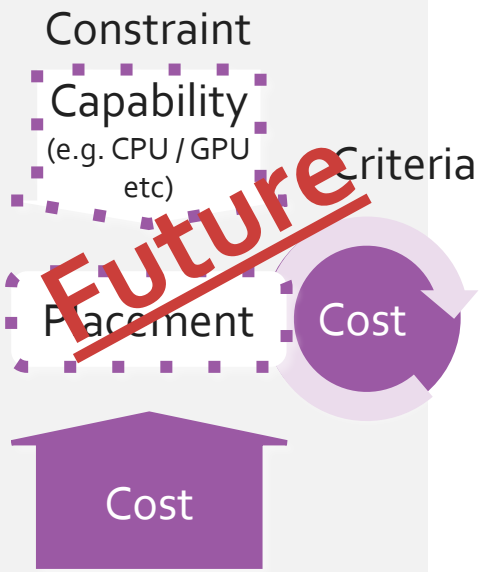

## 3

**Utilization** optimization **with Latency** constraint




## 4

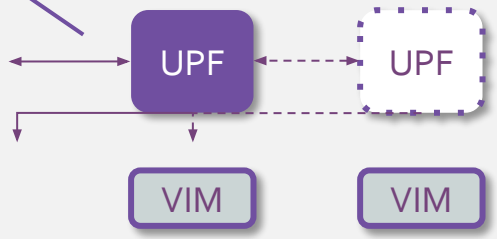
**Cost** optimization **with Capability** constraint



# Examples of use cases

## UPF supporting Low-latency

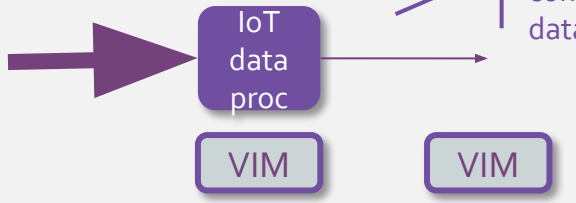
Placement of UPF close to customer to achieve latency constraint




Deploy as close as it has to be

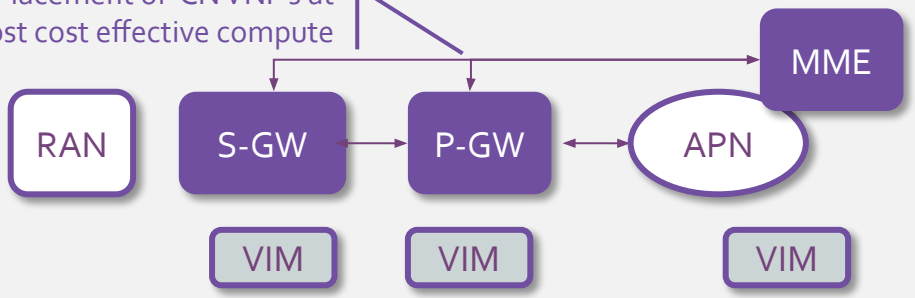
## Transport optimization (cost) for Application components

Placement of Application components close to the source of data to reduce transport cost / load




## Compute cost optimization for slicing

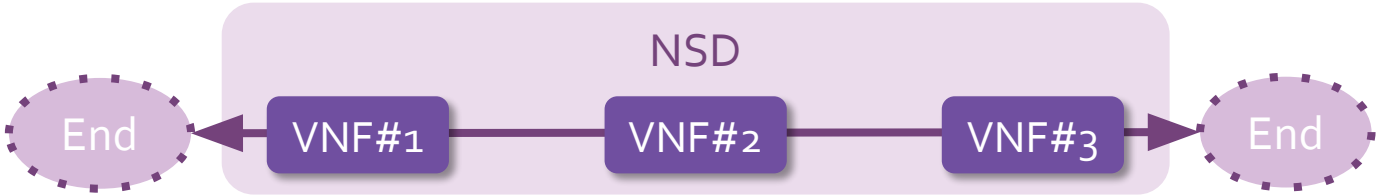
Placement of CN VNF's at most cost effective compute




Deploy as far away as it can be

# VNF Pinning

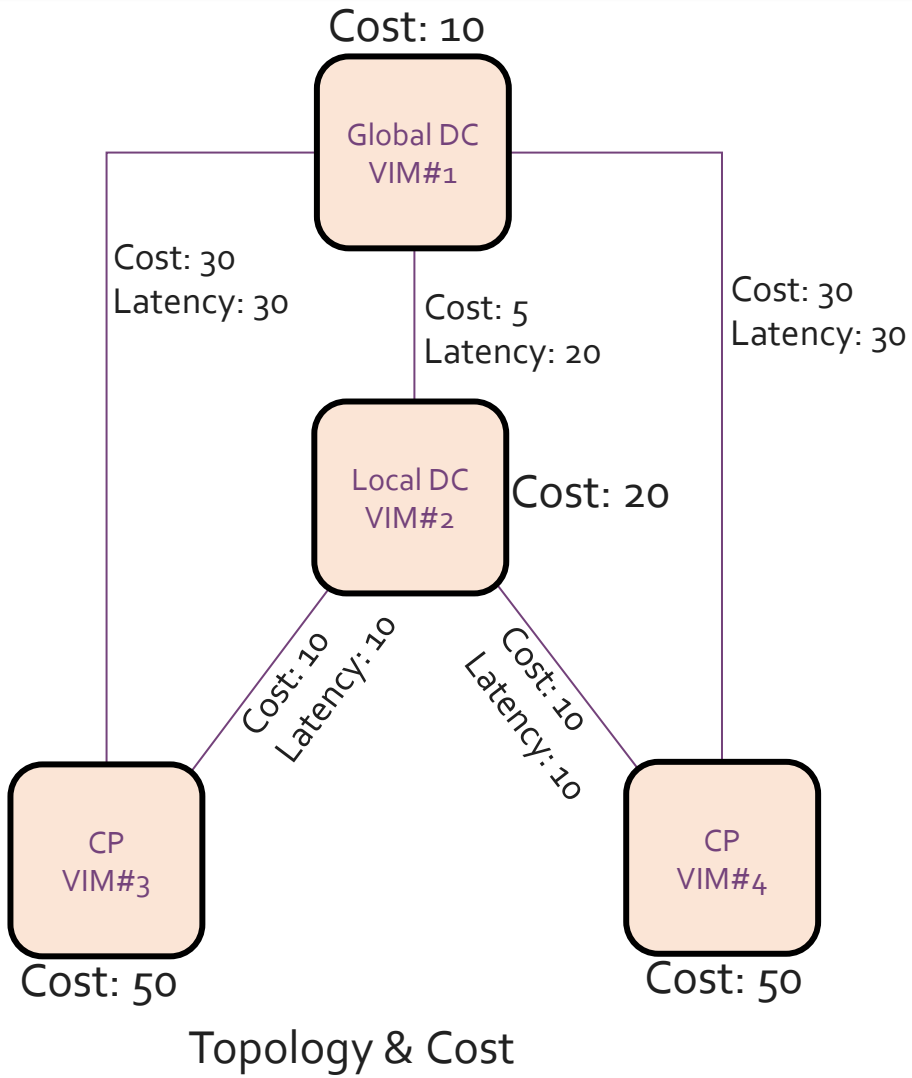
| | | | |
|---|---|---|---|
| Example 1: | Auto | Auto | VIM#3 |
| Example 2: | VIM#2 | Auto | Auto |
| Example 3: | Auto | Auto | Auto |




- Ability to "pin" a VNF to e.g.
  - the VIM with a specific VNF (e.g. P-GW)
  - the VIM with connectivity to a PNF
  - a CPE (customer location)

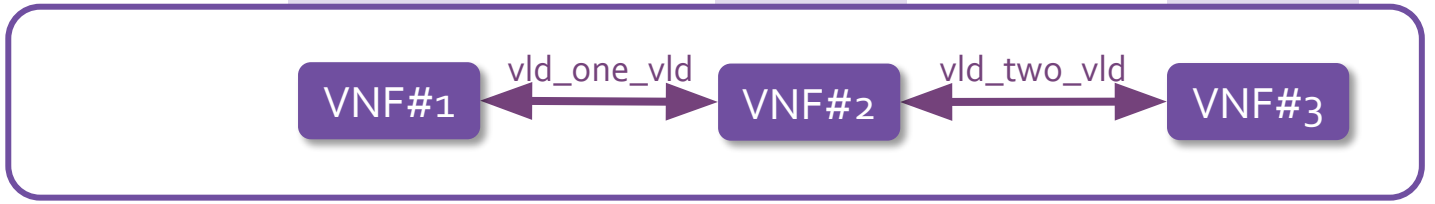
*Auto implies there is no VIM specified, this placement is therefore subject to placement optimization*
*=> this is what Placement is all about – finding out where VNFs should (or must) be deployed in a multi-VIM NFVI*

# Some different scenarios




Topology & Cost

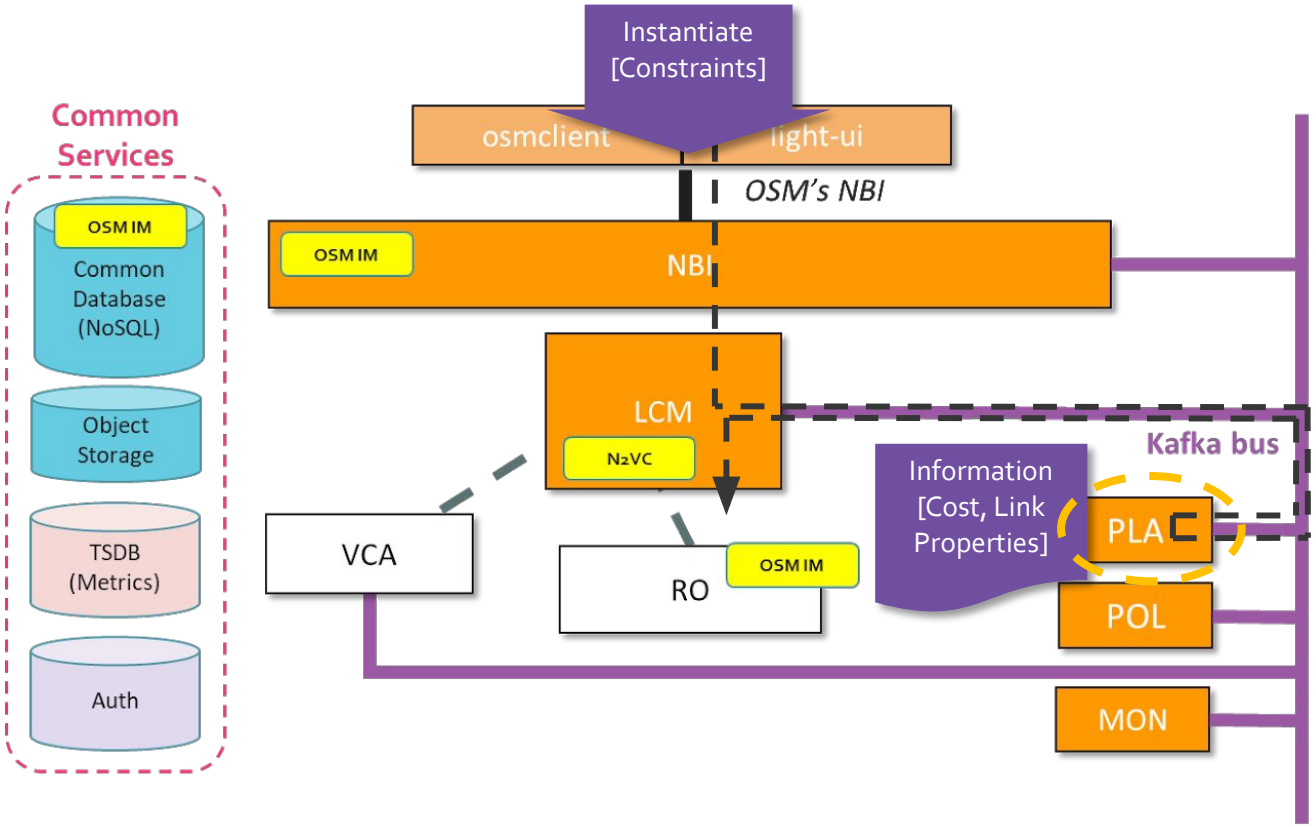
| | Pinning | Constraints | Pinning | Constraints | Pinning |
|---|---|---|---|---|---|
| Scenario a: | Auto | | Auto | | Auto |
| Scenario b: | VIM#3 | | Auto | | Auto |
| Scenario c: | VIM#4 | | Auto | latency=15 | Auto |



Open Source MANO

# Install and configure PLA in OSM


# The PLA component in OSM




- New component
  - Optional, install with --pla

- Basic functionality initially
- Automatic placement is optional, invoked by the user at instantiate of Network Service
  - --config '{placement-engine: PLA, placement-constraints: {}, ...}'
  - Constraints given in the instantiation request
  - Will consider placement over the VIMs available to the user
- Interacts with LCM, Common Services

# Configure PLA

- You need two configuration files
  - vnf_price_list.yaml
  - pil_price_list.yaml
- The configuration files are copied to the PLA container using the following commands:

```
$ docker cp vnf_price_list.yaml $(docker ps
-qf name=osm_pla):/placement/.
```

```
$ docker cp pil_price_list.yaml $(docker ps
-qf name=osm_pla):/placement/.
```

The price list for compute determines the price for each VNF at each VIM . The file (vnf_price_list.yaml) is written in Yaml

```
- vnfd: hackfest_magma-agw-enb_vnfd
  hackfest:
      prices:
      - vim_url: http://172.21.247.1:5000/v3
        vim_name: etsi-openstack
        price: 5
      - vim_url: http://172.21.7.5:5000/v3
        vim_name: etsi-openstack-lowcost
        price: 1
  admin:
      prices:
      - vim_url: http://172.21.247.1:5000/v3
        vim_name: etsi-openstack
        price: 5
      - vim_url: http://172.21.7.5:5000/v3
        vim_name: etsi-openstack-lowcost
        price: 1
```

The price list and characteristics for transport links between VIMs (PoP Interconnecting Link – PiL). In current release the price is given per link without any consideration to BW or other QoS parameter. The file (pil_price_list.yaml) is written in Yaml.

```
pil:
  - pil_description: Link between vim1 and vim2
      pil_price: 5
      pil_latency: 10
      pil_jitter: 2
      pil_endpoints:
      - etsi-openstack
      - etsi-openstack-lowcost
```

*Note: In current OSM release the link characteristics are hard coded into this file, in future releases this data should be retrieved from the infrastructure by monitoring mechanisms.*

Note: Don't copy the files as hackfest participant, it's already done!

# Invoke PLA

| | | |
|---|---|---|
| 1 | Request Placement Cost Optimization | --config '{ **placement-engine: PLA** }' |
| 2 | Request Placement Cost Optimization with pinning of specified VNF | --config '{placement-engine: PLA, **vnf: [{member-vnf-index: "1", vim_account: OpenStack3}]**}' |
| 3 | Request Placement Cost Optimization with VLD Constraints | --config '{placement-engine: PLA, **placement-constraints: {vld-constraints: [{id: vld_1, link-constraints: {latency: 120, jitter: 20}}, {id: vld_2, link-constraints: {jitter: 20 }}]}}**' |
| 4 | Combo of 2 and 3 | --config '{placement-engine: PLA, **vnf: [{member-vnf-index: "1", vim_account: OpenStack4}], placement-constraints: {vld-constraints: [{id: vld_1, link-constraints: {latency: 15}}]}}**' |



*Note: GUI is also supported, with or without YAML file*

Open Source MANO

# Hands-on: Placement of the Magma AGW + emulator VNF

# Launch a 2nd slice

- Create another VIM

The vim name is important, it must match content of the vnf_price_list.yaml file

--user, --password and –tenant follows your personal settings for the hackfest

```
osm vim-create --name etsi-openstack-${HACKFEST-TENANT}-lowcost --user osm_hackfest_${HACKFEST-TENANT} --password
osm_hackfest_${HACKFEST-TENANT} --auth_url http://172.21.7.5:5000/v3 --tenant osm_hackfest_${HACKFEST-TENANT} --account_type
openstack --config '{management_network_name: management, dataplane_physical_net: physnet2, microversion: 2.32}'
```

Don’t forget the additional configuration

# Launch a 2nd slice

- Run hfscripts/lunch_nsi_placement.sh

```
cd hfscripts/
./launch_nsi_placement.sh
```

- create PDU

- params_slices2.yaml

placement-engine: PLA
wimAccountId: False

Need another agw_id, agw_name e.g. 101

```
netslice-subnet:
- id: slice_hackfest_nsd_epc
  placement-engine: PLA
  wimAccountId: False
  additionalParamsForVnf:
  - member-vnf-index: '1'
    additionalParams:
      agw_id: 'agw_101'
      agw_name: 'AGW101'
      orch_ip: '172.21.251.XXX'  ## change this to the MetalLB IP address of your orc8r_proxy service.
      orch_net: 'osmnet'

- id: slice_hackfest_nsd_epcmgmt
  additionalParamsForVnf:
  - member-vnf-index: 'orc8r'
    additionalParamsForKdu:
    - kdu_name: orc8r
      additionalParams:
        proxyserviceloadBalancerIP: '172.21.251.XXX' # MetalLB IP Address
```

# Launch a 2nd slice

- Check where the vnf was deployed

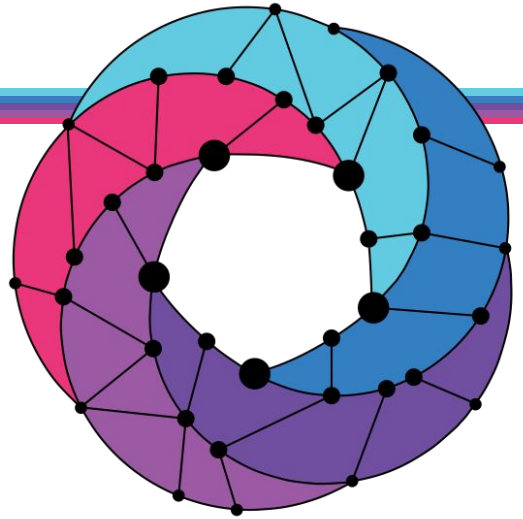
```
osm vnf-list
```

- vim_account_id should correspond to etsi-openstack-x-lowcost for the new slice
- same Magma orc8r as before
- You may configure and send traffic over the new slice

- Clean up: delete the slice

```
$ osm nsi-delete <nsi_name> or <nsi_id>
```

- Clean up: remove parameter file



# Open Source MANO

Find us at:

osm.etsi.org
osm.etsi.org/wikipub