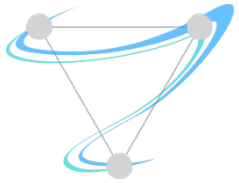


High-level VNF Descriptors using NEMO

draft-aranda-nfvrg-recursive-vnf-03

Pedro A. Aranda paranda@it.uc3m.es
Diego López diego.r.lopez@telefonica.com
Stefano Salsano stefano.salsano@uniroma2.it
Elena Batanero elena.batanerogarcia@telefonica.com

Rationale

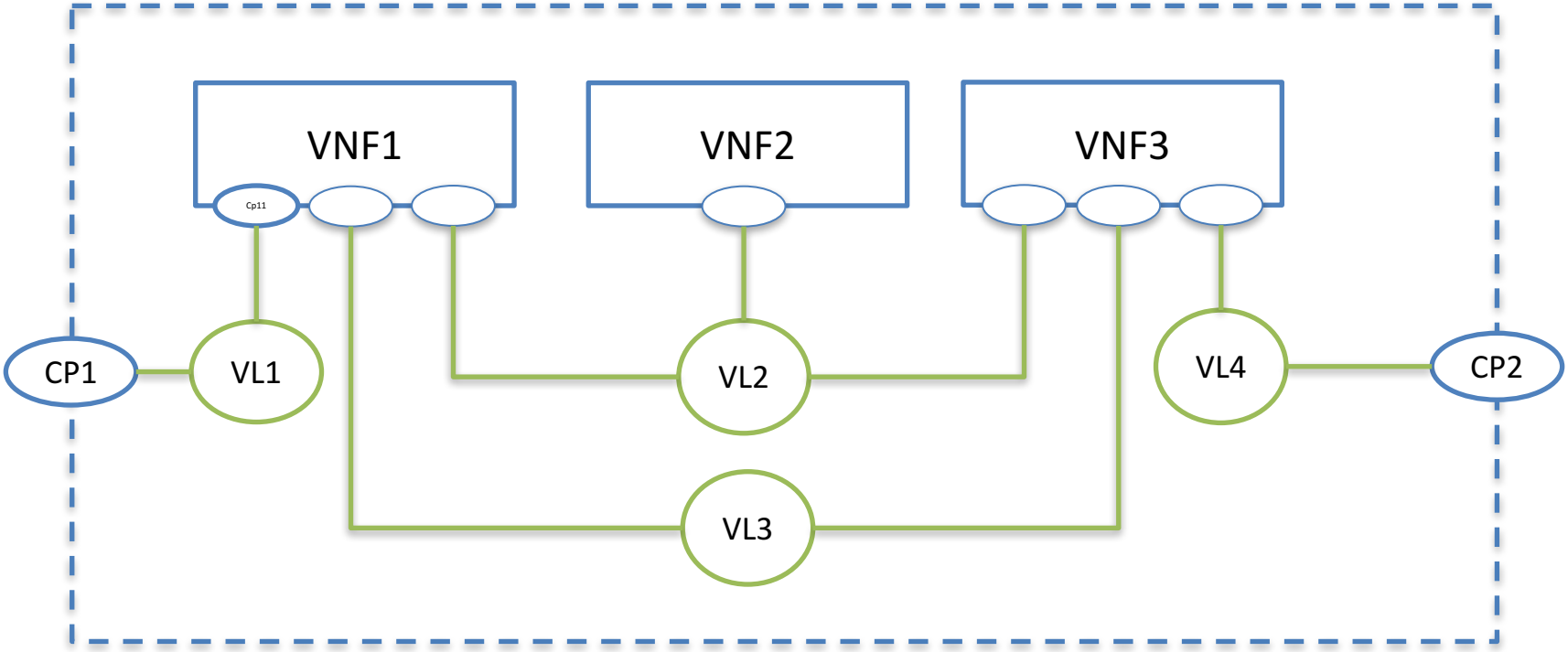
- No one in a clean state of mind can read VNFDs or NSDs easily
  - And less relate them to general network definitions (or policies, or…)
- There is no simple way of reusing tested VNFs to build more elaborate VNFs
  - Current descriptors are most focused on deployment and resource management aspects
- What is right in terms of orchestration goals
  - But goes against one of the goodies of software design/production: RE-USABILITY
- Bridge the gap between network definition and NFV (and SDN) orchestration
  - And this is when intent starts to play

How Would this Work?

- Use NEMO as declarative network definition language
  - Intent declarations
  - https://wiki.opendaylight.org/view/NEMO:Main
- VNFDs like those in OSM are used as low level blocks
- NEMO allows us to describe how VNFs are used to build a network service
  - Caveat emptor: Service has many meanings here!
- Forwarding graphs become more clear using the Connection concept
  - And suitable for parameterization by matching them to VNF interfaces
- Reuse models
  - Opening the door to recursiveness

What We Want

- Find a way to describe VNFs as close as possible to this graph



- And be able to translate this definition into an appropriate orchestration script

Bringing a VNFD into NEMO

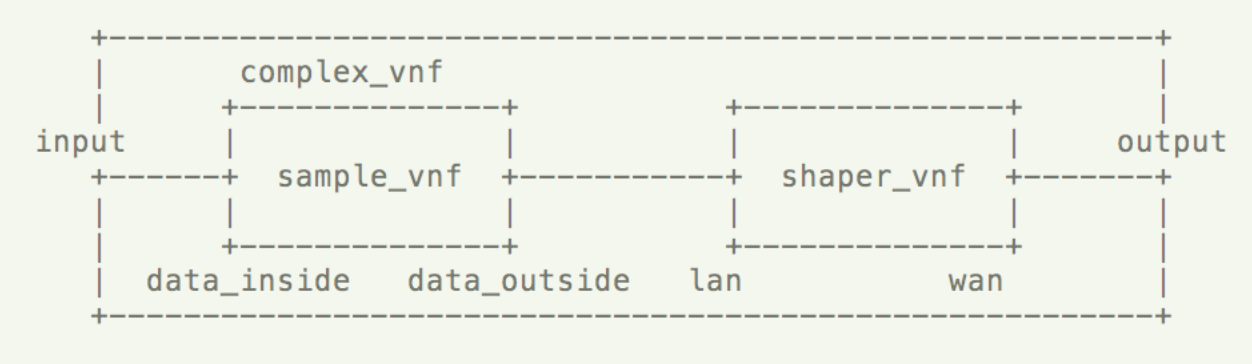
- Reference the descriptor URI
  - VNF producers are required to provide them to orchestrators
- Reference the VNF interfaces relevant for the NodeModel
  - Identifying them by the ConnectionPoint construct

```
CREATE NodeModel sample_vnf VNFD
https://github.com/nfvlabs
/openmano.git/openmano/vnfs/examples/dataplaneVNF1.yaml;
       ConnectionPoint data_inside at VNFD:ge0;
       ConnectionPoint data_outside at VNFD:ge1;
```

Making It Recursive

- Use the imported NodeModel to build more complex functionality
- The Connection construct is used to define the service graph
  - Referencing connection points in the composed VNFs
- And this becomes an NS (in ETSI terms) or a composed VNF, or…
  - Recursion at any level you see fit

```
CREATE NodeModel complex_vnf;
        Node input_vnf Type sample_vnf;
        Node output_vnf Type shaper_vnf;
        ConnectionPoint input;
        ConnectionPoint output
        Connection icon Type p2p Endnodes input, input_vnf:data_inside;
        Connection ocon Type p2p Endnodes output, output_vnf:wan;
        Connection intn Type p2p input_vnf:data_outside, output_vnf:lan;
```



The Current Status

- Changes to the NEMO syntax already implemented
  - OpenDaylight Beryllium release
- Extended parser with the necessary constructs
  - `NodeModel` reference to VNFD (as a URI)
  - `ConnectionPoint` construct referring to VNFD interfaces
  - `Connection` able to use `ConnectionPoint` references
- (Partial) support for recursion
  - For `NodeModel` with VNFD
  - Arbitrary `NodeModel` composition being considered
- Working in the translation to OSM descriptors
  - Full recursion would imply OSM plus ODL => SDN/NFV convergence
- We plan a demo for Prague

SUPERFLUIDITY

Acknowledgement

This work has been partially performed in the scope of the SUPERFLUIDITY project, which has received funding from the European Union Horizon 2020 research and innovation programme under grant agreement No.671566 (Research and Innovation Action).