Microservices pattern for network design

Marco Marzetti
PCCW Global (AS3491)

Who is PCCW Global?

Telecommunication provider (AS3491):

- Headquarters in Hong Kong
- Voice and Data solutions
- Global footprint

Fairly large, fairly complex network built around big routers.

Why Microservices?

Complexity and scale made changes nearly impossible.

NFV helps to simplify your architecture, but it’s an abstract concept. Microservices make it real.

Why NFV in first place?

Network architecture is typically designed to maximally leverage expensive hardware and software systems.
Services are coupled with big routers which become critical shared resources.

NFV decouples network functions from proprietary hardware appliances and runs them on standardized hardware.

Why Microservices again?

Microservices is as a way to simplify large, complicated software systems by breaking them into sub-components and distributing them across many computing servers or in the cloud.

Microservices allow the applications to be managed and coordinated over a large virtualized infrastructure.

Rui Costa to wake you up!

Please don’t tell me you’d prefered Cristiano …

NFV + Microservices

Decouple network functions from proprietary hardware.

+

Allow the applications to be managed and coordinated over a large virtualized infrastructure.

Or (in our interpretation)

Decouple network functions from ~~proprietary~~ *specific* hardware *models*.

+

Allow the ~~applications~~ *services* to be managed and coordinated over a large virtualized infrastructure.

Let’s apply it to the real world

Beloved Cisco’s *Three-tier Hierarchical Network Model* isn’t going away.

Core, *Access* and *Edge* are still there, but smaller architectural elements are *Functional Areas*, not routers.

Functional Areas

- Clusters made of at least two devices
- Provide ~~one~~ few specific functions
- Hide internal topology
- Managed by one *Controller* instance
- Enforce security at borders
- Re-use *NVF* provided by other Areas

Functional Areas (Microservices inheritance)

- Support CD/CI
- Loosely coupled
- Independently deployable
- Scale horizontally
- Developed by small teams

Functional Areas (Hardware)

Every *Functional Area* resembles a cluster.

Cheap standardized hardware doesn’t necessary mean x86. ASICs are OK.

Remember: The smaller, the simpler, the better!

So how does your architecture look like?

Four Area types identified so far:

- **Core**: IP over MPLS underlay
- **Edge**: Abstract Network Functions
- **Access**: L2 Ethernet transport
- **External Parties**: Generate traffic

Edge Areas

Abstract Network Functions doesn’t mean anything really.

The following sub-types have been defined:

- **IP Edge**: IP Transit
- **MPLS Edge**: MPLS VPNs
- **SD-WAN Edge**: OTT-alike VPNs
- **Edge Cloud**: Micro clouds

That doesn’t sound new …

It doesn’t! But we’re still applying the pattern to the existing network.

New *Edge Areas* we’re working on:

- **SPAN Areas**
- **Bridging Areas**
- **DPI Areas**
- **Traffic shaping Areas**

How do you put everything together?

Edge Areas can be attached to one Access Area only.

Compound of Edge and Acces Areas is called *Edge Islands*.

Withis Islands redundancy is provided by multiple Edge Areas of the same type.

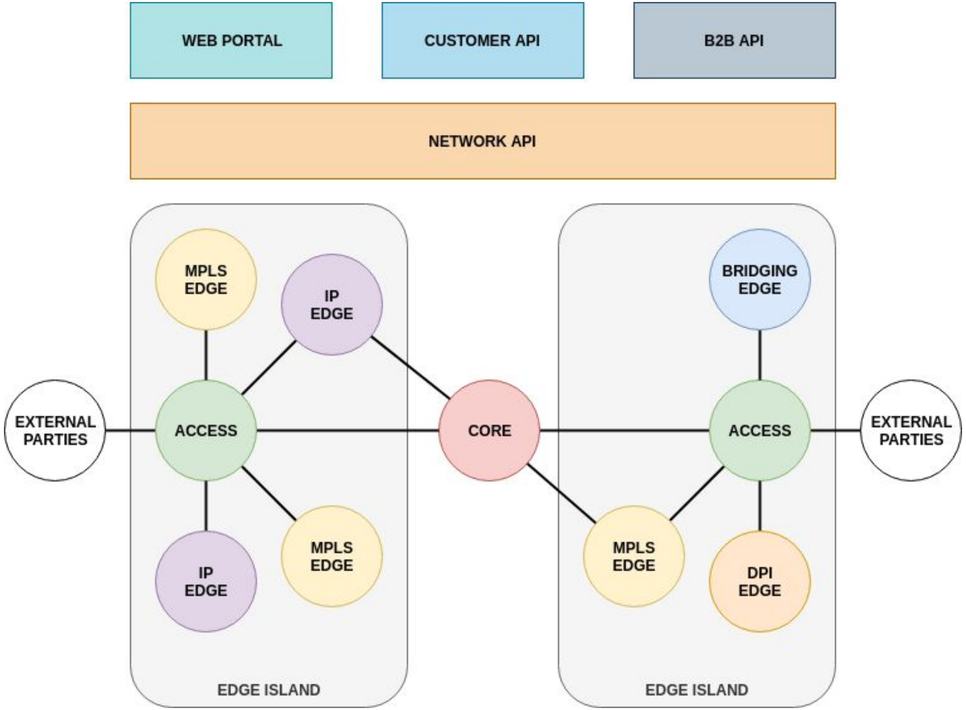
Service Chaining

Service Chaining of *VNF* is applied to *Functional Areas* via multiple software layers:

1. **Web Portals:** Receive user requests
2. **Network API***: Translates requests to chains of functions
3. **Controllers**: Create function instances

* AKA *Orchestrator*

Network Diagram



(Both Areas and Islands MAY span multiple POPs if needed)

Summary

- Create standalone Areas for each (or few) *VNF*
- Treat Areas as if they were Microservices
  (Use 1 *Controller* for each Area)
- Deploy multiple Area instances for redundancy
- Enforce ~~*Service*~~ *Function Chaining* with *Orchestrators*
- **Keep everything small and simple!**

Questions?