PCCW Global®

Microservices pattern for network design

Marco Marzetti
PCCW Global (AS3491)

HKT Here To Serve

a PCCW Group member

a PCCW Group member

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 about taking the best out of expensive hardware and software.

Services are coupled with expensive devices which become critical shared resources.

NFV decouples networking functions from proprietary hardware

Why Microservices again?

Microservices split few large complex systems into many small components

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

NFV + Microservices

Decouple networking functions from proprietary hardware.

+

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

Or (in our interpretation)

Decouple networking 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

- Made of at least two devices
- Provide few specific functions
- Hide internal topology
- Managed by dedicated *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?

Five Area types identified so far:

- **Core**: IP over MPLS underlay
- **Edge**: Abstract Network Functions
- **Access**: L2 Ethernet transport
- **Transmission**: L1 Wavelength transport
- **External Parties**: Generate traffic

Few words on 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
- **Virtualization Edge**: Edge Cloud

Few words on Virtualization Edge Areas

Nothing else but K8s or OpenStack clusters.

Run on top of regular servers (make use ASICs for network offloading).

Can be used to virtualize any Edge Area.

How do you put everything together?

Edge Areas can be attached to one Access Area only.

External parties are connected to Access or Transmission Areas only.

Redundancy is provided by redirecting External Parties to other devices of the same Edge Area during failure.

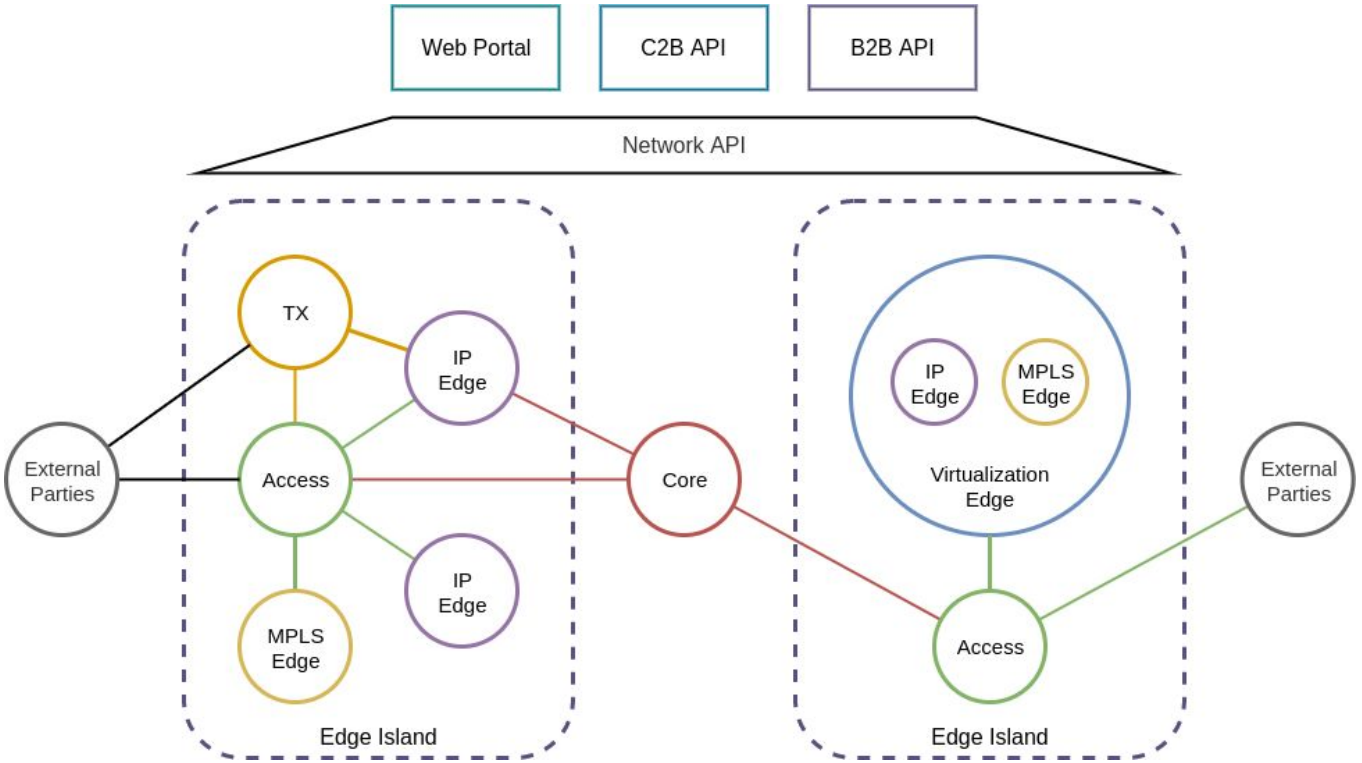
What about Services?

Services are chain of functions delivered via multiple software layers:

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

* 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 dedicated *Controller* for each Area)
- Deploy multiple Area instances for redundancy
- Enforce ~~*Service*~~ *Function Chaining* with *Orchestrators*
- **Keep everything small and simple!**

Questions?