SDN: A NEW PARADIGM

Kireeti Kompella
CTO, JDI

AGENDA

What is SDN?

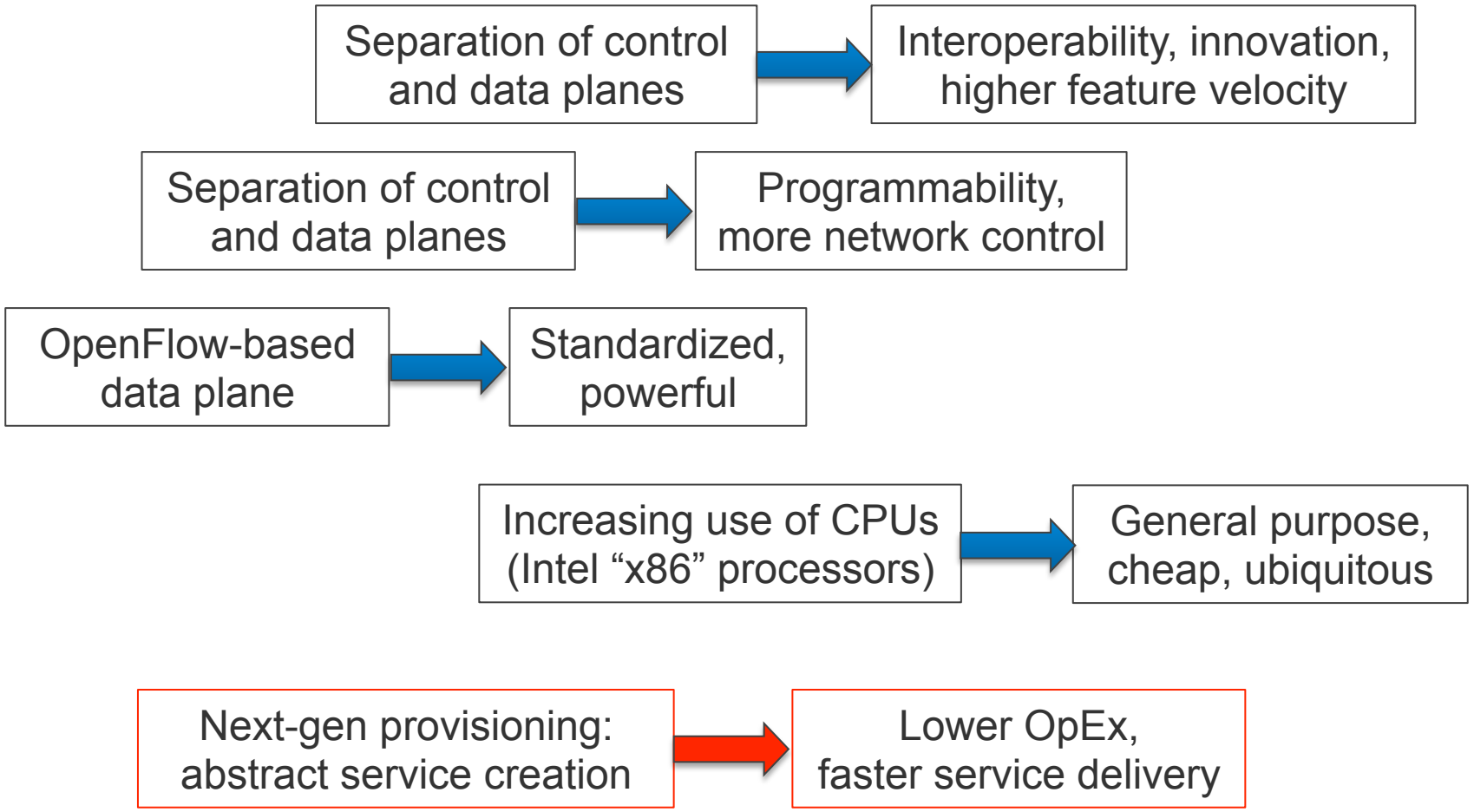
- Definition and goals of SDN
- Analogy with Compute Virtualization
- Orchestration for Agile Service Provisioning
- Unified SDN

What parts of the network does SDN touch?

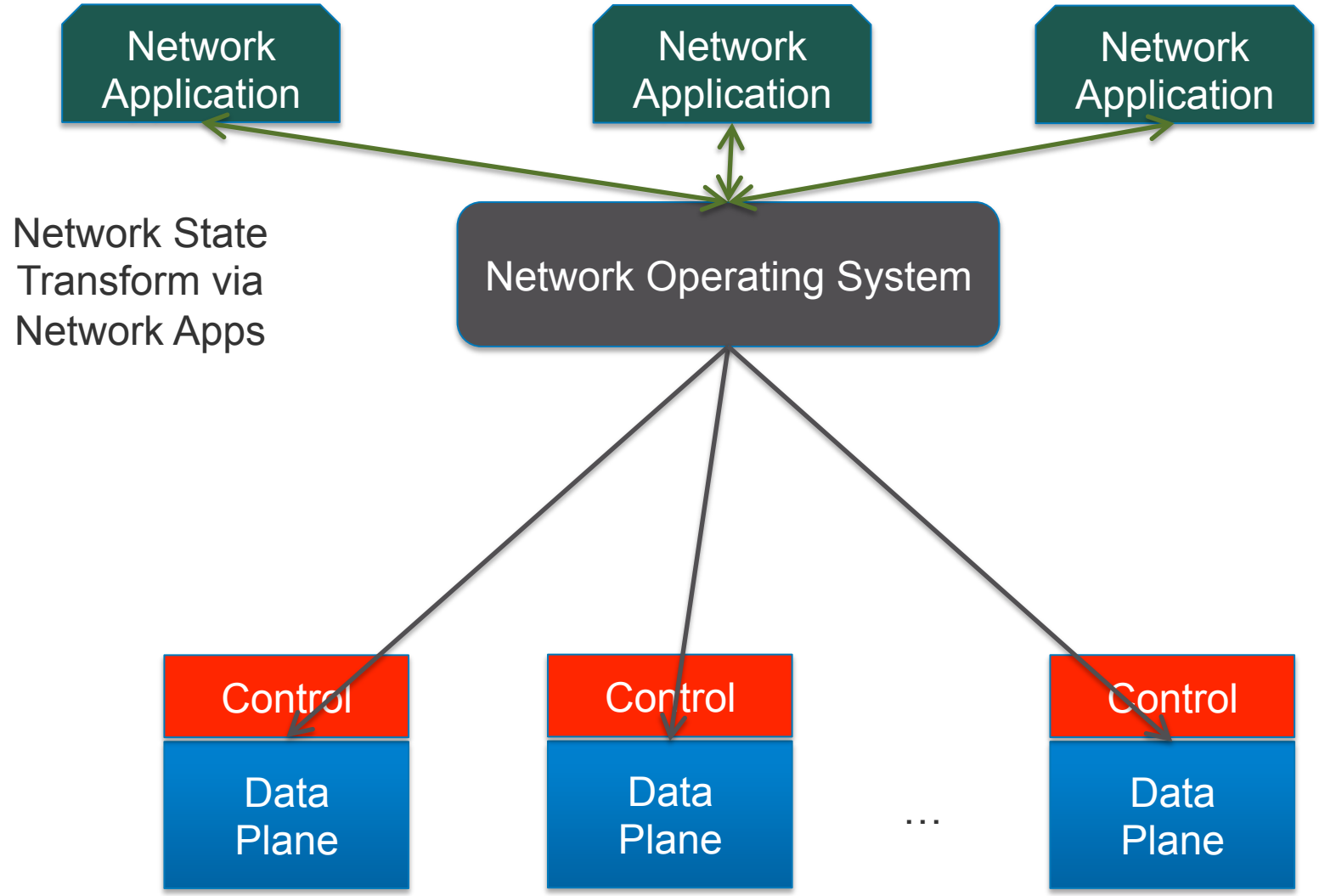
Conclusion

DEFINITION AND GOALS OF SDN

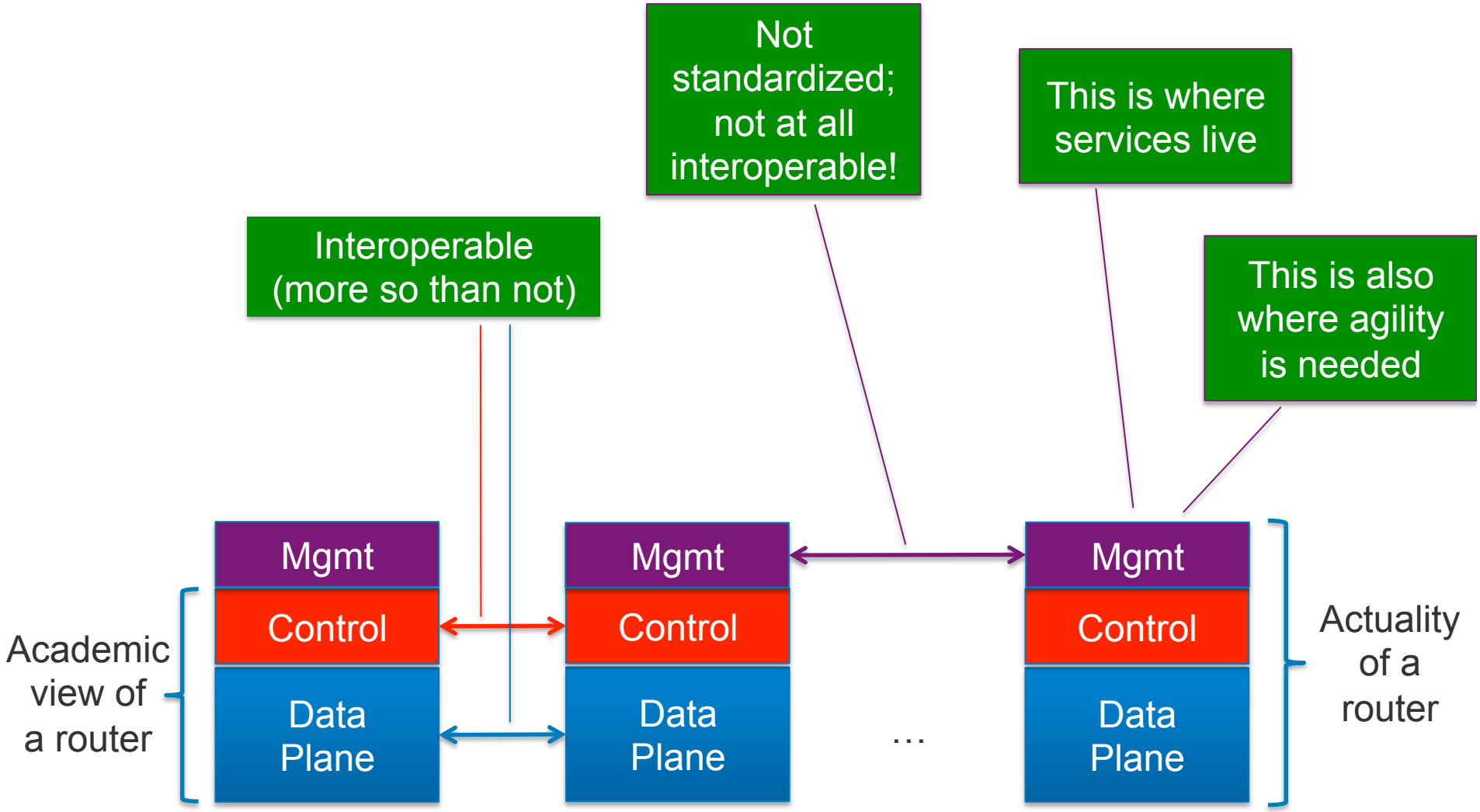
SDN has many definitions and many goals



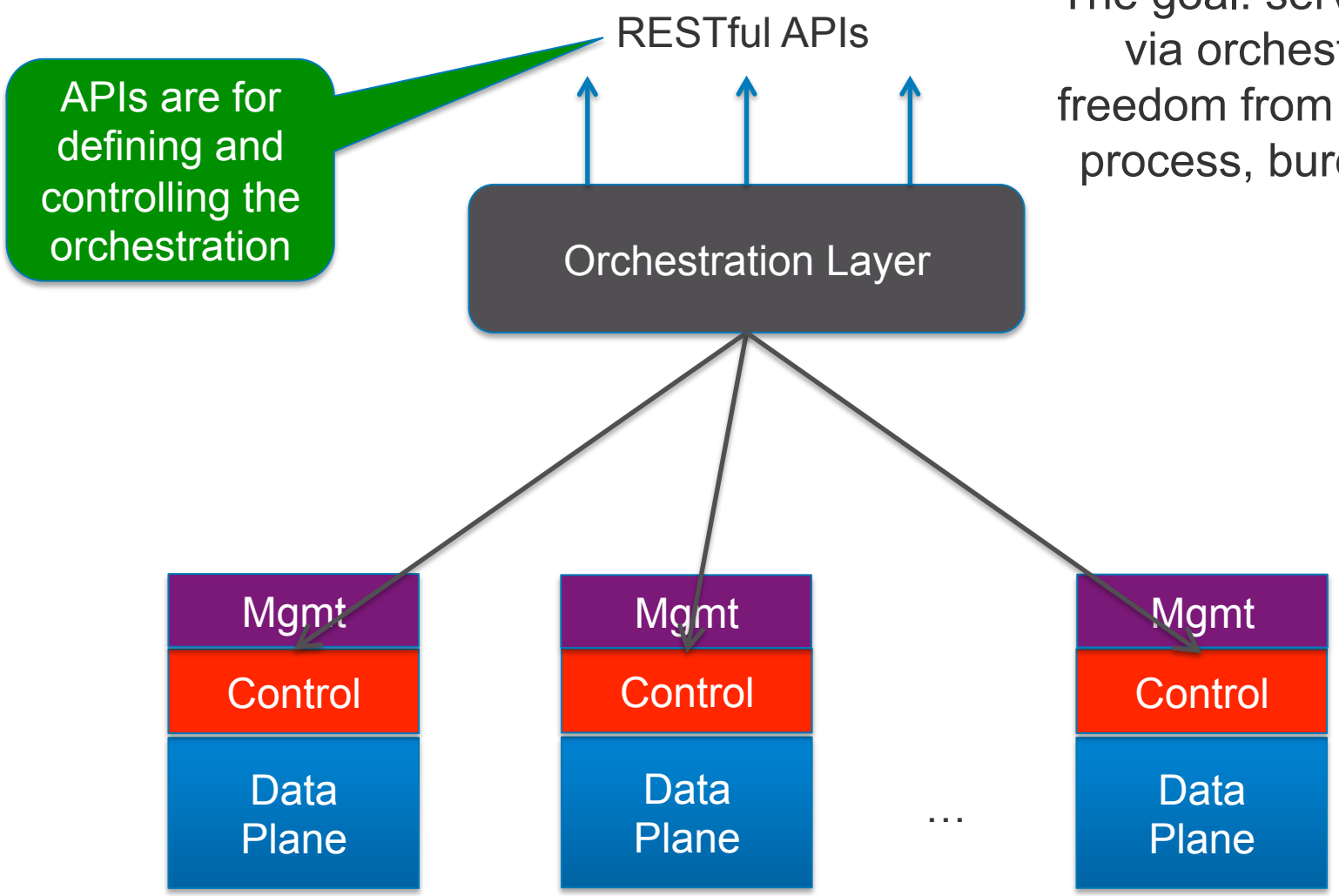
SEPARATION OF CONTROL AND DATA PLANES CENTRALIZATION OF CONTROL PLANE → "NOS"



IS A ROUTER JUST CONTROL AND DATA PLANE?



RESTORING AGILITY: SEPARATE MANAGEMENT FROM REST OF ROUTER



The goal: service agility via orchestration: freedom from “physics”, process, bureaucracy

ORCHESTRATION = *AGILE SERVICE PROVISIONING*

Just as in Compute Virtualization, so in networks: we need the ability to *orchestrate* and *automate*

- In this case, service creation and management

This offers greater network control and speeds up service rollout

- Dual effect of an increase in revenues and lower OpEx
- It will also improve resource efficiency, leading to CapEx savings

But how is this different from current Network Management Systems, or OSSs?

DISCUSSION: WHY PROGRAMMABILITY?

Cool!
New control plane!

really?

ANALOGY: COMPUTE VIRTUALIZATION

First came the technology (actually, quite long ago)

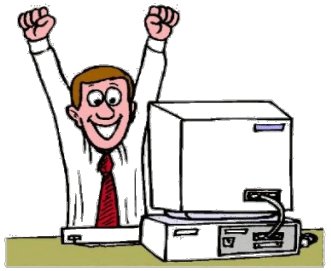
- "Let's emulate virtual CPUs on a physical CPU"

Then came orchestration and automation

- Bring up manage, terminate, new VMs

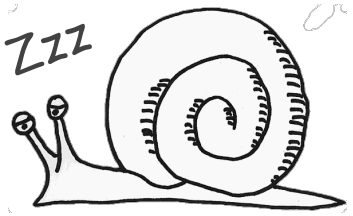
Finally, came the realization

- "I'm free from the constraints of physics!"
- and the constraints of process, bureaucracy



Instead of a few months to deploy an application

- agonize over how many servers are needed, purchase, rack and power them, connect them up



… it can take a few hours, or even a few minutes

- especially with the help of automation

PROGRAMMABILITY IS MOSTLY ABOUT AUTOMATION

Automation allows repetitive tasks to be done easily, quickly and in an error-free manner

Automation allows pre-programmed responses to events:

- faults, congestion, bandwidth surges, bandwidth requests, …

Automation allows humans to offload what they don't do well …

- quick, accurate responses to anticipated events

… to get on with what humans do best:

- defining new services, defining policies, monitoring for unanticipated events and providing customized responses

This means that the ***language*** in which this automation is expressed needs to be suited for the purpose

DISCUSSION: WHY PROGRAMMABILITY?

Practical, efficient, gets the job done

practical,
efficient,
sexy-ish

SERVICE ABSTRACTION

Service definition is based on **abstract** information models

- These are **high-level**: device and OS and version independent
- They are **standardized**, but allow for provider-specific enhancements
- Service deployment is **transformation** of an abstract service definition to device-specific data models

In this system, service deployment will be:

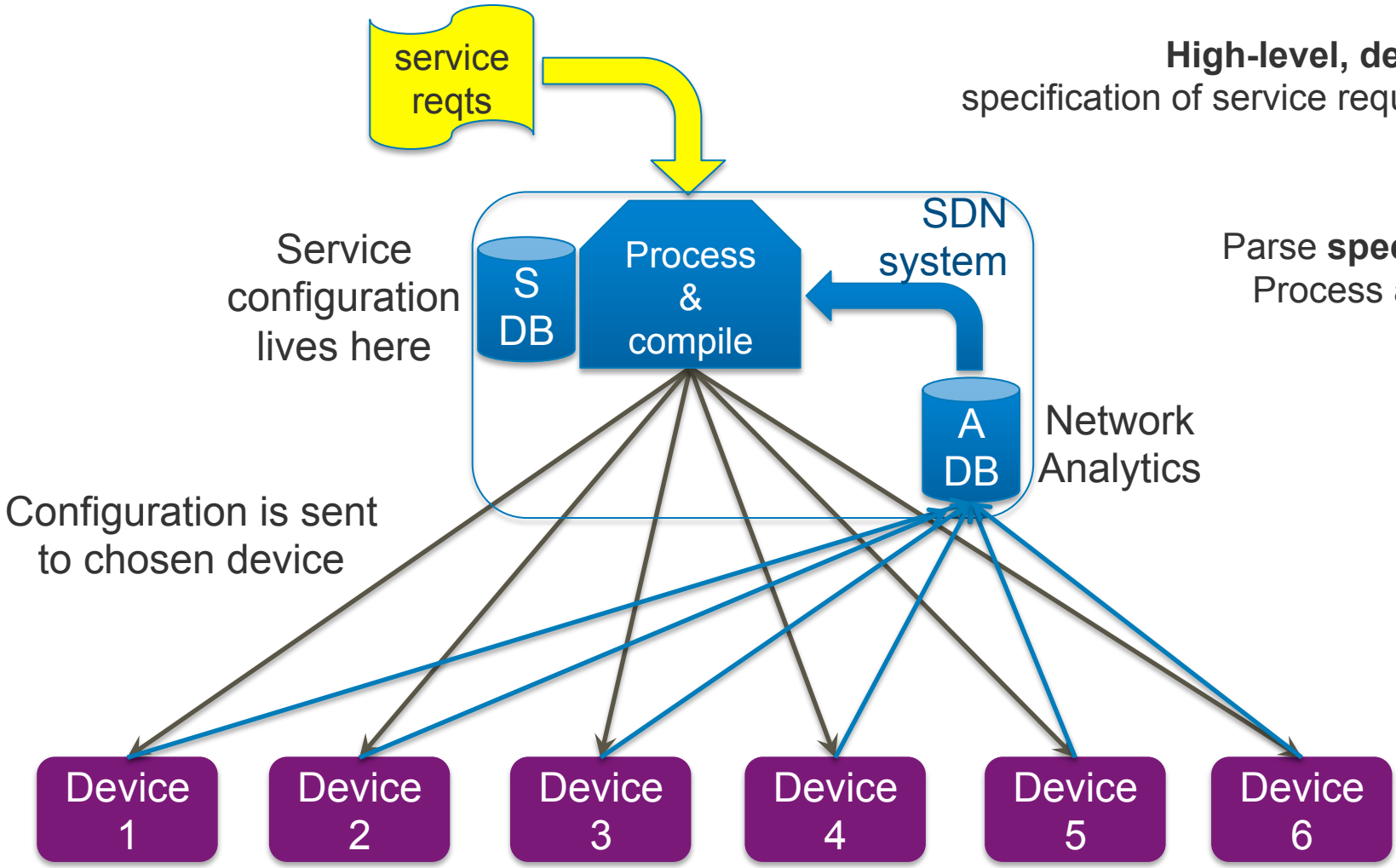
- **Fungible** – I can use a given device for many services
- **Flexible** – I can deploy a service at many devices (placement)
- **Fast** – I can roll out a service quickly, redeploy quickly

Provisioning

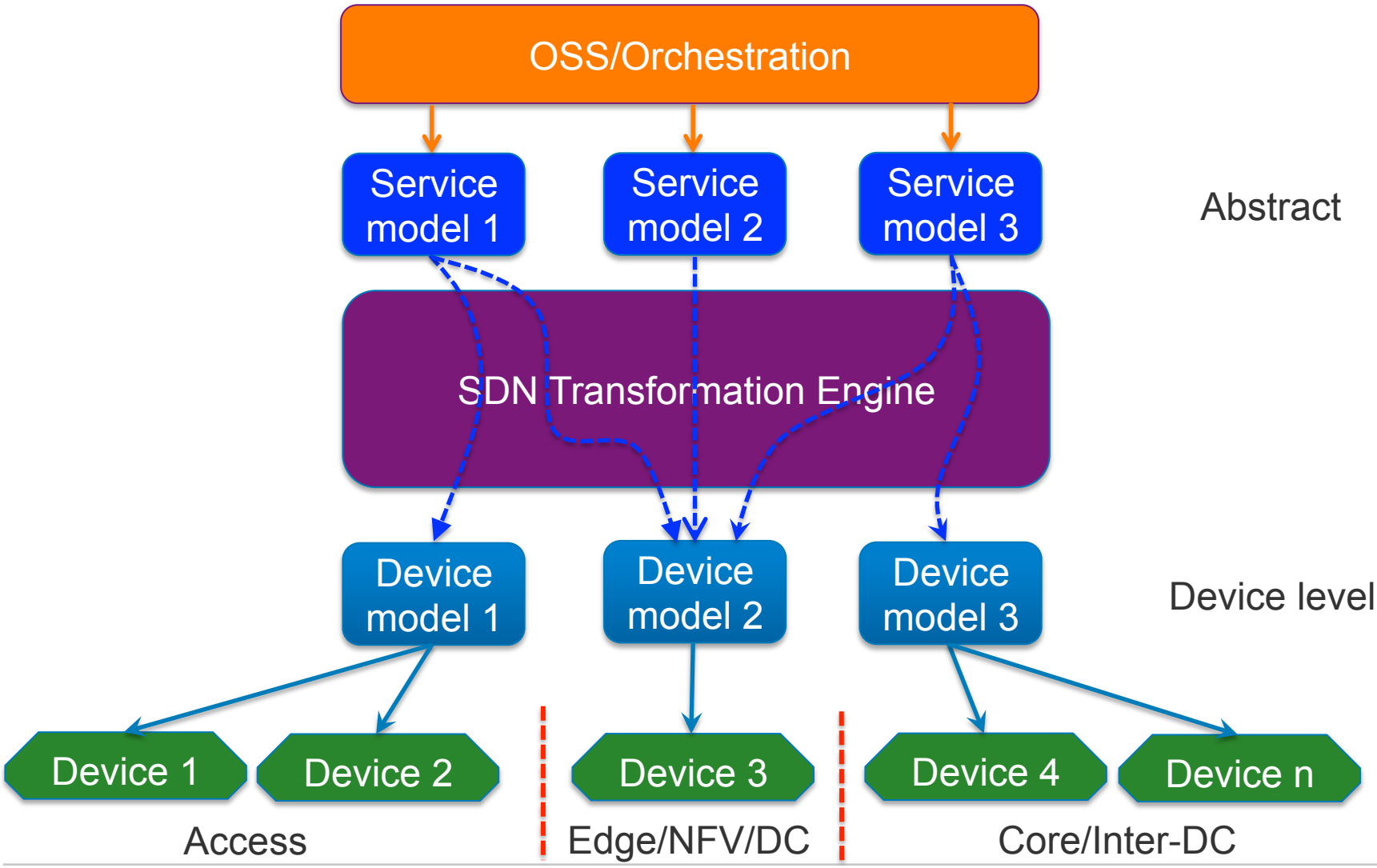
- **Responsive** – the service adapts dynamically to changes

Analytics and Automation

SDN AS A COMPILER
SAY *WHAT* YOU WANT, NOT *HOW* TO DO IT



UNIFIED SDN



CUSTOMIZATION OF SERVICE MODELS AND DEVICE MODELS

aka programmability

Standard model for some service

Service Template

Customized service model for use case

Customized Service Model

Service Instance for some customer

Customized Service Instance

Standard device level model for service

Base device model

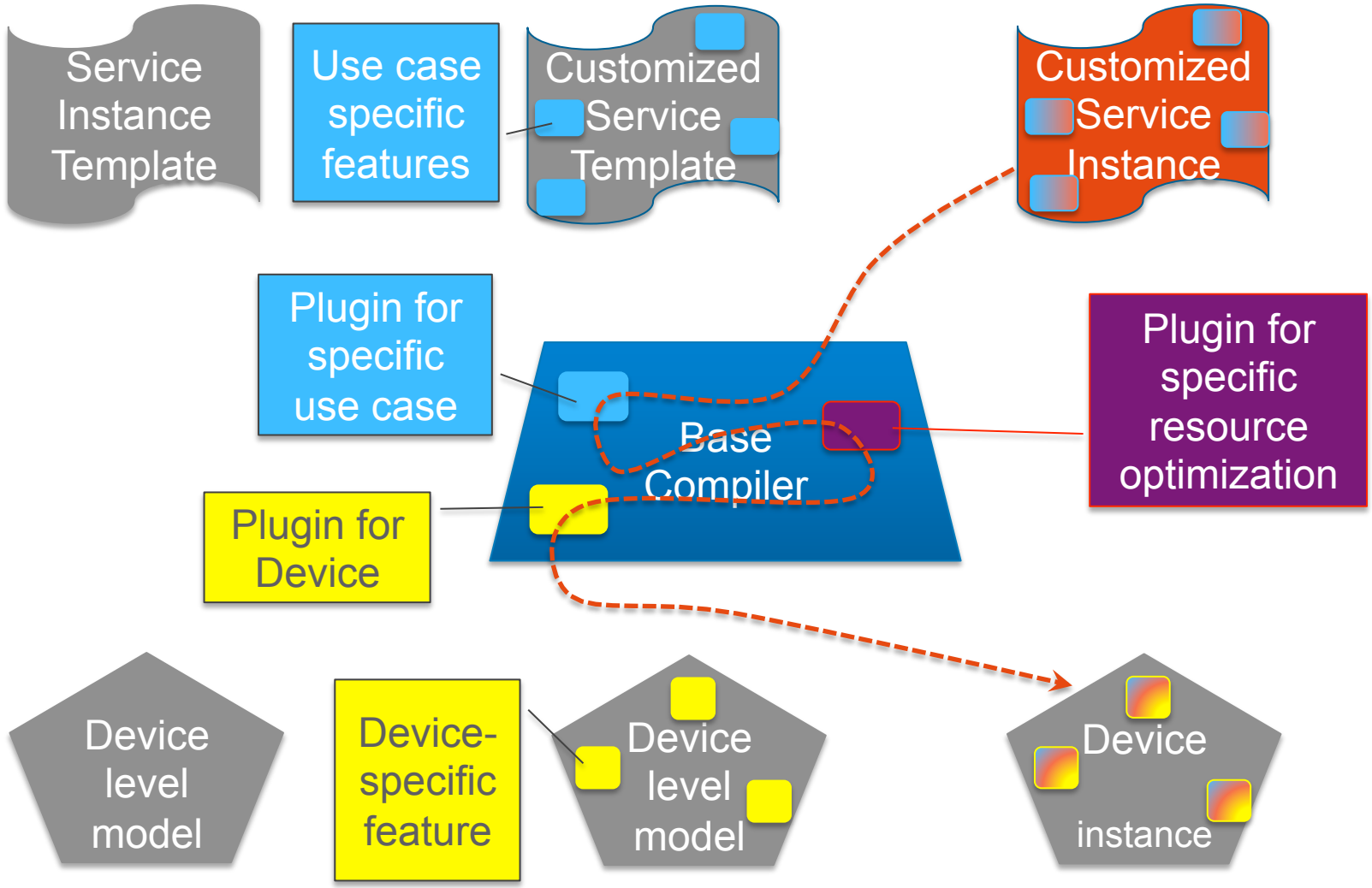
Vendor-customized device level model

Customized device model

Device model for service instance

Device instance

SDN AS A COMPILER WITH PLUGINS TO HANDLE CUSTOMIZATIONS



AGENDA

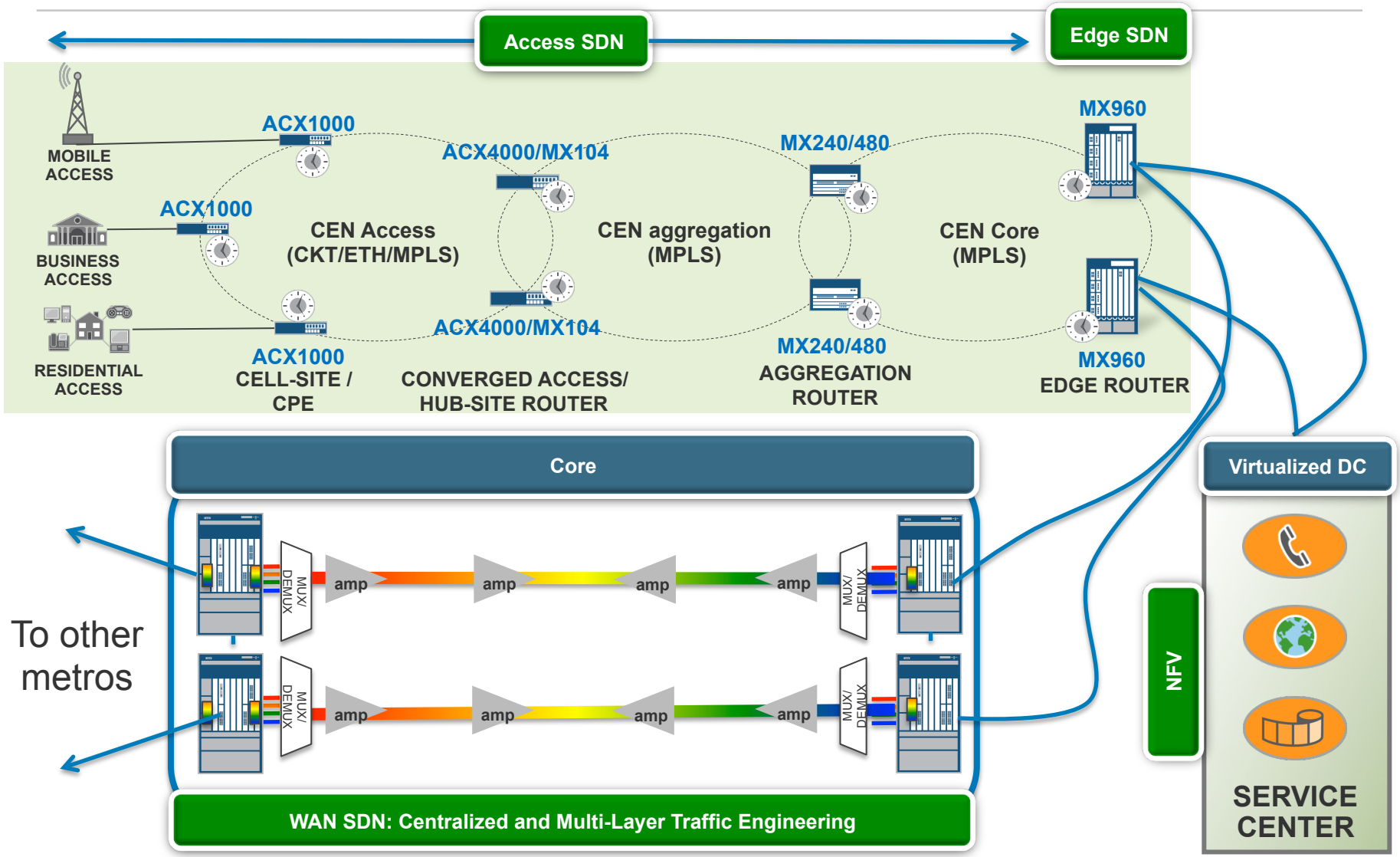
What is SDN?

What parts of the network does SDN touch?

- Virtualized Data Centers
- SDN for NFV
- Core SDN
- Access/Edge SDN

Conclusion

WHAT PARTS OF THE NETWORK DOES SDN TOUCH?

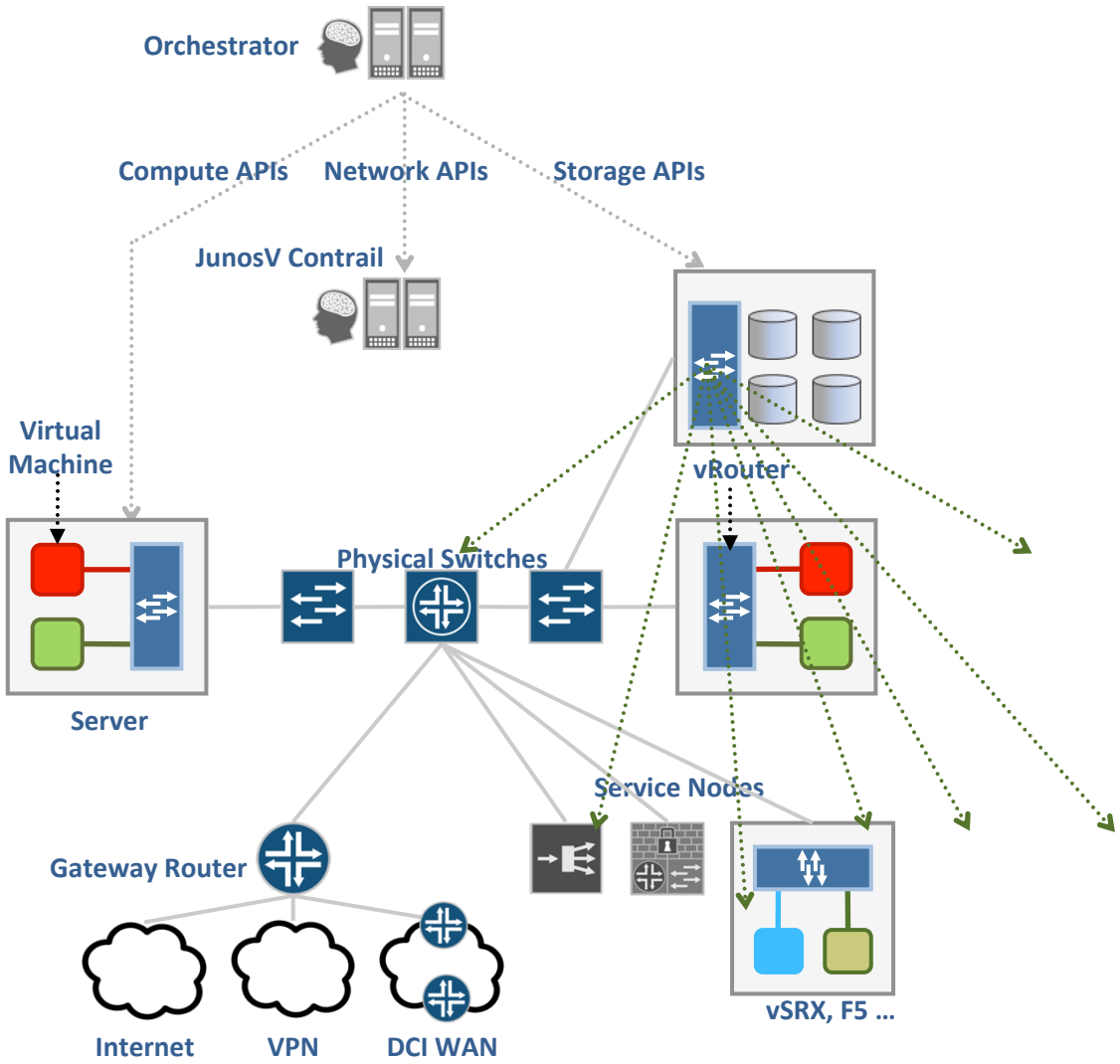


SDN FOR VIRTUALIZED DATA CENTRE

Orchestration of Compute (Virtual Machines) in a Virtualized DC is mature

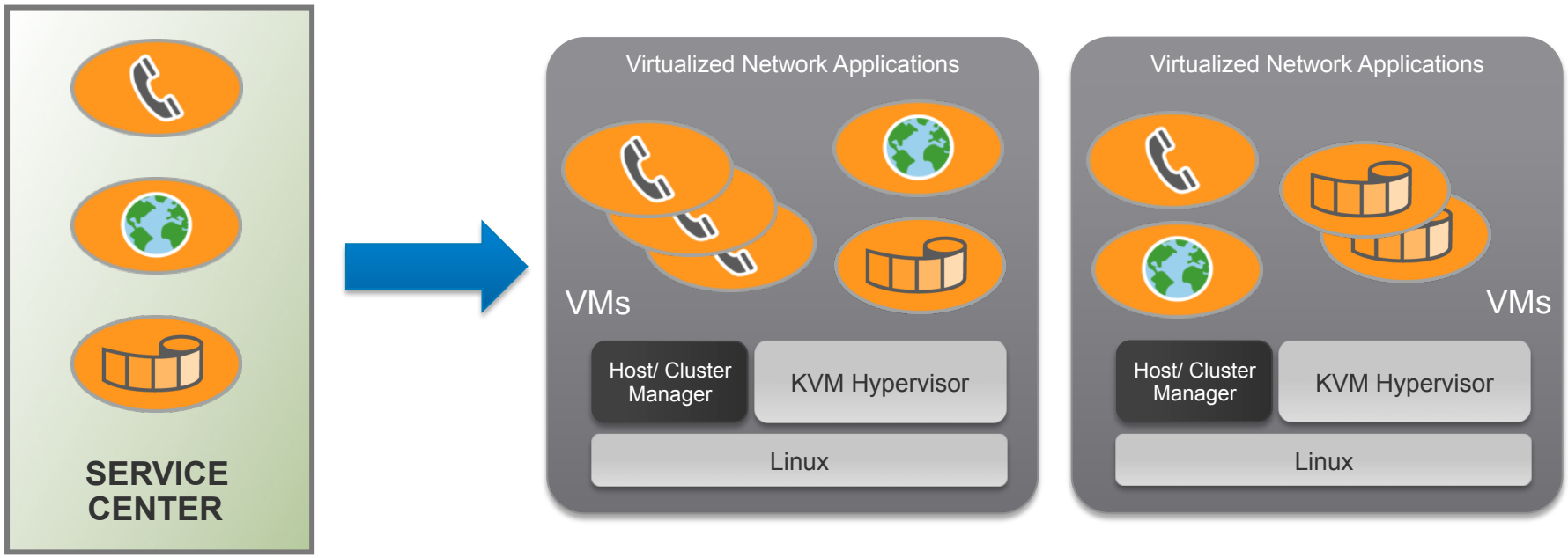
One can, via the Contrail controller, define “Virtual Networks” (VNs), inter-VN policies and service chains, as high-level data models

When a VM is placed on a server, the Network Orchestrator calls into to Contrail to connect the VM to its VN, and has all the policies it needs

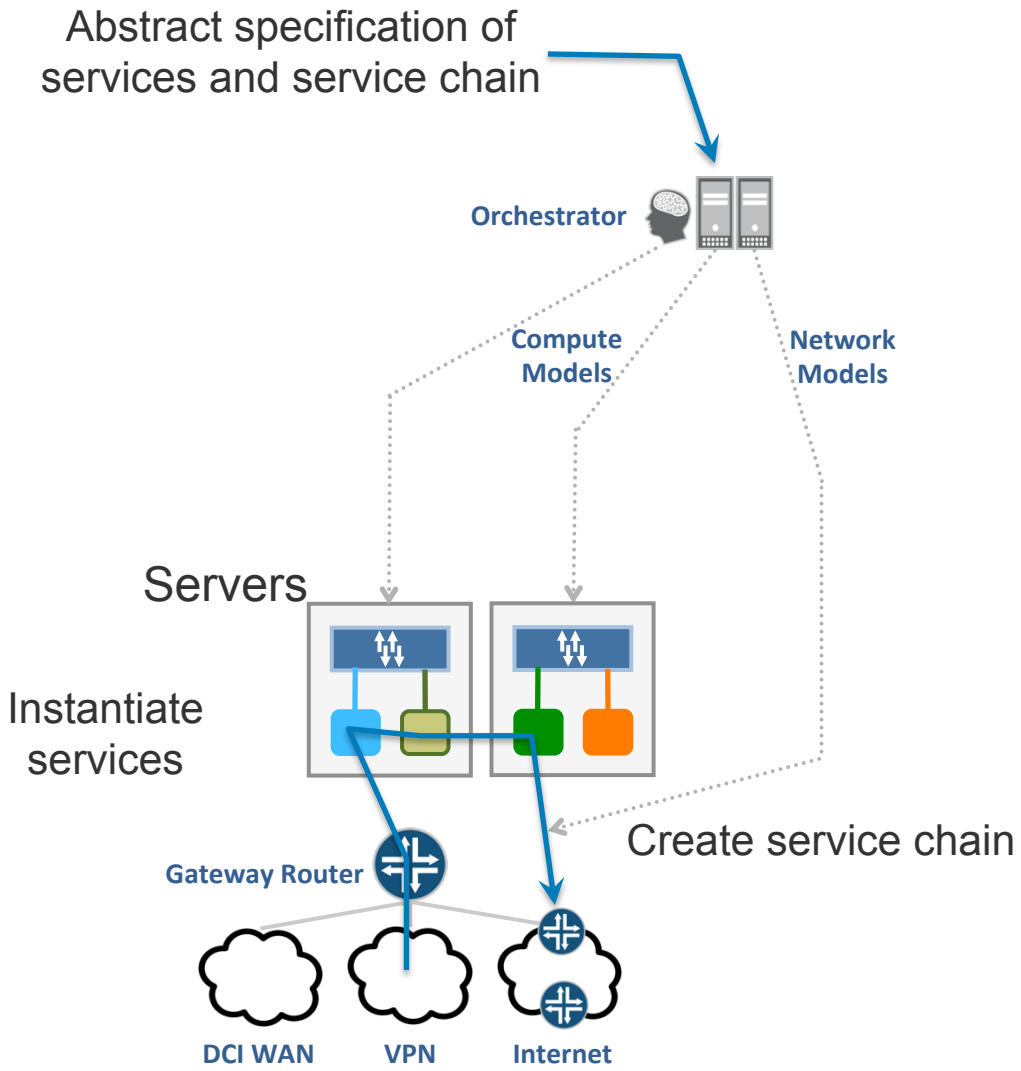


NETWORK FUNCTION VIRTUALIZATION

NFV is the use of general purpose servers instead of specialized devices for network functions such as voice gateways, video encoders, IMS, MME, firewalls, DPI, etc.

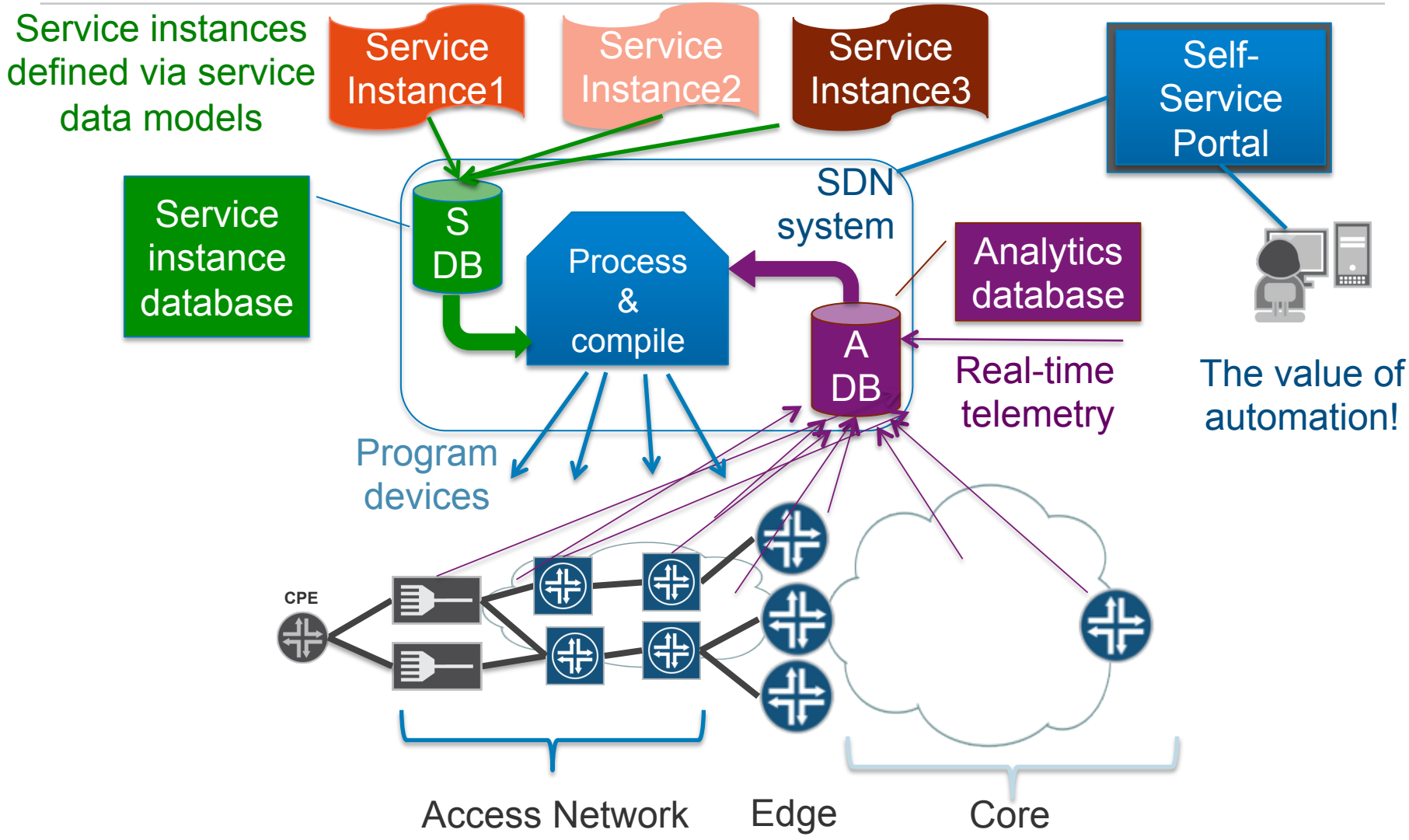


NFV ORCHESTRATION



Specification of services includes details of **instances** and **config**; and details of the connecting **service chain**

SDN FOR THE EDGE



CORE (OR WAN) SDN: PATH OPTIMIZATION

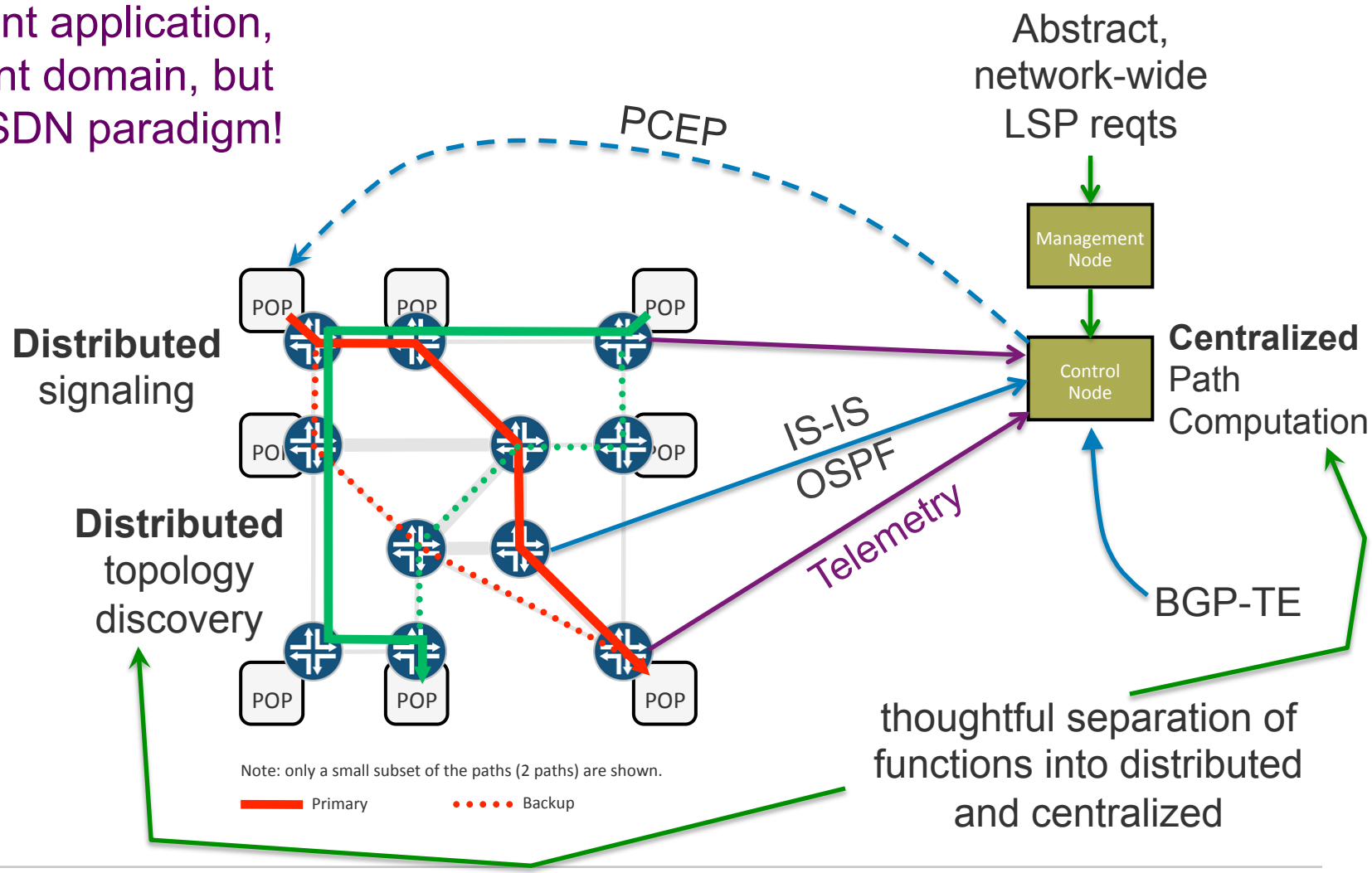
SDN in the core of the network focuses on optimized multi-layer Traffic Engineering based on centralized path computation

There is a thoughtful separation of functions into those that are distributed and those that are centralized

The goal is to have a higher-level view of TE, and to make this TE serve the needs of services

CORE SDN: MULTI-LAYER PATH COMPUTATION & BANDWIDTH CALENDARING

Different application, different domain, but same SDN paradigm!



CONCLUSION: THE SDN PARADIGM

The paradigm of SDN as a Network Operating System to form the basis for network programming is *too low-level*

- This attacks the problem of new control plane implementations
- Worthwhile problem for some, but not for all

The paradigm of SDN as a Compiler for provisioning via **abstract** service models is a **high-level, declarative** approach

- This attacks the problem of service provisioning
- Real problem for most who provide and manage network services

This paradigm is **standards-based** while allowing for provider-specific enhancements

This paradigm applies to **all parts of the network**: DC, inter-DC, access, edge, NFV, core

everywhere