CORD: Putting the Pedal to the Metal

Larry Peterson
ONF

#OpenCORD

CORD
Central Office Re-architected as a Datacenter

CORD and Car Analogies

CORD is like a car in that it is an integrated whole, as opposed to a pile of parts that must be assembled.

→ *Yes, but it doesn't just use off-the-shelf components.*

CORD is like a concept car.

→ *Yes, but it's built to be driven on the street, not just admired in the showroom.*

Halo Car

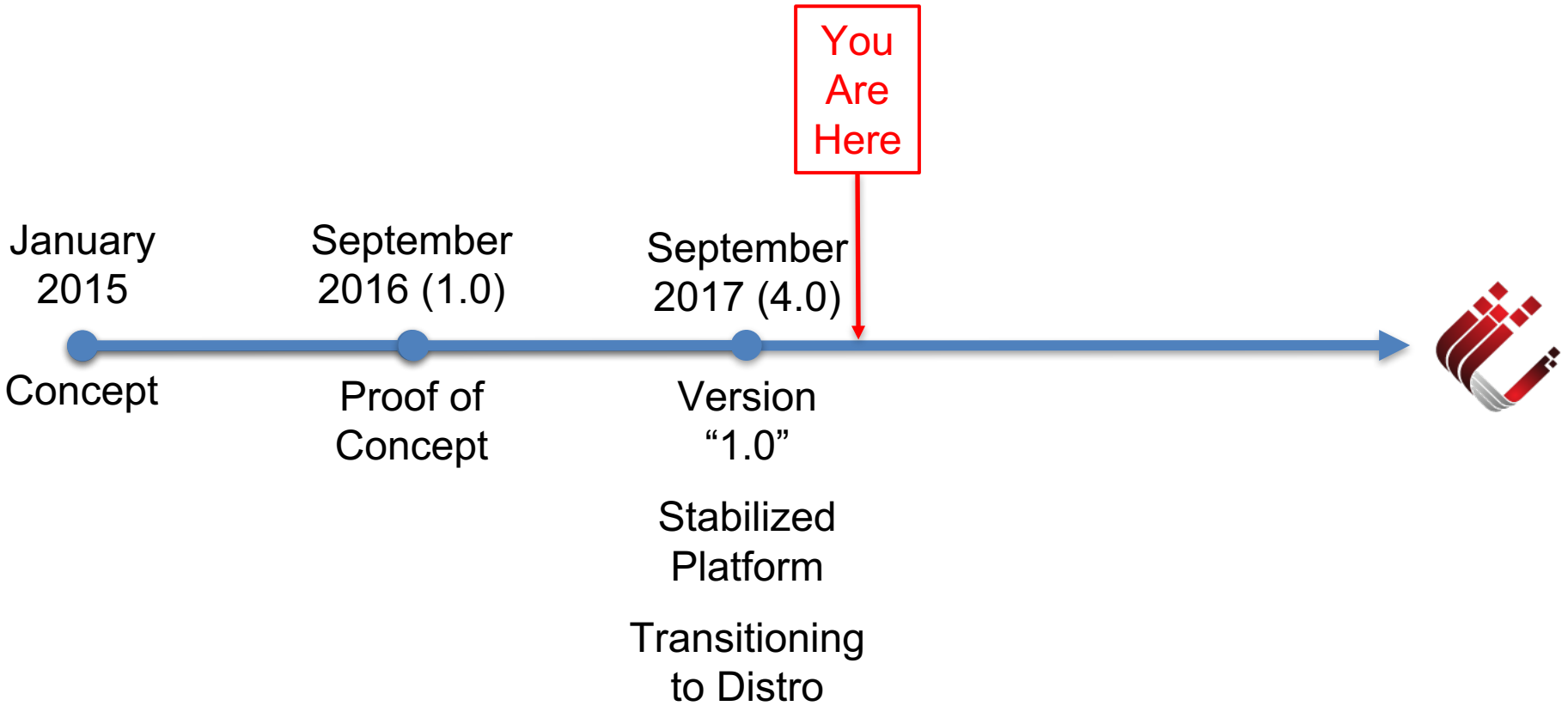
"…goes out of its way to push the technology…"

"…spurs the imagination…"

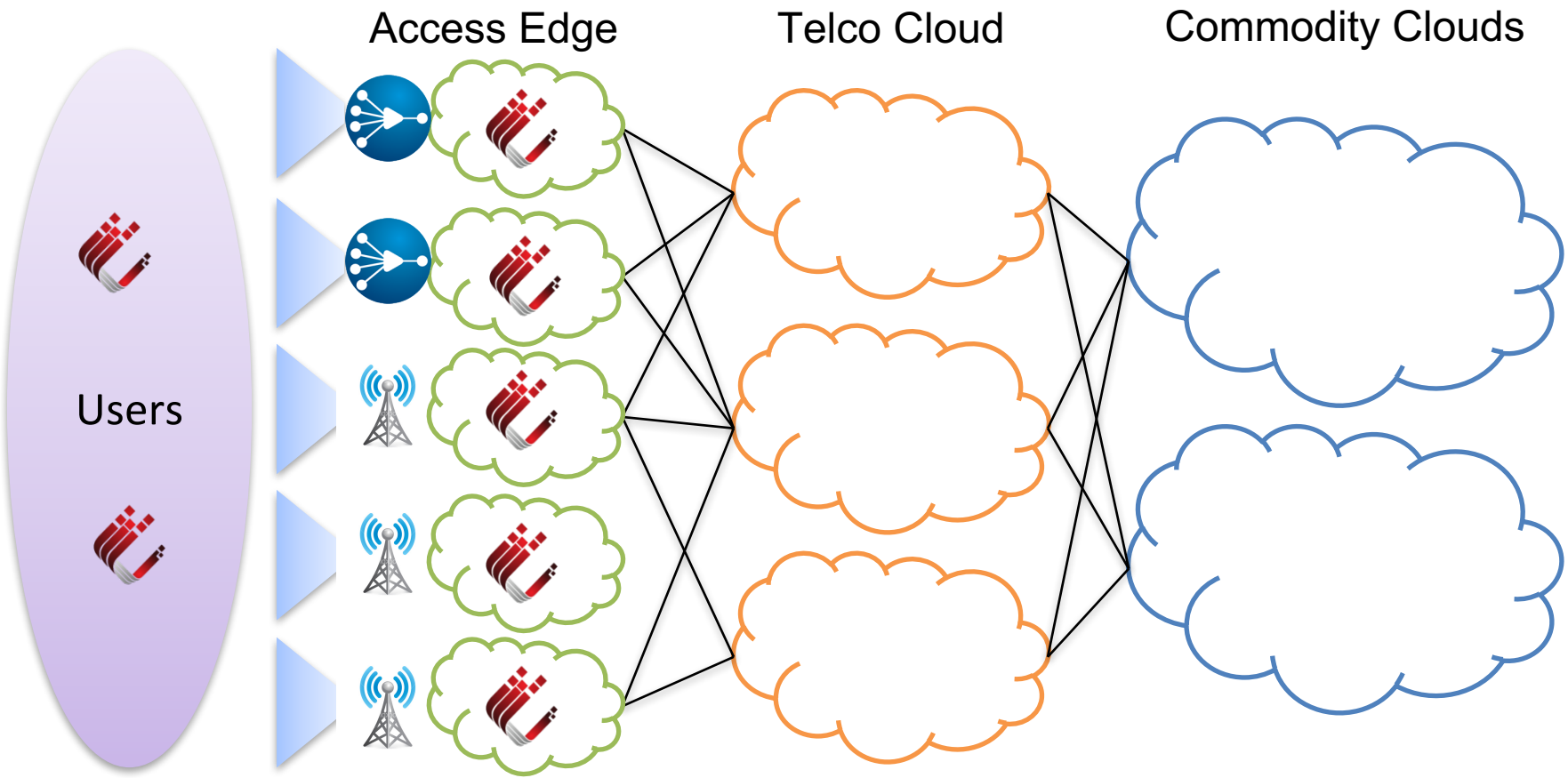
''...closest thing to a concept car you can find on the street…"

"…the technology developed is often incorporated into future production vehicles…"

CORD: A Community Halo Project



Edge Service Delivery Platform



Architectural Requirements

1) Built around **commodity servers** and **white-box switches**, and to the extent possible, leverages merchant silicon.

2) Enables **disaggregation**, and is not restricted to running bundled legacy VNFs in virtual machines.

3) Leverages **SDN** to both interconnect the virtual and physical elements and as a source of **innovative services**.

4) Leverages an **extensible platform** that can be configured to include multiple access technologies and services.

5) Adopts best practices in building, composing, and operating **scalable multi-tenant cloud services**.

Architectural Requirements

(1) Built using commodity servers and white-box switches

- Leverage merchant silicon
- Achieve performance and reliability in software

(2) Support a wide range of services

- Bundled Legacy and Disaggregated Greenfield
- Server-based (NFV) and Switch-based (SDN)

(3) Built as an extensible platform

- Scale hardware up/down to meet performance requirements
- Select access devices and services to meet functional needs
- Leverage declarative models to configure and control

Architectural Requirements

(4) Support multi-tenancy

- Platform isolates multiple business units and service vendors
- Each service isolates multiple end-users (subscribers)

(5) Operationally robust

- Adopts best practices in scalable cloud design
- Supports zero-touch provisioning

Pushing Technology Boundaries

Merchant Silicon

x

Disaggregation

x

Extensible Platform

x

Multi-Tenant

Bring Cloud Technology to the Access Network

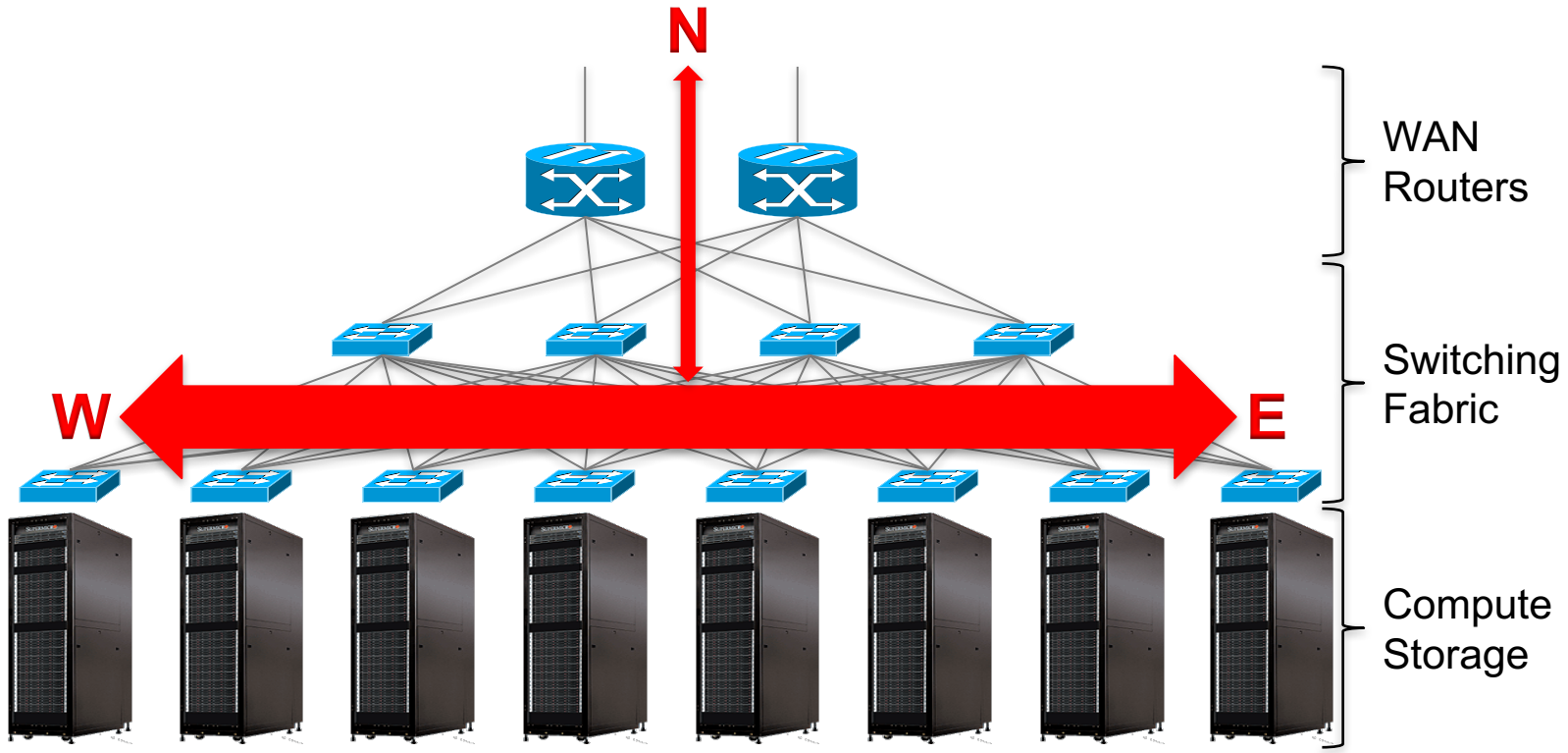
Bring Access Technology to the Cloud

Edge Service Delivery Platform

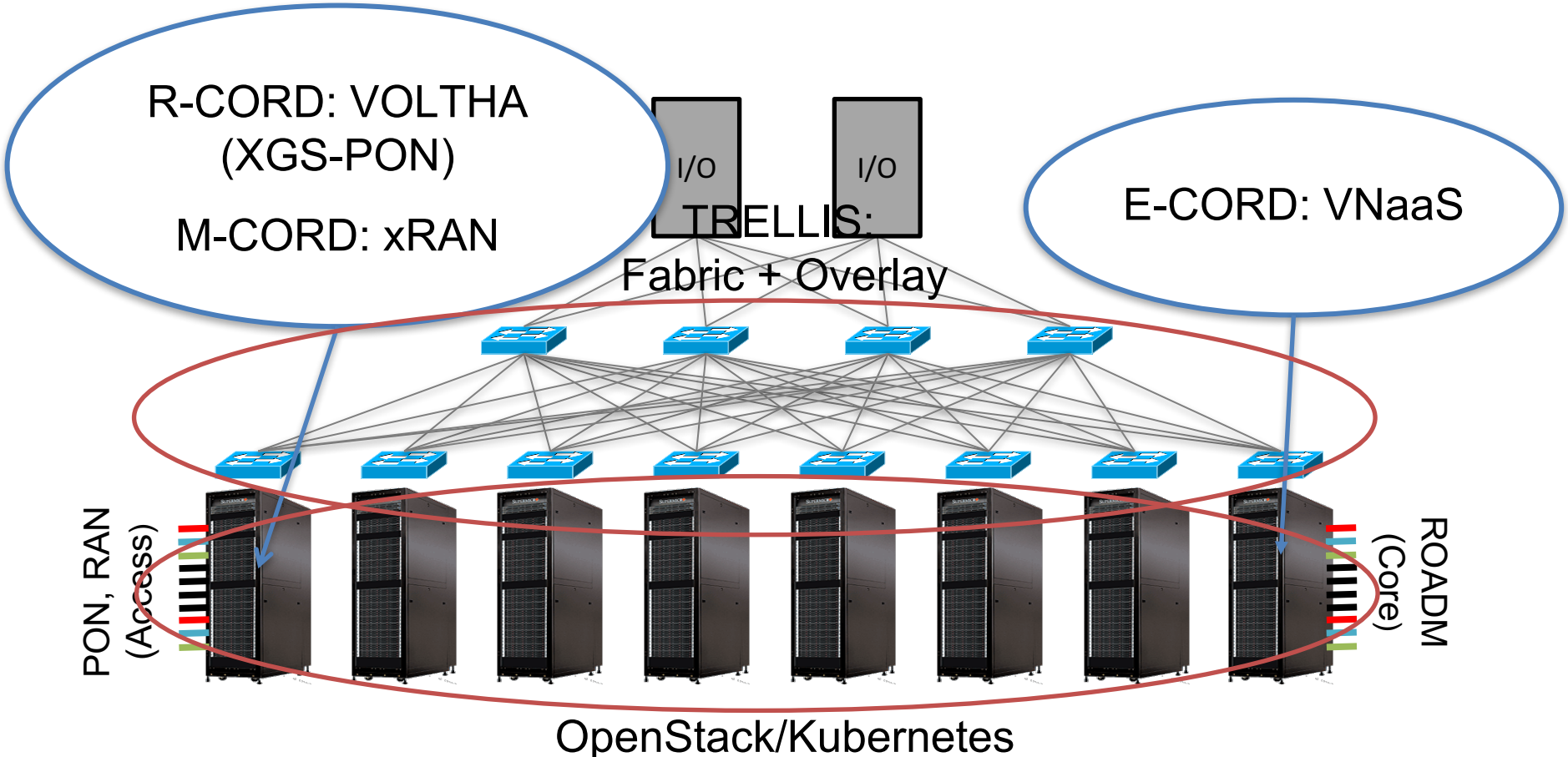
Bring Cloud Technology to
the Access Network

Bring Access Technology to
the Edge Cloud

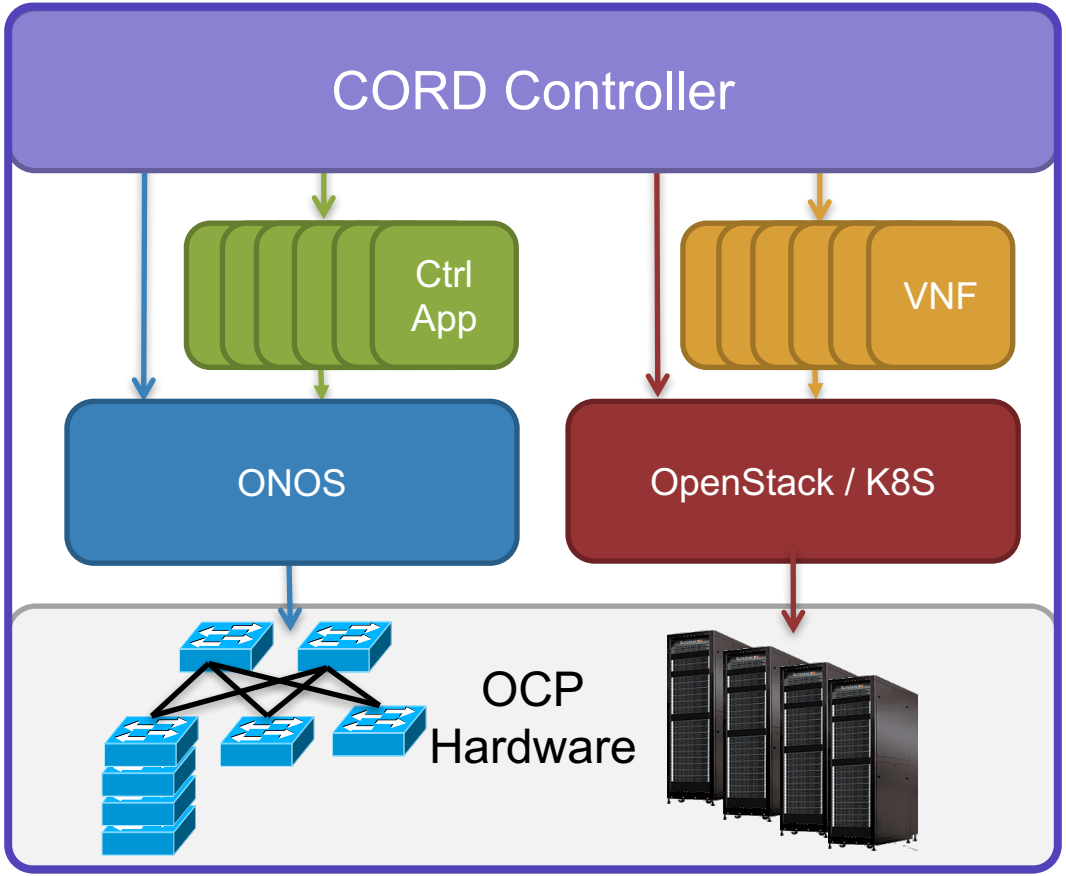
Data Center



Access Edge


R-CORD: VOLTHA
(XGS-PON)
M-CORD: xRAN
I/O
I/O
TRELLIS:
Fabric + Overlay
E-CORD: VNaaS
PON, RAN
(Access)
ROADM
(Core)
OpenStack/Kubernetes

CORD Architecture

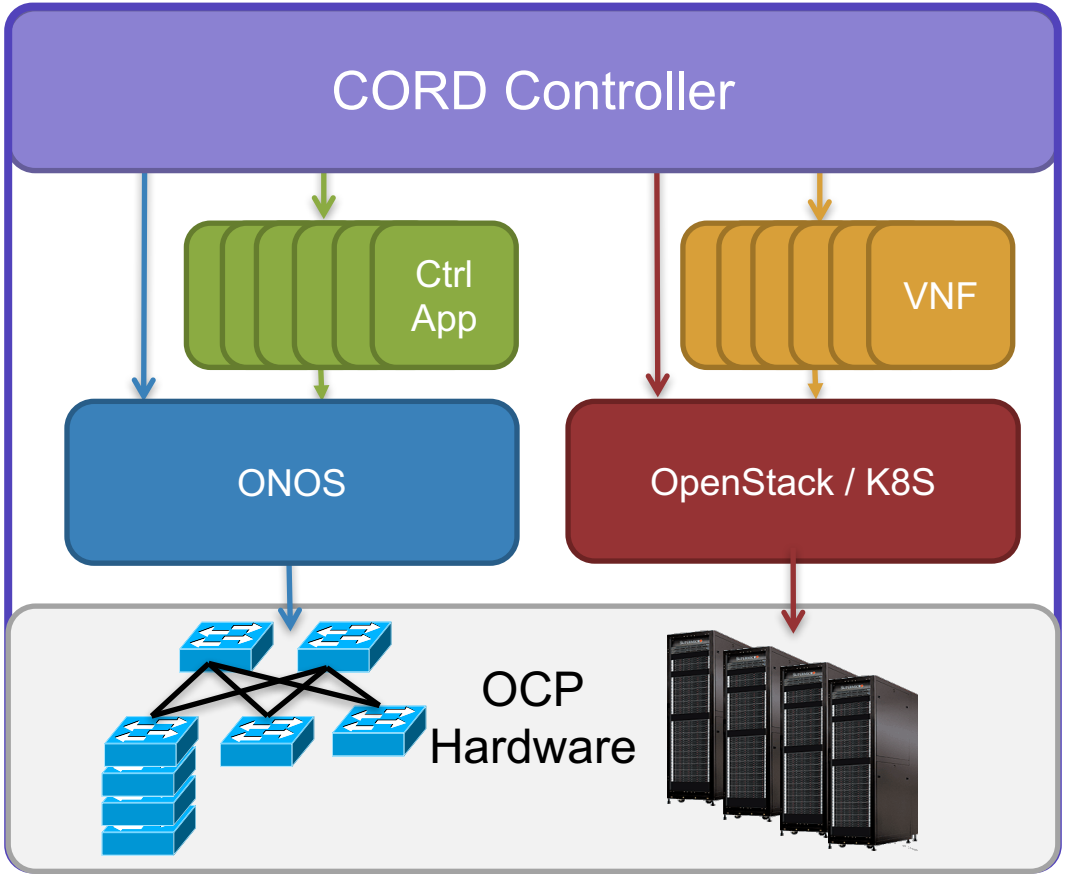


(Req 1)

Commodity Servers
White-Box Switches
Merchant Silicon Access

CORD Architecture

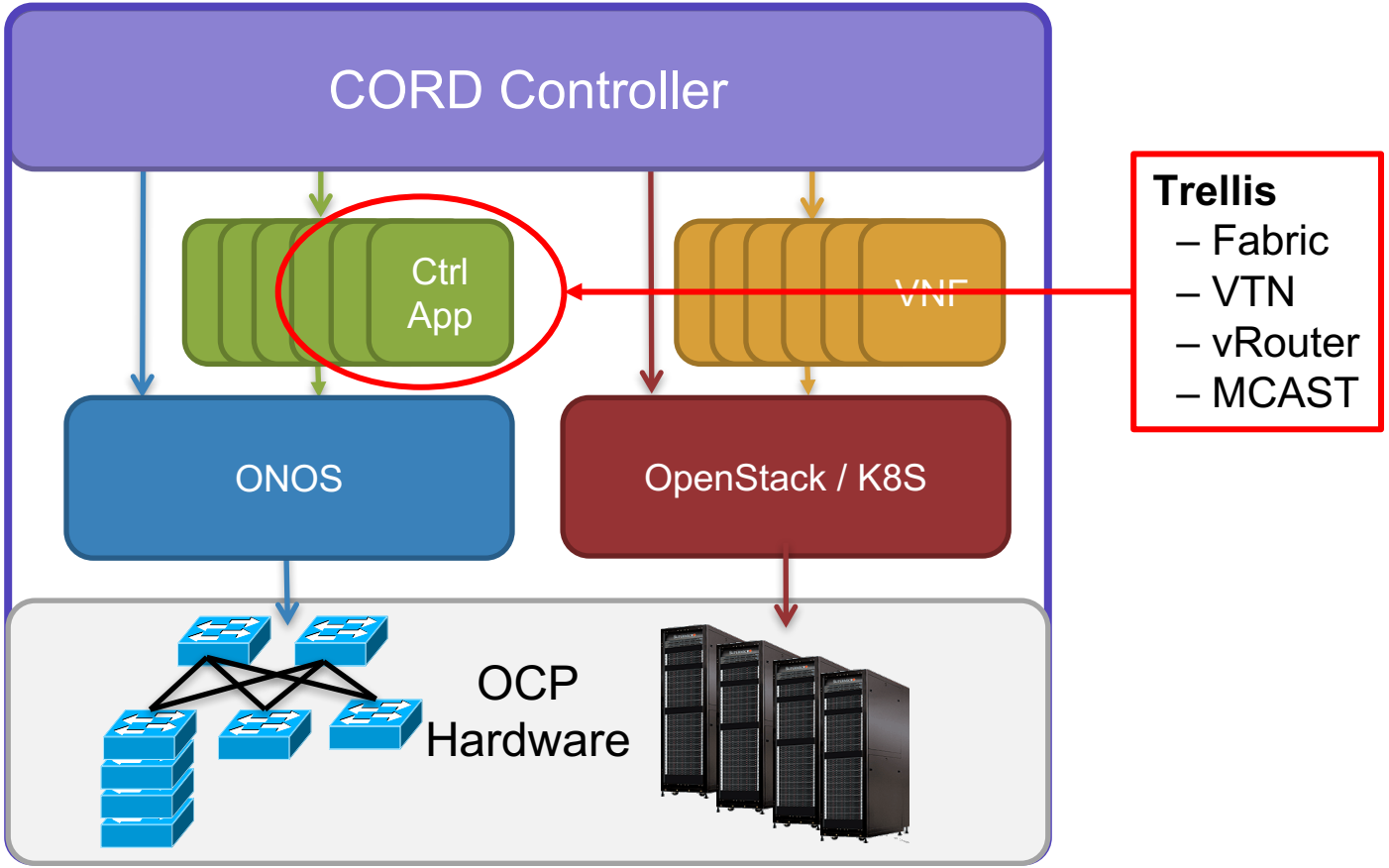
(Req 2,5)



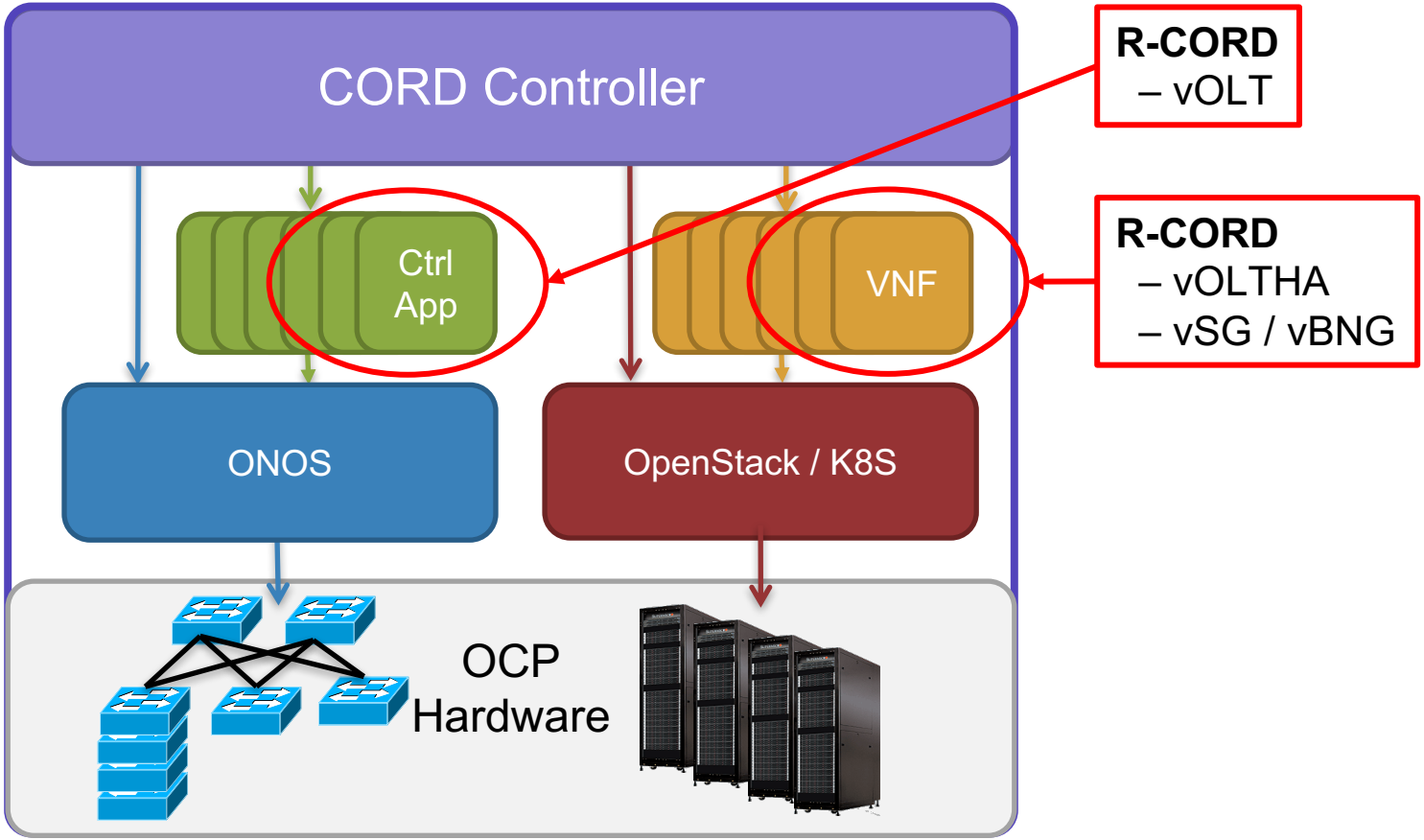
Virtualization Agnostic
- VMs
- Containers
- Micro-Services
- …

Instruction Set Agnostic
- Server-based (VNF)
- Switch-based (SDN)
- …

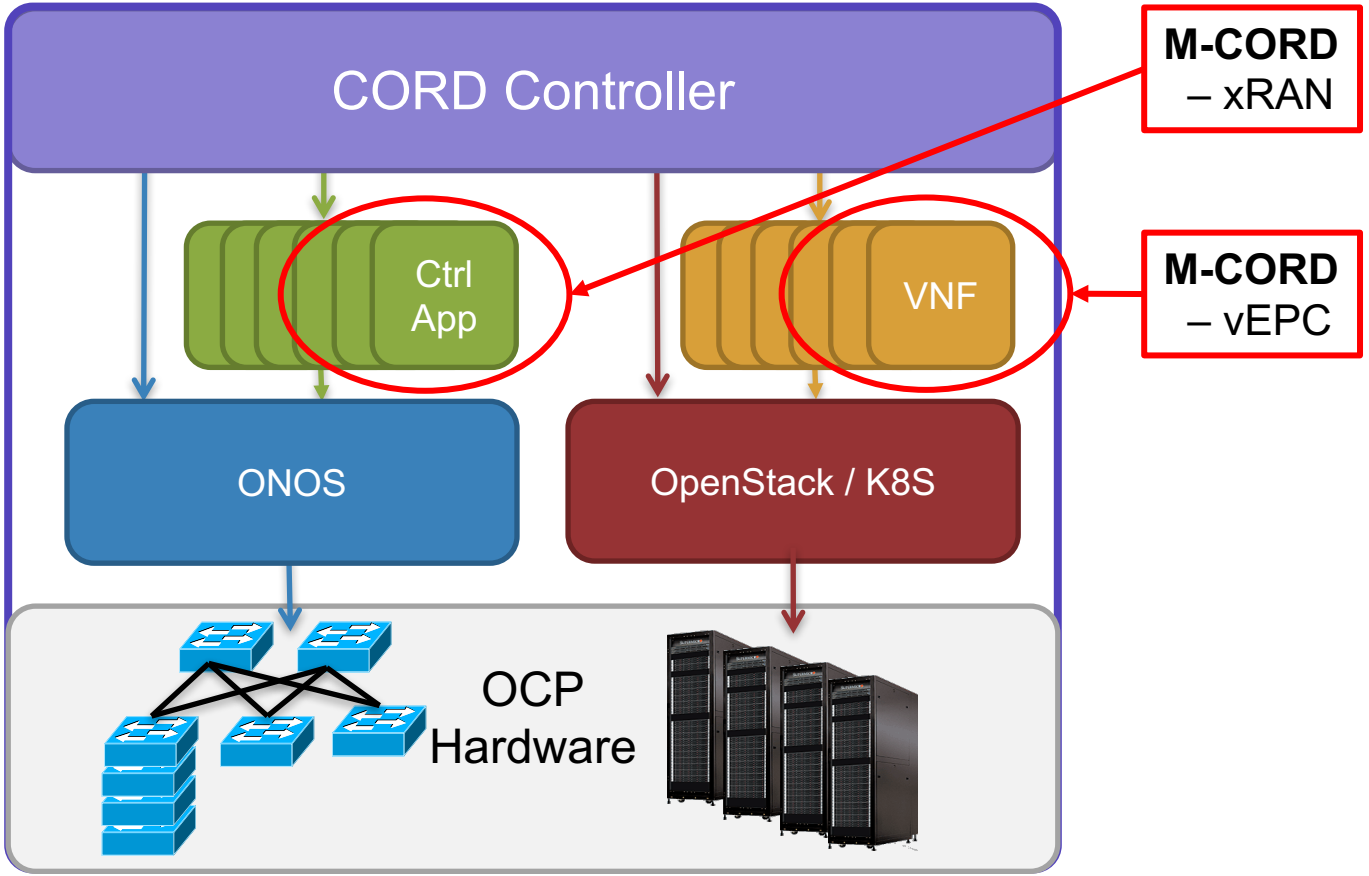
CORD Architecture



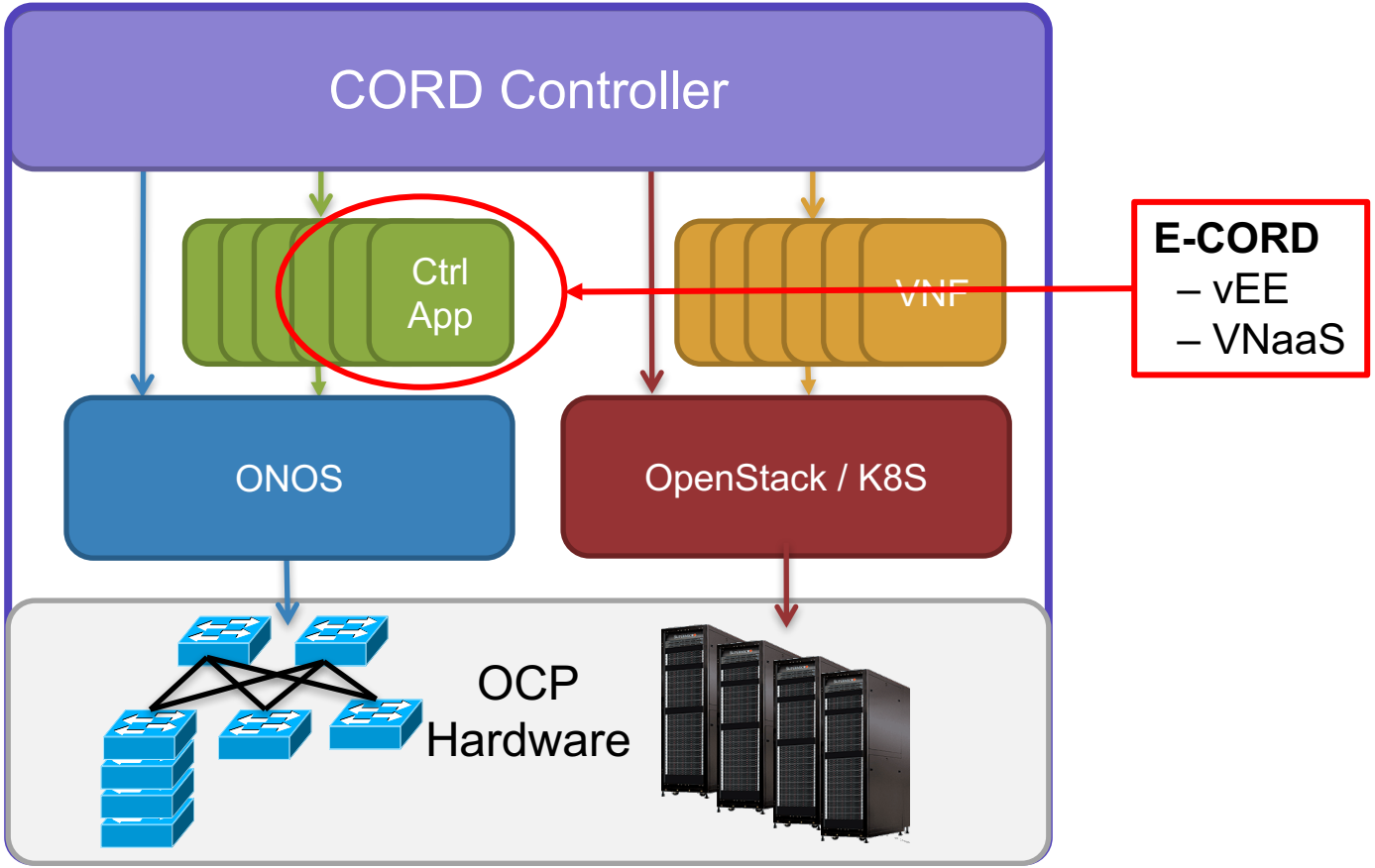
CORD Architecture



CORD Architecture

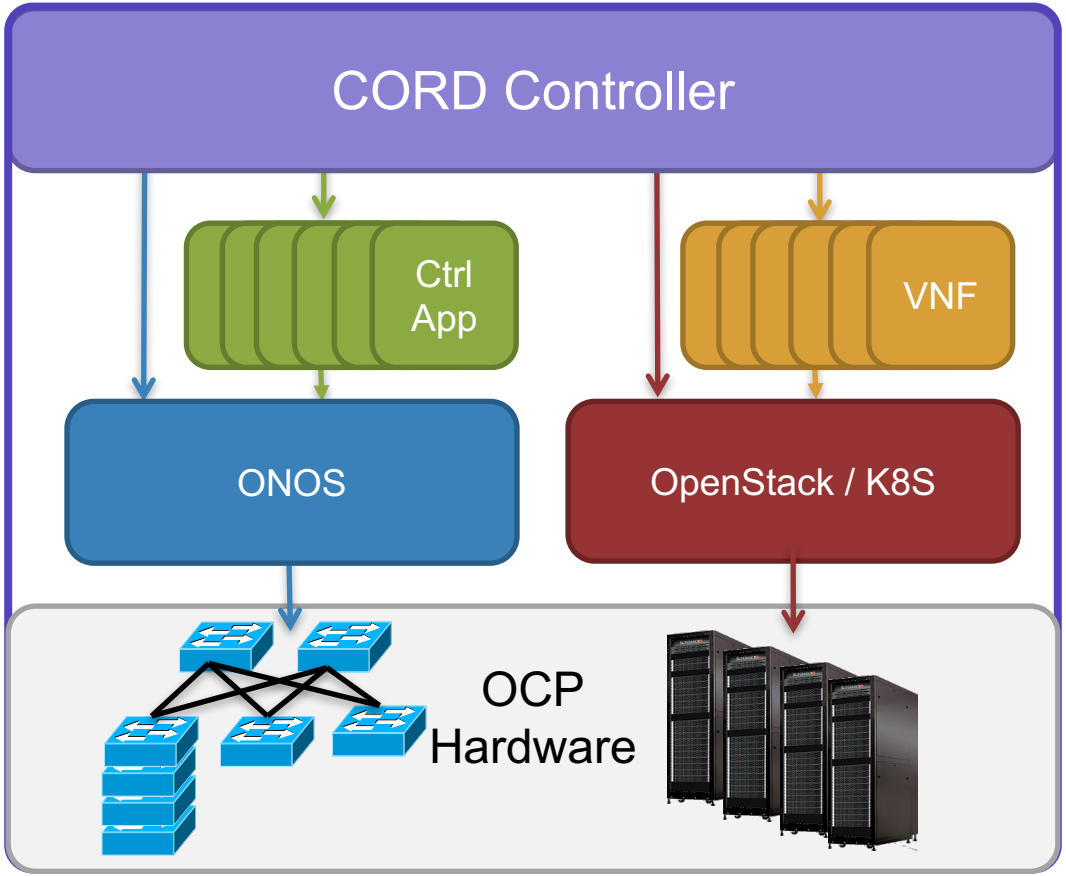


CORD Architecture



CORD Architecture

(Req 2,5)

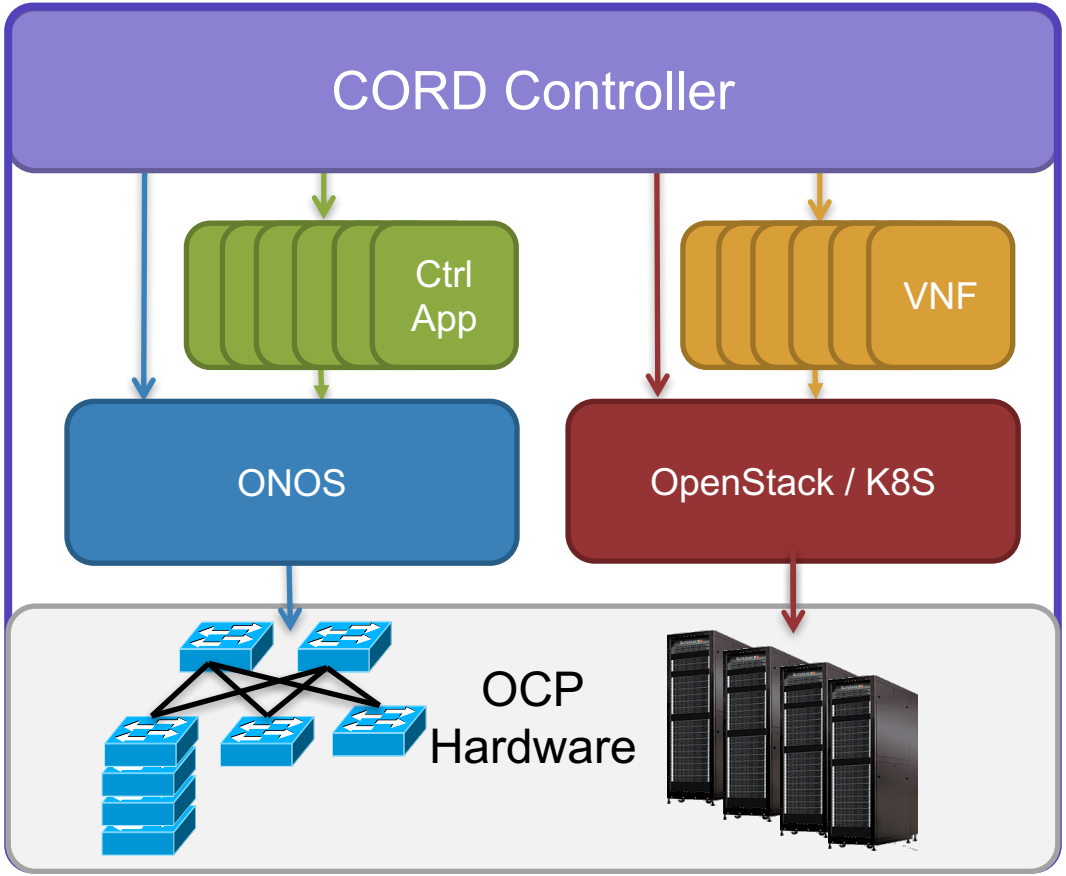


Virtualization Agnostic
- VMs
- Containers
- Micro-Services
- …

Instruction Set Agnostic
- Server-based (VNF)
- Switch-based (SDN)
- …

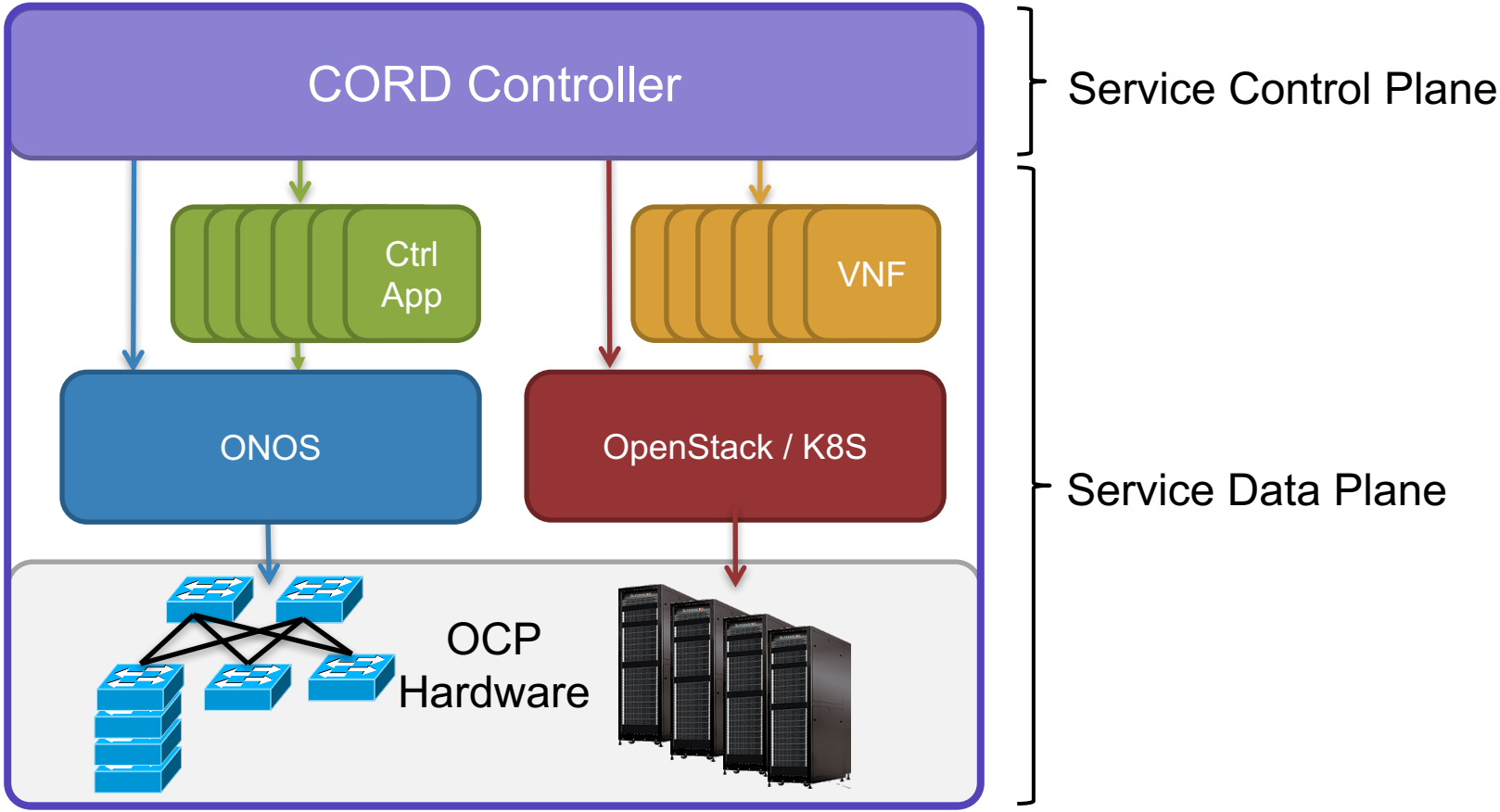
CORD Architecture

(Req 3,4)



- Provisions Services
- Mediates Trust
- Enforces Policies
- Assembles Data Paths

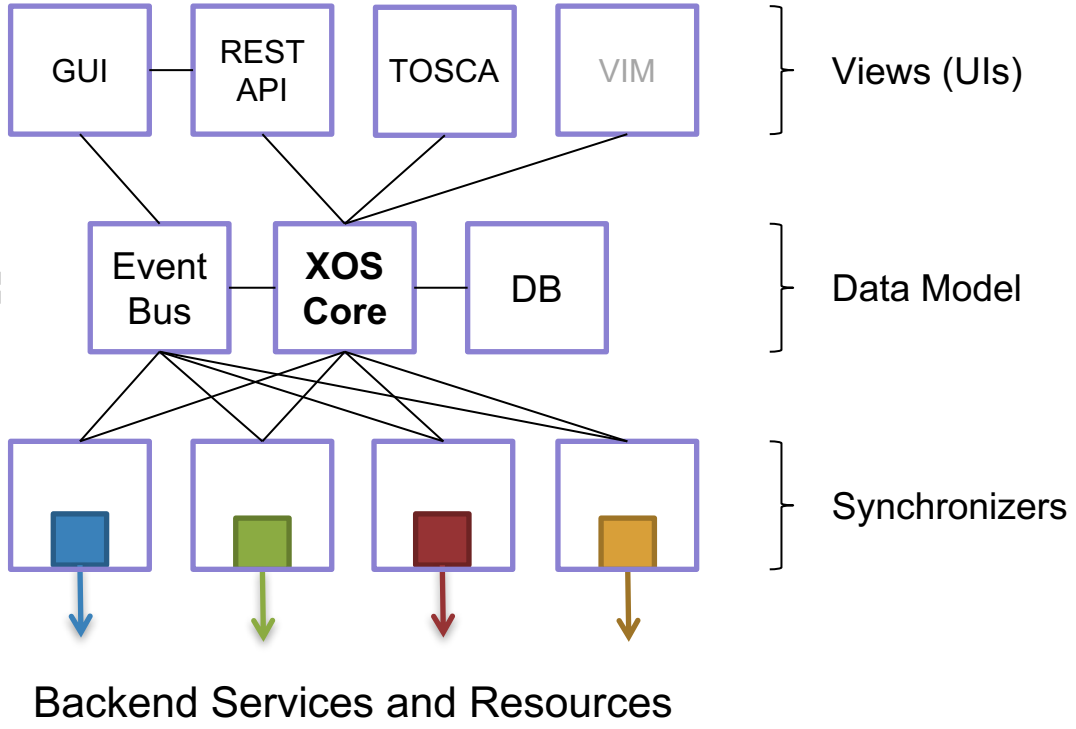
CORD Architecture



XOS Constructed from Micro-Services

CORD Controller (XOS)

=



XOS Generative Toolchain

Models Policies

XOS Core =

Language to Define & Operate on Models

– *xproto*

models: protobufs + extensions

policies: logic formulae applied to models

Generative Tool Chain

– *xosgen*

Auto-generated code to…

implement interfaces, execute synchronizers, enforce security

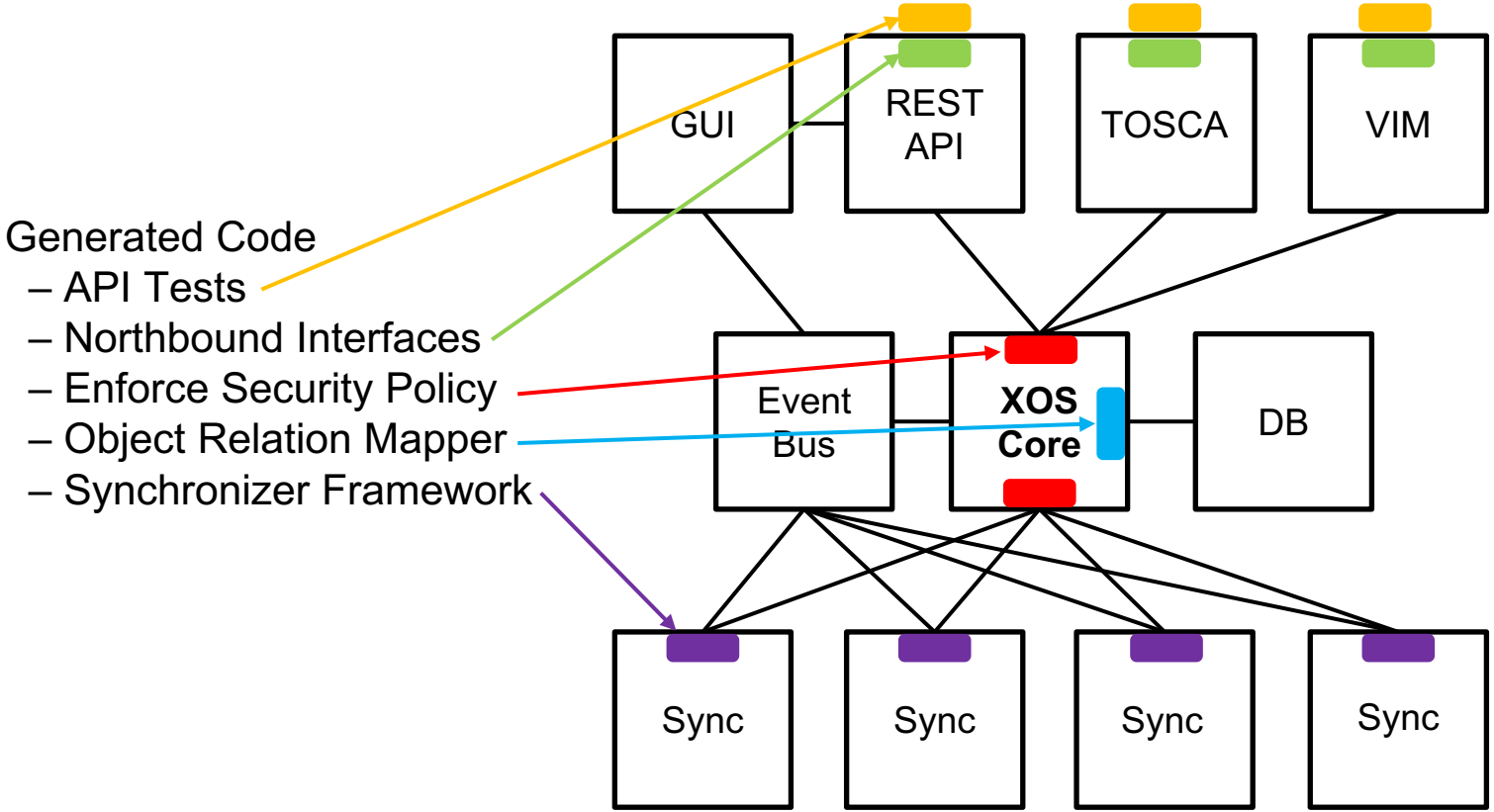
test end-to-end integration, validate consistency, apply policy

Example Model and Policy

```
policy grant_policy < ctx.user.is_admin
      | exists Privilege:Privilege.object_type = obj.object_type
      & Privilege.object_id = obj.object_id
      & Privilege.accessor_type = "User"
      & Privilege.accessor_id = ctx.user.id
      & Privilege.permission = "role:admin" >

message Privilege::grant_policy (XOSBase)
      { required int32 accessor_id = 1 [null = False];
        required string accessor_type = 2 [null = False, max_length=1024];
        required int32 controller_id = 3 [null = True];
        required int32 object_id = 4 [null = False];
        required string object_type = 5 [null = False, max_length=1024];
        required string permission = 6 [null = False, default = "all", max_length=1024];
        required string granted = 7 [content_type = "date", auto_now_add = True, max_length=1024];
        required string expires = 8 [content_type = "date", null = True, max_length=1024]; }
```

XOS Generative Toolchain



Synchronizer Framework

XOS auto-generates code for…

- Dependency management
- Error recovery
- Work partitioning
- Parallelization
- Logging

Service developer writes…

- A *Sync_Step( )* that is invoked when Service model changes
- An *Ansible Template* that specifies a VNF-specific playbook

Models and Frameworks

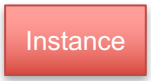
Previous five slides have been about mechanism

- XOS is a framework for specifying and evaluating models

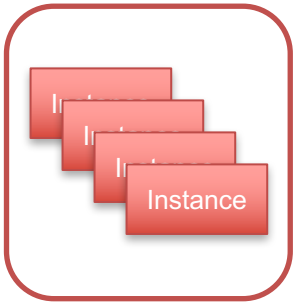
CORD also includes a set of core models

- Familiar building blocks – *Instances*, *Networks*
- Virtualization-agnostic infrastructure – *Slice*
- ISA-agnostic Service graph – *Service, ServiceDependency*
- Subscribers – *CordSubscriberRoot*
- Per-user service chains – *ServiceInstance*, *ServiceInstanceLink*

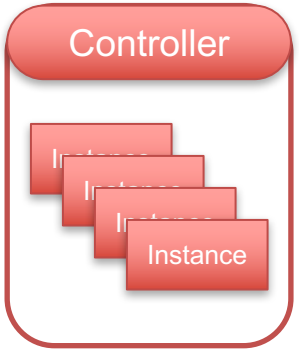
Core Models



Instance =
(VM | Container |
Container-in-VM)



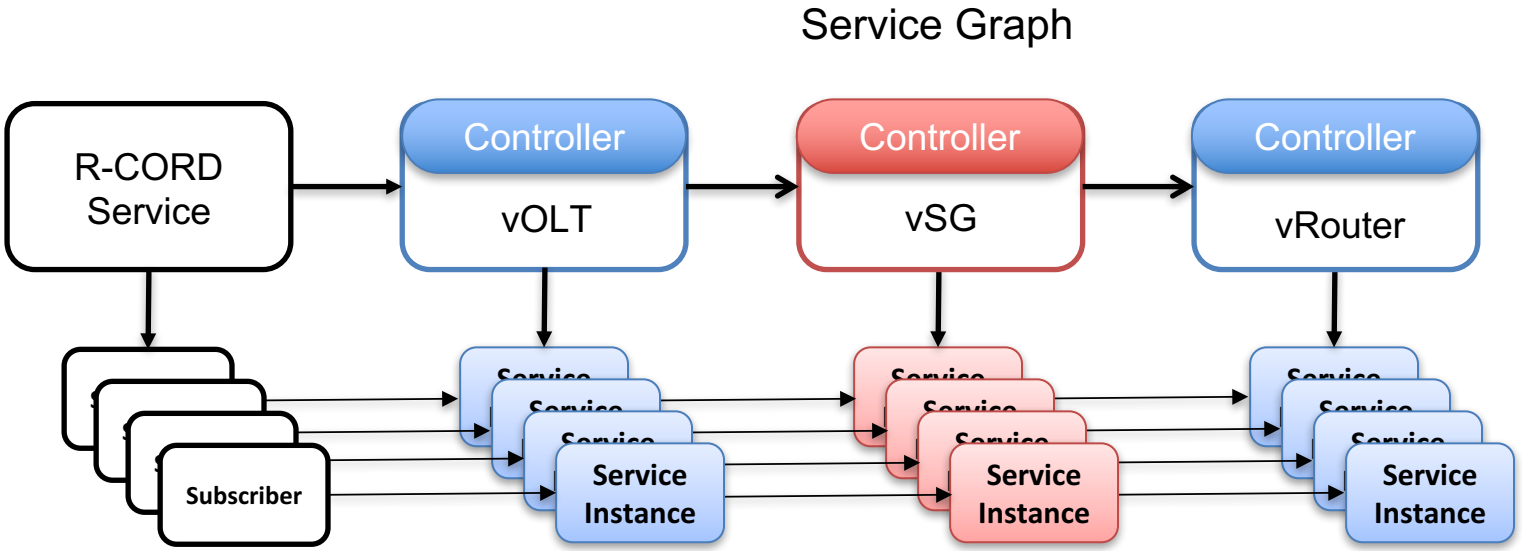
Slice =
(Instances[ ] + Networks[ ])



Service =
(Controller + Slices[ ])

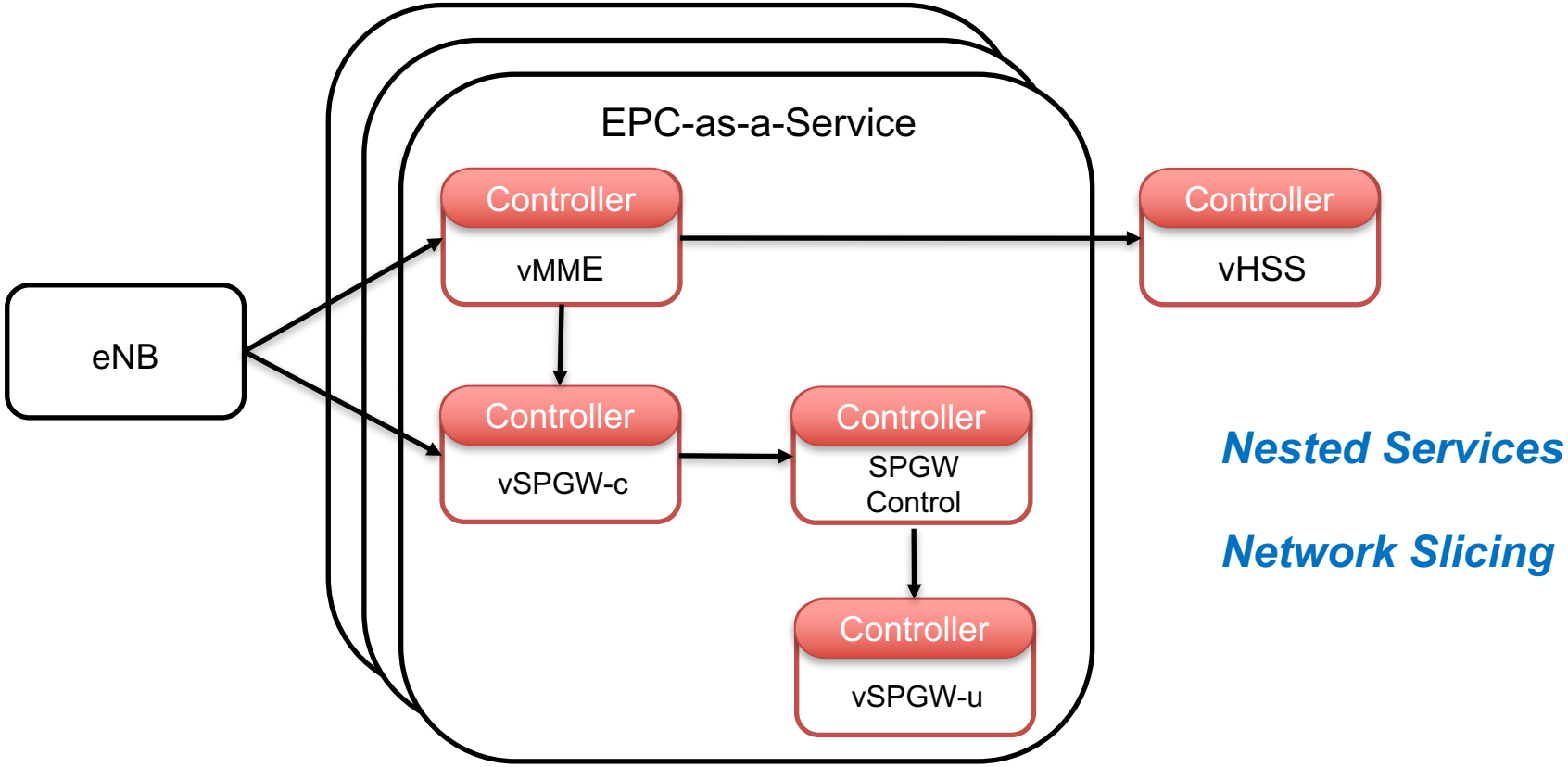
Service Graph =
(Services[ ] + Dependencies[ ])

Core Models



Service Chain =
(ServiceInstances[ ] + ServiceInstanceLinks[ ])

Core Models



Nested Services

Network Slicing

Build System

Serves multiple masters…

Developers that want tight development loops for the components they are working on.

Integrators that want flexible configurability to combine and test targeted solutions.

Operators that want determinist builds that include only certified components and supports zero-touch.

Multi-Stage Build System

Configure

```
cord_profile: rcord, mcord, ecord,…
cord_scenario: local, mock, single, cord,…
```

Build

- Fetch – onto development machine
- Build – containers (if necessary)
- Publish – to repository on head node

Multi-Stage Build System

Deploy

- Run management containers (XOS, ONOS, OpenStack) on head node
- Docker Compose today / plans to have Kubernetes help manage

Boot

- Bring up compute nodes and switches
- Leverage MAAS and PXE

Build Tooling

A set of *YAML* files represents all configuration state

→ All builds start at `build/podconfig/`*profile-scenario*`.yml`

A set of *Docker images* define the canonical representation of the system

→ Makes it easier to identify “golden” components

→ Makes it easier to iterate on a specific component during development

A set of *Ansible roles* separates configuring/installing/deploying containers

→ Makes it easy to adapt CORD to new scenarios

A sequence of *Make targets* represent build milestones

→ Makes it easy to roll back and incrementally re-build

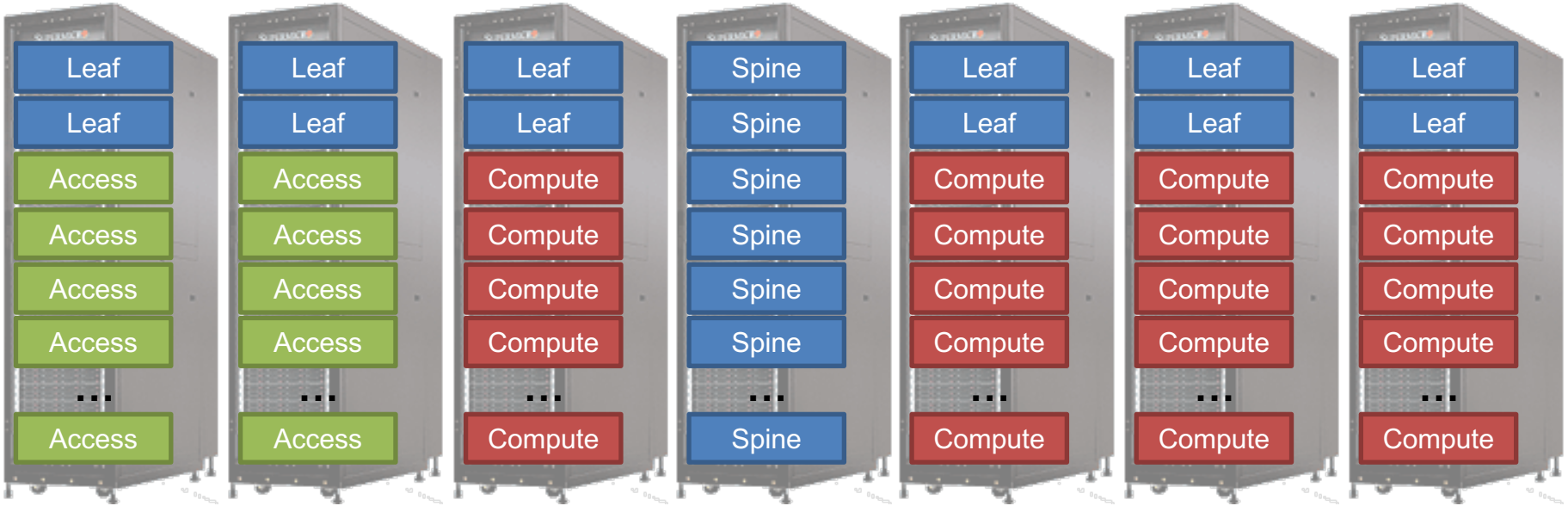
CORD as Configurable Platform

Having built CORD from commodity hardware and disaggregated software services, the operator has wide latitude in how to reassemble the building blocks

Two Adjustable Levers

- Sizing – Number and mix of hardware components
- Configuration – Set of on-boarded software services

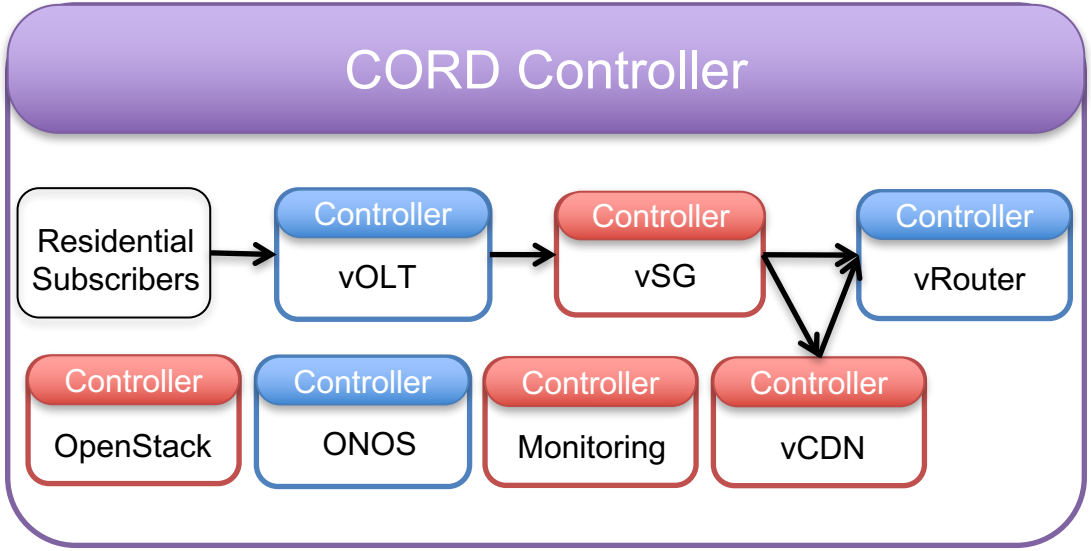
Scale Up – Hardware



Full POD

(Up to 16 Racks with 32x40GE switches)

Scale Up – Software



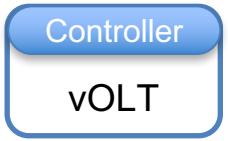
Provision Services On-Demand

Scale Down – Lite-and-Right CORD

Single/Partial Rack (No Spine Switches)

Minimal Compute (All services run in containers)

Merchant Silicon Access Blades (e.g., OLT)



If you also “scale down” the software so the Service Graph includes just vOLT, the resulting configuration = *“White-Box OLT”*

Multi-Access Edge Cloud

Access Devices = OLT + BBU + ETH

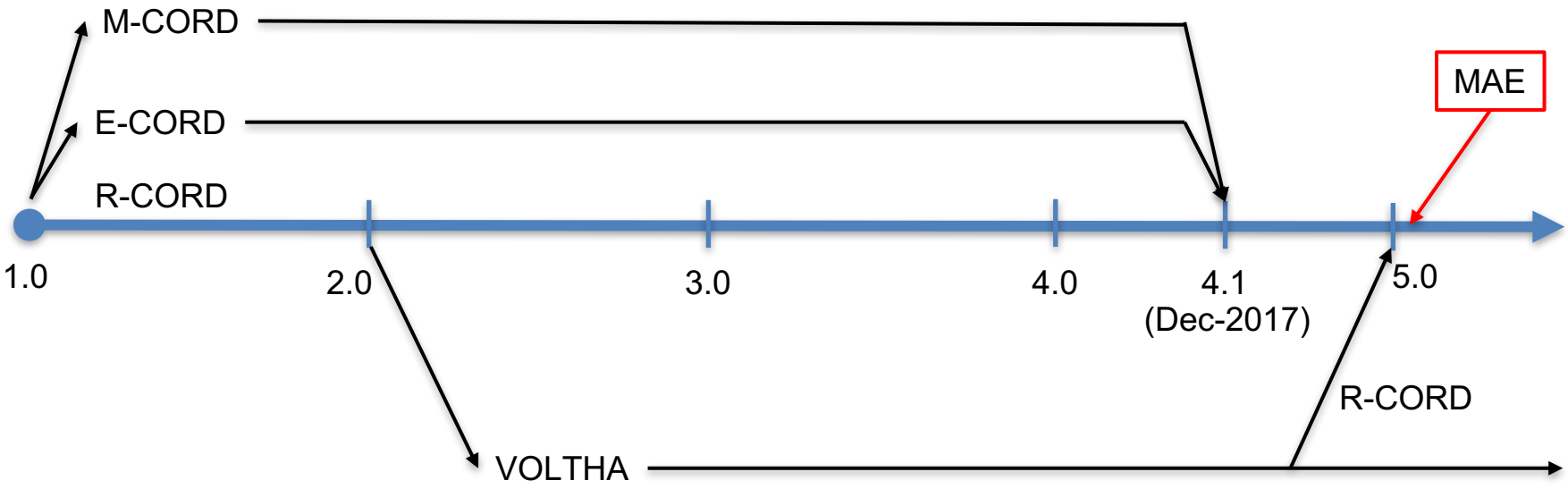
CORD Controller

Controller
vOLT
… R-CORD Sub-Graph…

Controller
vBBU
… M-CORD Sub-Graph…

Controller
vEE
… E-CORD Sub-Graph…

Current Status



Service Portfolio (4.1)

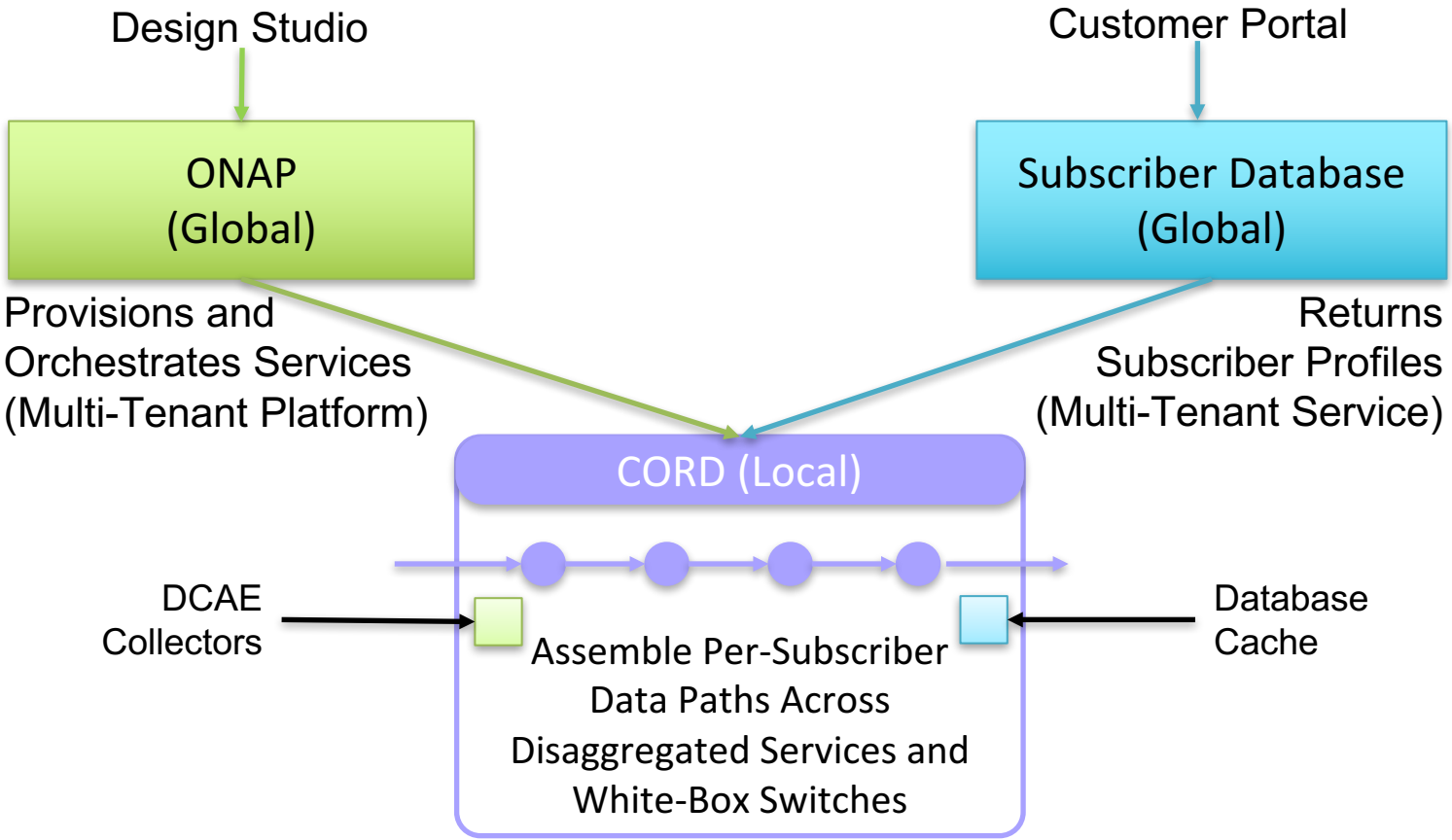
vSG – Virtual Subscriber Gateway
vOLT – Virtual OLT
vRouter – Virtual Router
vEG – Virtual Enterprise Gateway
vEE – Virtual Enterprise Ethernet
vHSS – Virtual Home Subscriber Server
vMME – Virtual Mobility Management Entity
vEPC – Virtual Evolved Packet Core
vTR – Virtual Truck Roll
HyperCache – Akamai CDN
SGW – Virtual Serving Gateway (User)
vSGWc – Virtual Serving Gateway (Control)
vPGW – Virtual Packet Gateway (User)
vPGWc – Virtual Packet Gateway (Control)
vBBU – Virtual Broadband Base Unit
xRAN – Virtual Radio Access Network

ONOS – Network OS
OpenStack – Intrastructure-as-a-Service
Swarm – Container Management Service
Fabric – Fabric Management Service
VTN – Virtual Tenant Network
A-CORD – Monitoring-as-a-Service
LBaaS – LoadBalancer-as-a-Service
VNaaS – VirtualNetwork-as-a-Service

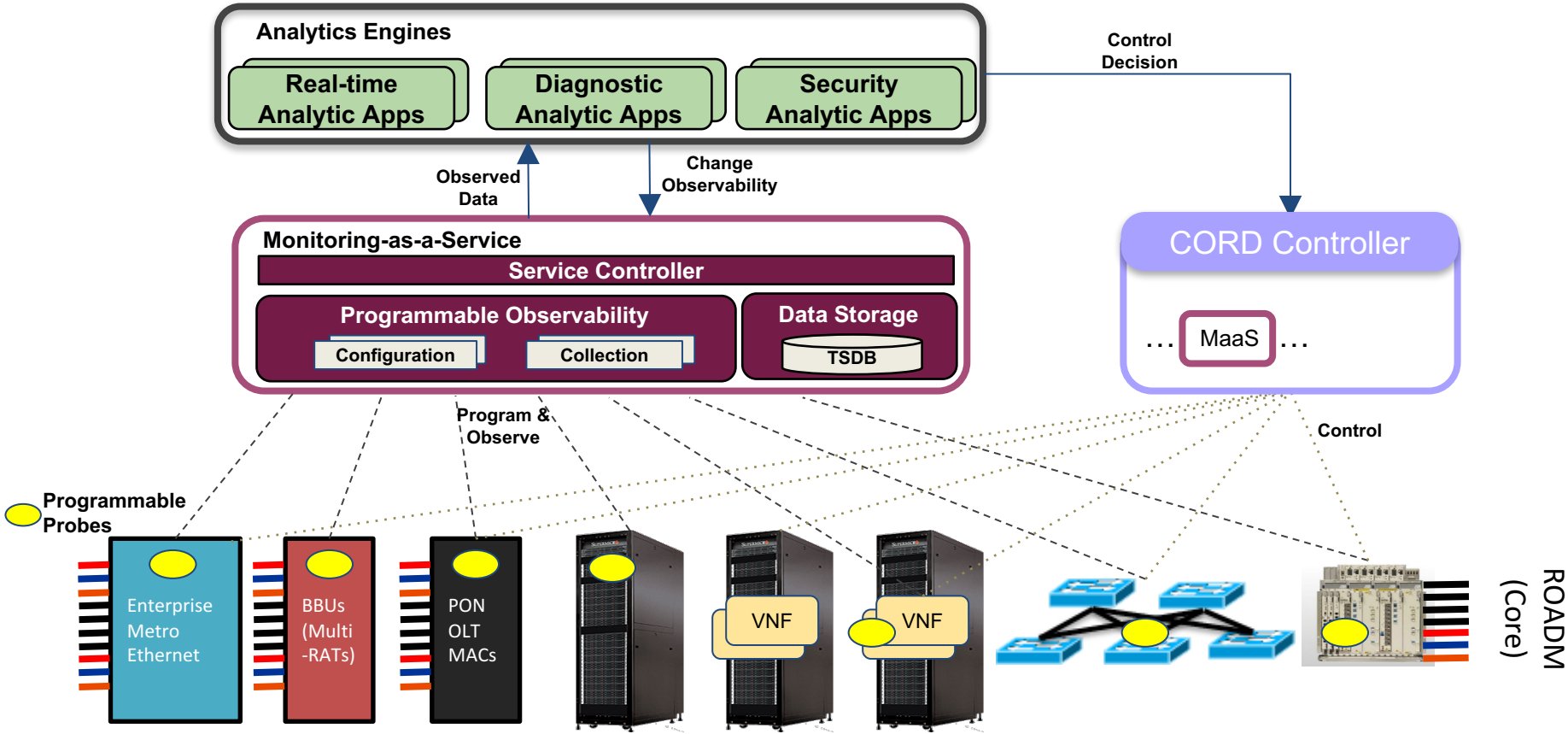
Helper Services

AAA – Access Control
AddressManager – Allocate IP Addresses
IGMP – Multicast Signalling
MCAST – Multicast
SADIS – xxx

CORD and ONAP



A-CORD – Monitoring-as-a-Service



Learn About CORD

Hands-on CORD learning and skill development

Developed and Hosted by Criterion Networks

In collaboration with ONF

Online subscription-based pricing model

Now in Beta (by invitation)

		Duration	
Total Lab		12 Hrs	
Lab I	CORD Network Management	4 Hrs	$125
Lab II	CORD Virtual Networks	4 Hrs	$125
Lab III	CORD Services Framework	4 Hrs	$125

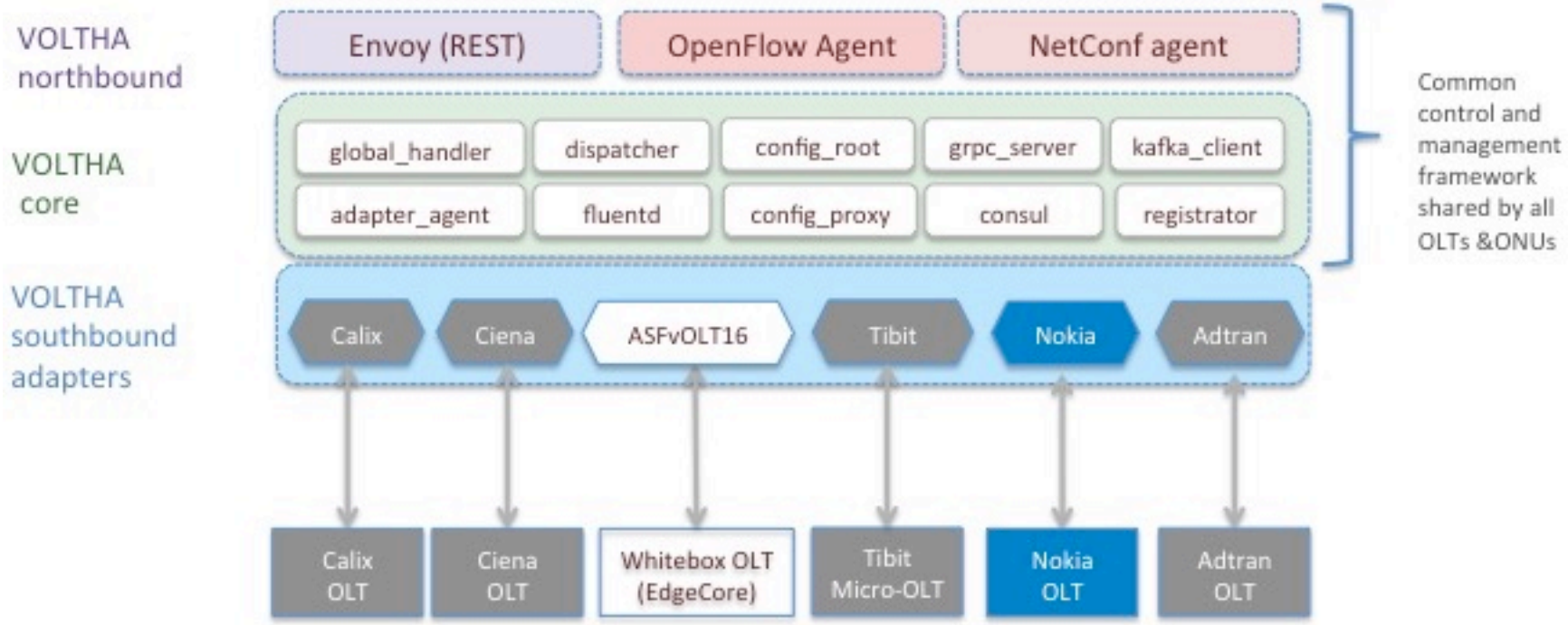
More Information

Software – https://guide.opencord.org

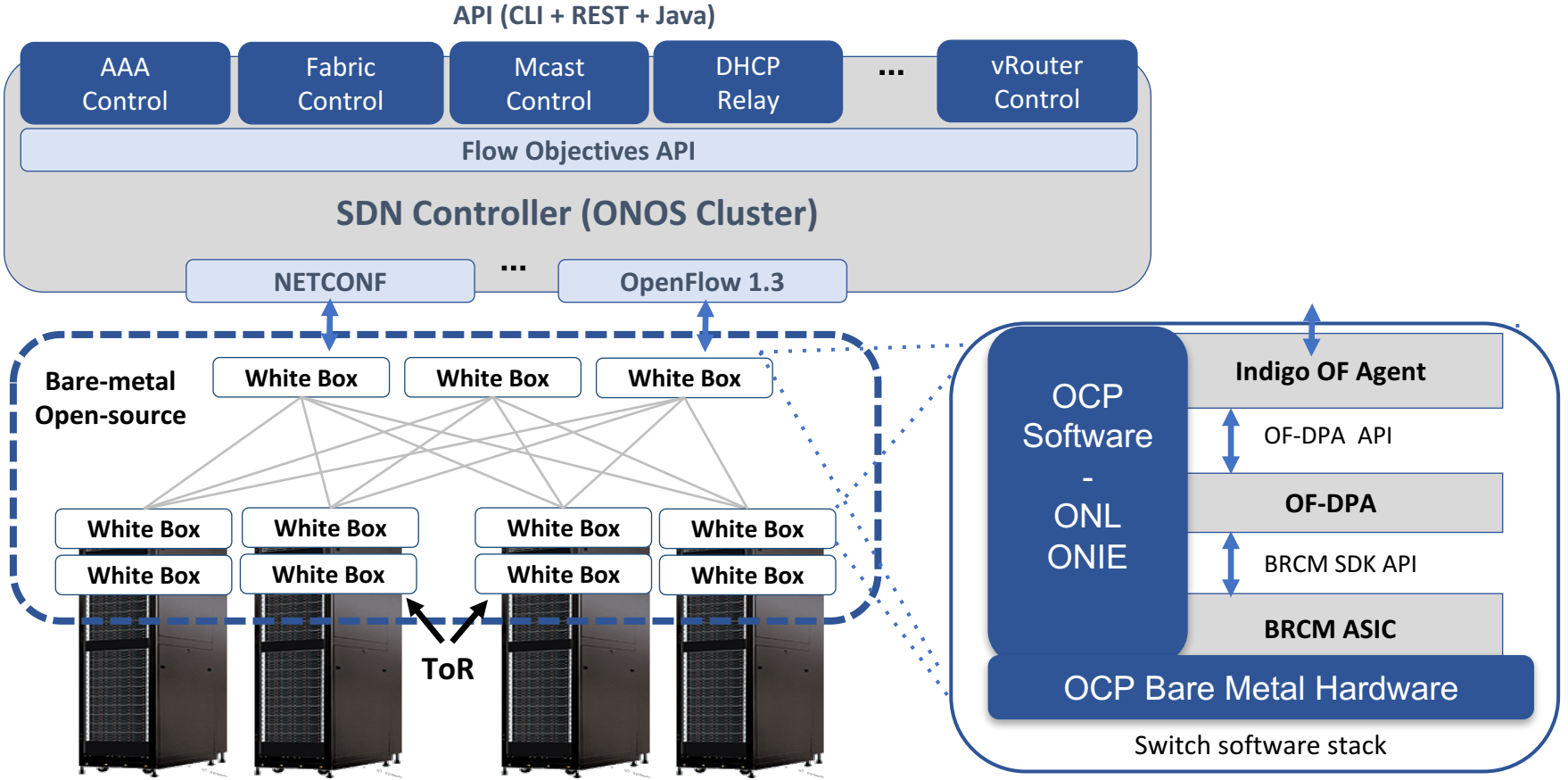
Community – https://wiki.opencord.org

VOLTHA

VOLTHA hides PON-level details (T-CONT, GEM ports, OMCI etc.) from the SDN controller, and abstracts each PON as a pseudo-Ethernet switch easily programmed by the SDN controller



Trellis


API (CLI + REST + Java)
AAA Control
Fabric Control
Mcast Control
DHCP Relay
...
vRouter Control
Flow Objectives API
SDN Controller (ONOS Cluster)
NETCONF
...
OpenFlow 1.3
Bare-metal Open-source
White Box
White Box
White Box
White Box
White Box
White Box
White Box
White Box
White Box
White Box
ToR
OCP Software - ONL ONIE
Indigo OF Agent
OF-DPA API
OF-DPA
BRCM SDK API
BRCM ASIC
OCP Bare Metal Hardware
Switch software stack