Lecture: Week 4 - 1

Introduction to Open Network Operating System (ONOS)

James Won-Ki Hong, Jian Li, Seyeon Jeong

Dept. of Computer Science & Engineering
POSTECH

http://dpnm.postech.ac.kr/~jwkhong
jwkhong@postech.ac.kr

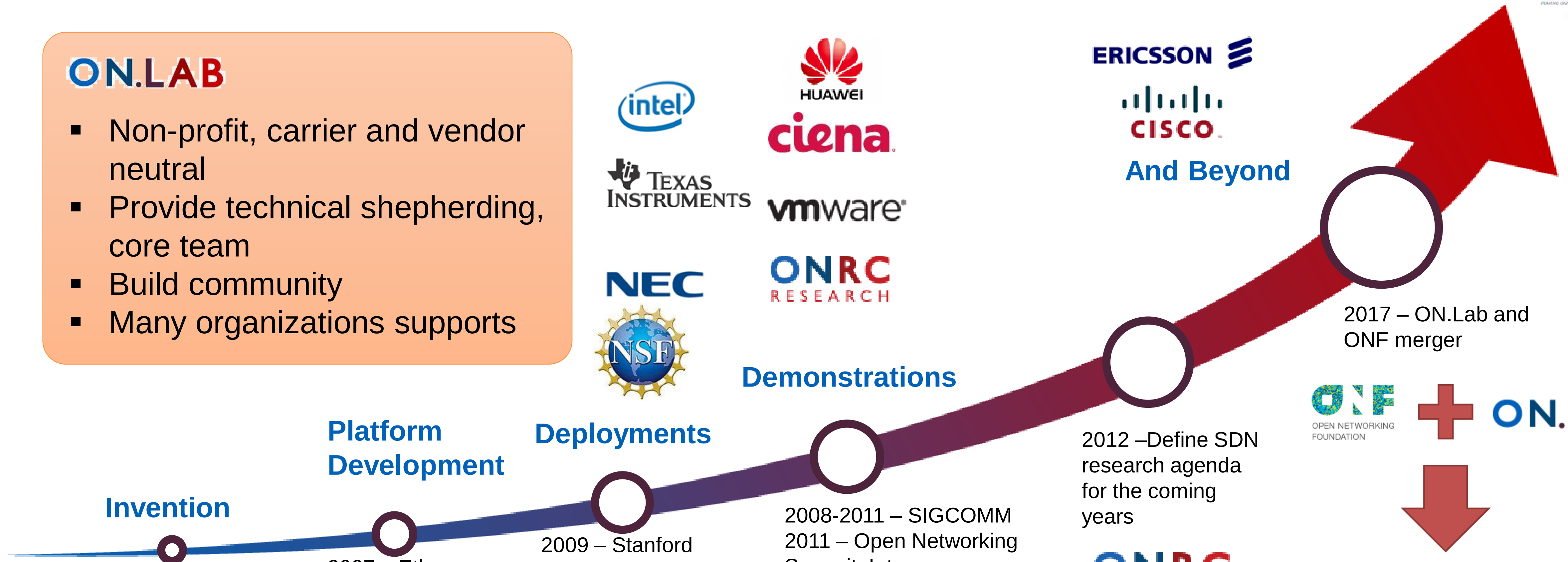
Outline

- W4-1: Introduction to ONOS
- W4-2: ONOS Distributed Core
- W4-3: ONOS Northbound
- W4-4: ONOS Southbound & Application

SDN Evolution and ON.LAB

ON.LAB

- Non-profit, carrier and vendor neutral
- Provide technical shepherding, core team
- Build community
- Many organizations supports



Invention

2007 – Creation of SDN Concept

Platform Development

2007 – Ethane
2008 – OpenFlow
2009 – FlowVisor, Mininet, NOX
2010 – Beacon

Deployments

2009 – Stanford
2010 – GENI started and grew to 20 universities
2013 – 20 more camp uses to be added

Demonstrations

2008-2011 – SIGCOMM
2011 – Open Networking Summit, Interop

2012 –Define SDN research agenda for the coming years

And Beyond

2017 – ON.Lab and ONF merger

Why are SPs Interested in SDN and ONOS?

Reduce CAPEX and OPEX

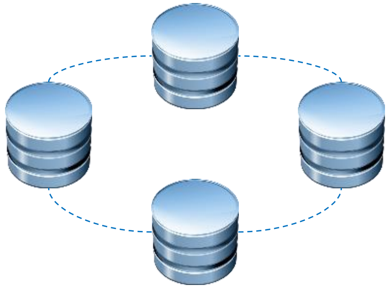
Bring cloud-style agility, flexibility, scalability to their networks

Roll out service rapidly

Reduce operational complexity, increase visibility

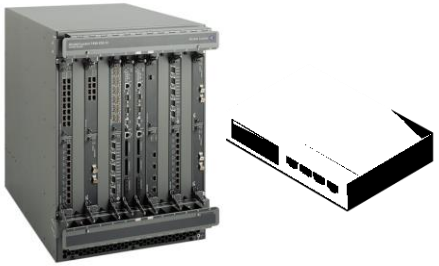
But Service Provider networks place stringent requirements on SDN control plane

Handle tens of millions of fixed and hundreds of millions wireless end points



Provide five nines availability, high performance, low latency

Need ease of use, service creation and delivery

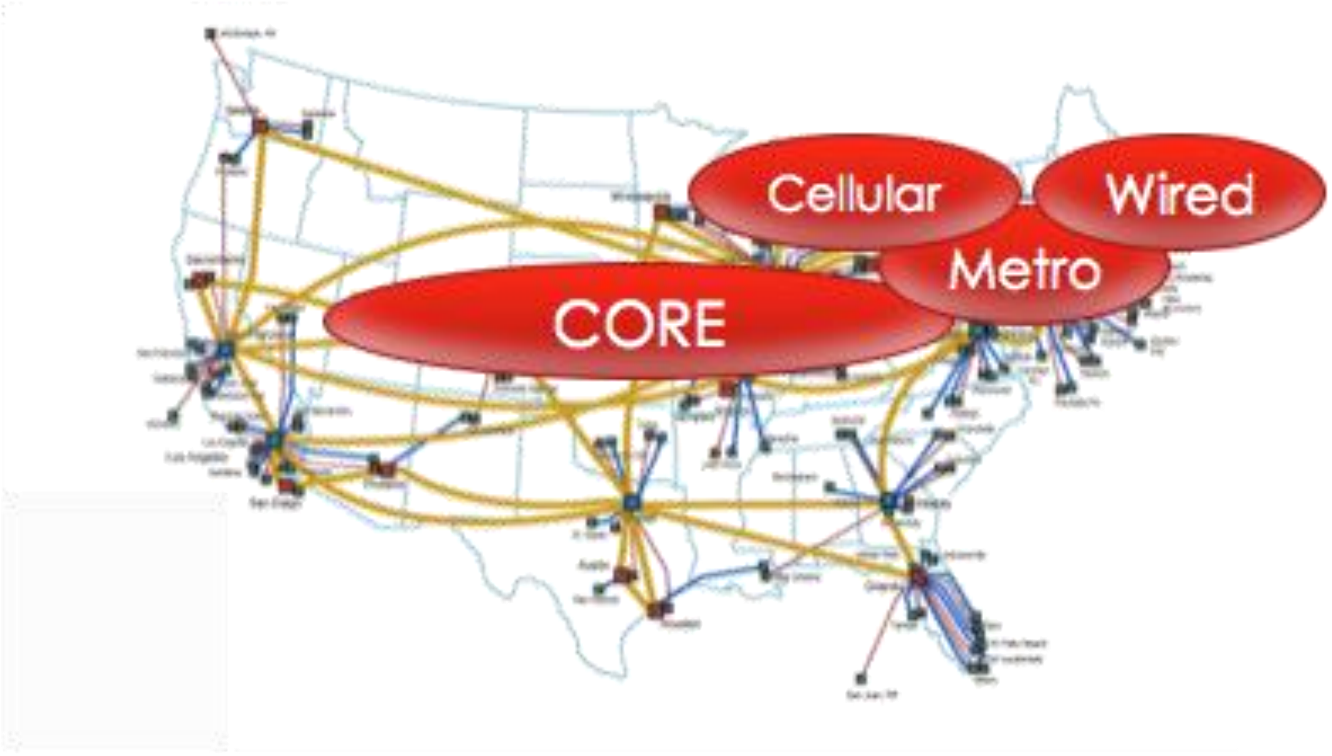


Allow seamless migration of existing N/W while capitalizing on white boxes

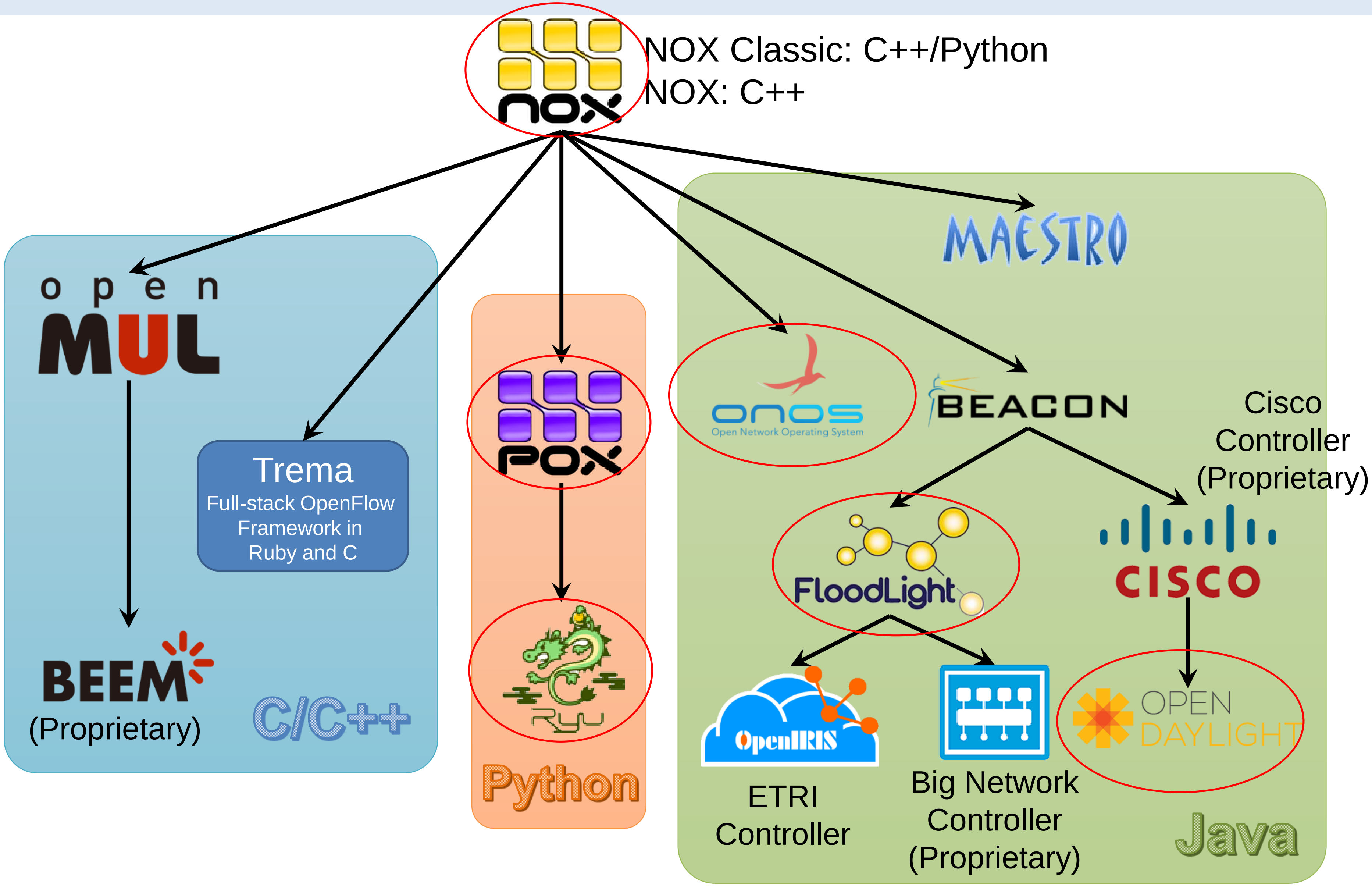
ONOS is a SDN network operating system (control plane platform) designed for these stringent Service Provider requirements

Service Provider Networks

- **WAN core backbone**
  - Multi-protocol Label Switching (MPLS) with Traffic Engineering (TE)
  - 200-500 routers, 5-10K ports
- **Metro Networks**
  - Metro cores for access networks
  - 10-50K routers, 2-3M ports
- **Cellular Access Networks**
  - LTE for a metro area
  - 20-100K devices, 100K-100M ports
- **Wired Access / Aggregation**
  - Access network for homes
  - 10-50K devices, 100K-1M ports



Pedigree Chart of OpenFlow Controllers



Introduction

❖ ONOS: Open Network Operating System

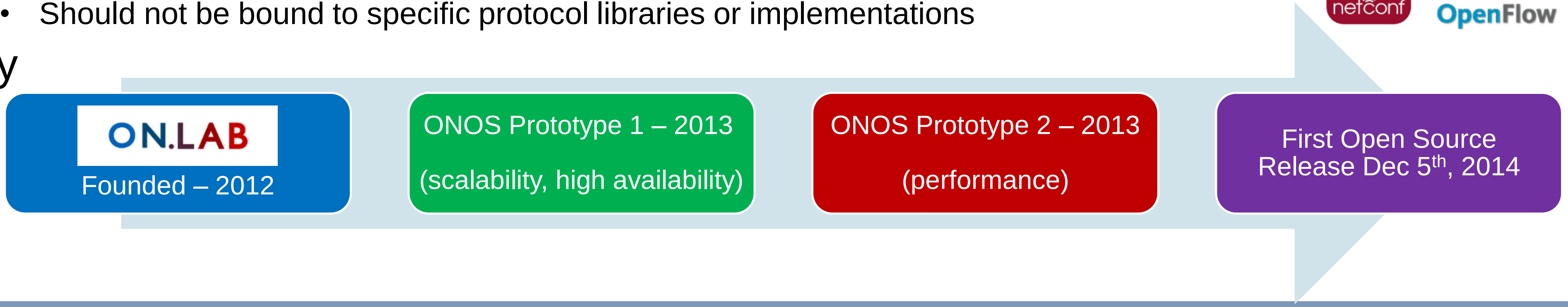
- SDN OS for service provider networks
- Design goals
  - Code modularity
    - Possible to introduce new functionalities as self-contained units
  - Configurability
    - Possible to load and unload various features in runtime
  - Separation of Concern
    - There should be clear boundaries between subsystems to facilitate
    - **Protocol-aware network-facing modules** → interact with network
    - **Protocol-agnostic system core** → tracks and serves info on network state
    - **Application** → consumes and acts on the information provided by core
  - Protocol agnosticism
    - Should not be bound to specific protocol libraries or implementations
- History

Open Network Operating System

Application

Core

Network Facing Modules



ON.LAB Founded – 2012

ONOS Prototype 1 – 2013 (scalability, high availability)

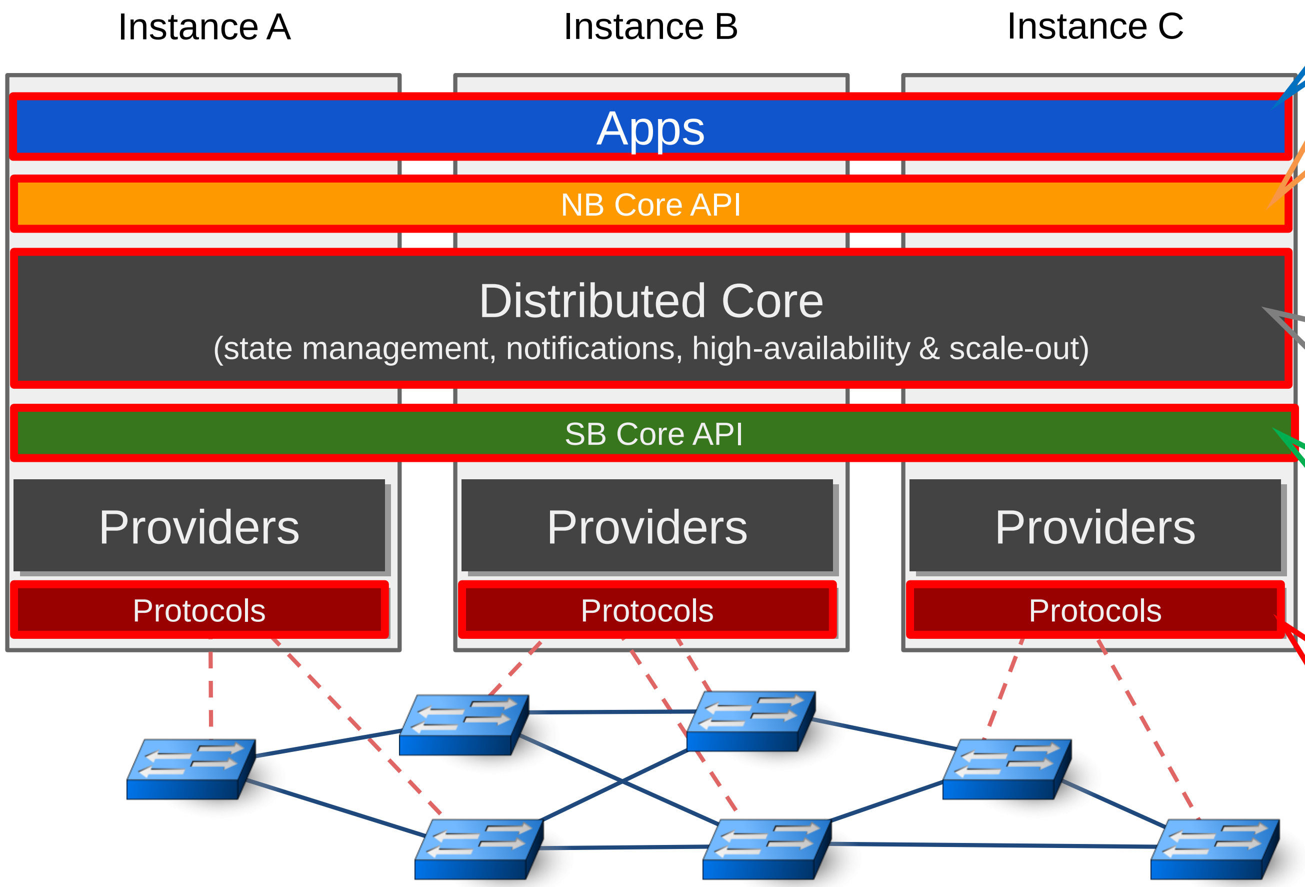
ONOS Prototype 2 – 2013 (performance)

First Open Source Release Dec 5th, 2014

ONOS Distributed Architecture (1/2)

❖ Distributed Architecture

- Each instance is equipped with the same stack



Contains user applications E.g., reactive forwarding, proxy arp, segment routing, SDN-IP, etc.

Transfer network info to application layer Provide management interface for controlling lower layer component

Contains many core features Provide distributed clustering function for supporting HA and scalability

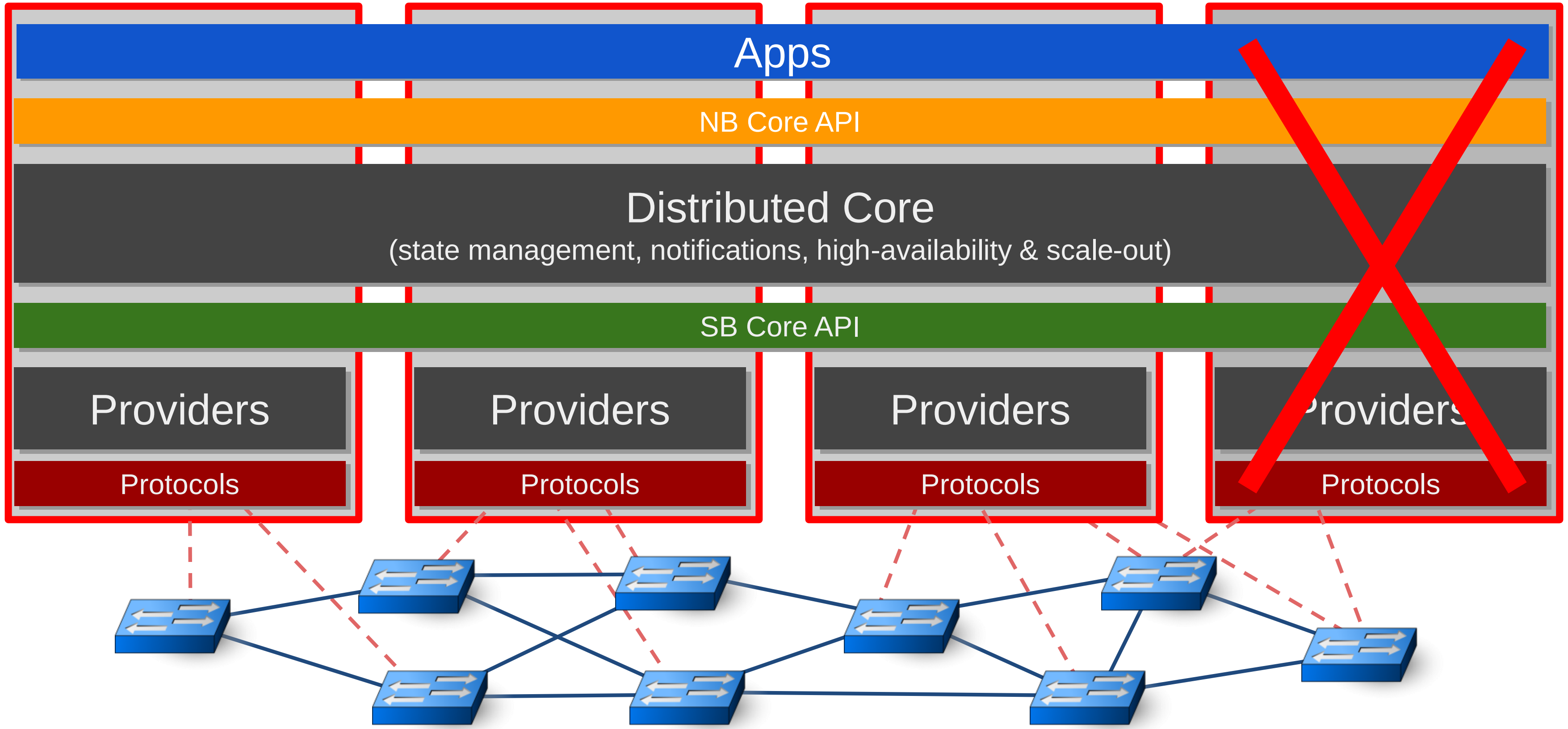
Provide a abstracted interface for controlling the network infrastructure

Real network protocol implementation for managing the network elements E.g., OpenFlow, NetConf

ONOS Distributed Architecture (2/2)

❖ Features

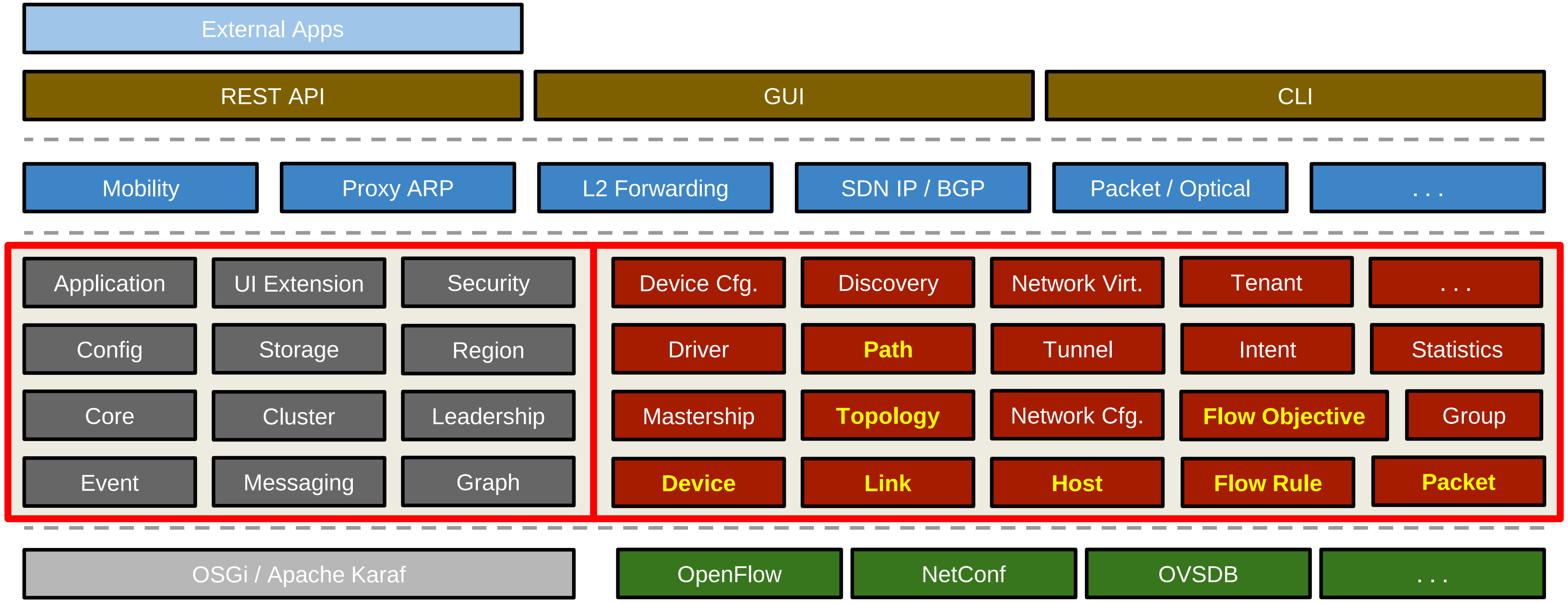
- High Availability (HA)
- Load Balancing (LB)



ONOS Core Subsystem

❖ Subsystem = Service

- An unit of functionality that is comprised of multiple components that create a vertical slice through the tiers as a software stack



External Apps

REST API | GUI | CLI

Mobility | Proxy ARP | L2 Forwarding | SDN IP / BGP | Packet / Optical | . . .

Application | UI Extension | Security | Device Cfg. | Discovery | Network Virt. | Tenant | . . .

Config | Storage | Region | Driver | **Path** | Tunnel | Intent | Statistics

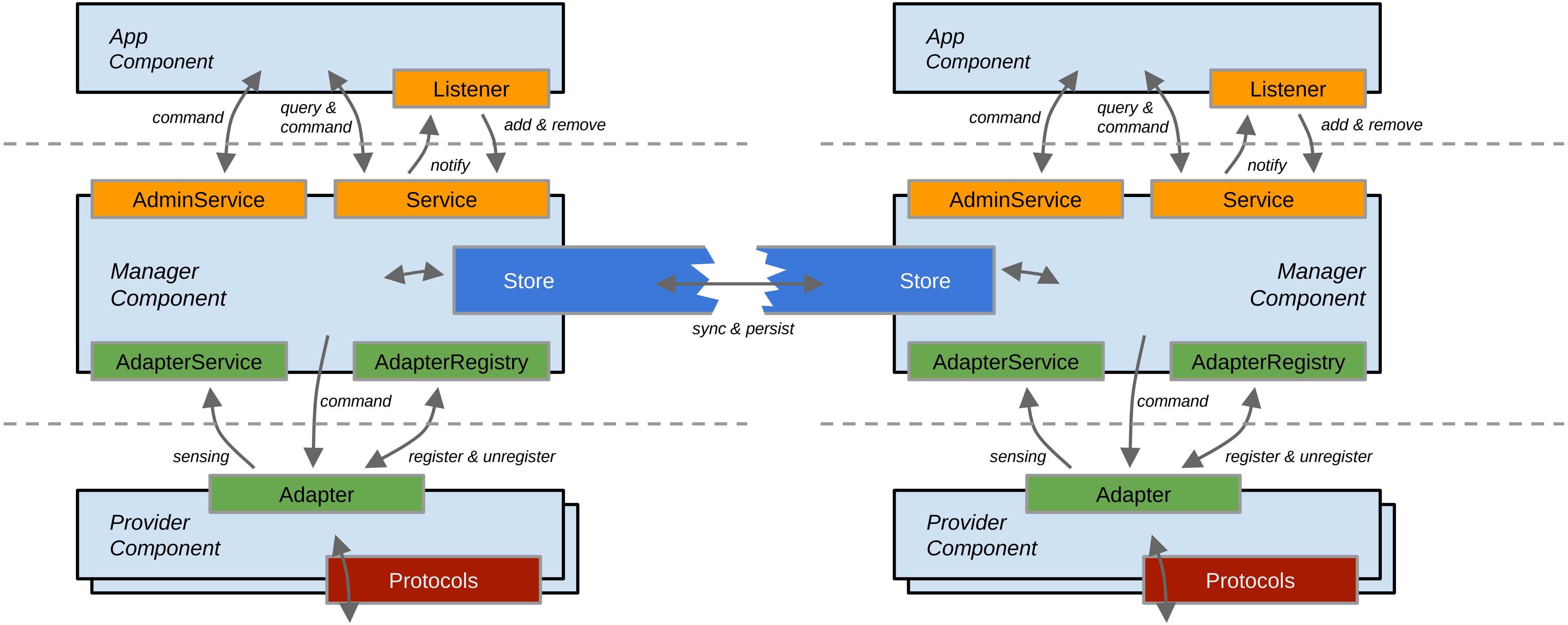
Core | Cluster | Leadership | Mastership | **Topology** | Network Cfg. | **Flow Objective** | Group

Event | Messaging | Graph | **Device** | **Link** | **Host** | **Flow Rule** | **Packet**

OSGi / Apache Karaf | OpenFlow | NetConf | OVSDB | . . .

⬜	OSGi Framework	🟦	Applications	🟩	South Bound Interface module
⬛	Non-networking core subsystem	🟫	Application Interfaces		
🟥	Networking core subsystem	🟦	External Applications		

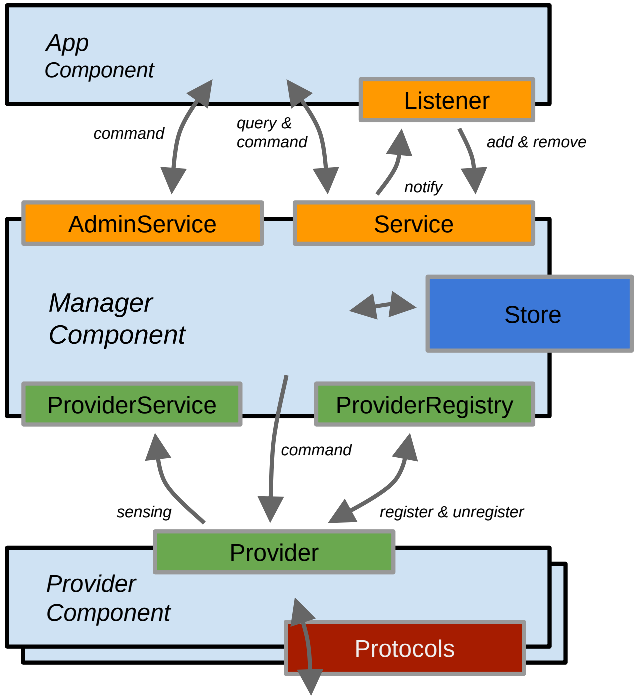
Subsystem Architecture (1/2)



Subsystem Architecture (2/2)

❖ Subsystem Structure

- Each of a subsystem’s components resides in one of three main tiers
- Provider
  - Interfaces with the network via protocol-specific libraries, with core via the *ProviderService* interface
- Manager
  - Resides in the core, receives information from Providers and serves it to applications and other services
  - Store
- Application
  - Provides wide range of functionality
  - Consumes and manipulates info. aggregated by the managers



App Component
Listener
command
query & command
add & remove
notify
AdminService
Service
Manager Component
Store
ProviderService
ProviderRegistry
command
sensing
register & unregister
Provider
Provider Component
Protocols