SDN & OpenDaylight

Aula Teórica nº1

2020/2021

Avaliação

- **Avaliação continua:**
  - Componente teórica (CT): 1 Prova Escrita (PE) de escolha múltipla sem mínimos (modalidade online)
  - Componente prática (CP): 2 Trabalhos Laboratoriais (TL1 e TL2), sem mínimos e com defesa individual única (modalidade online)
  - Classificação final: PE*0.3 + (TL1*0.35+TL2*0.35) * [0..1] defesa
  - Para a época de exame normal e de recurso são guardadas as notas CT e CP

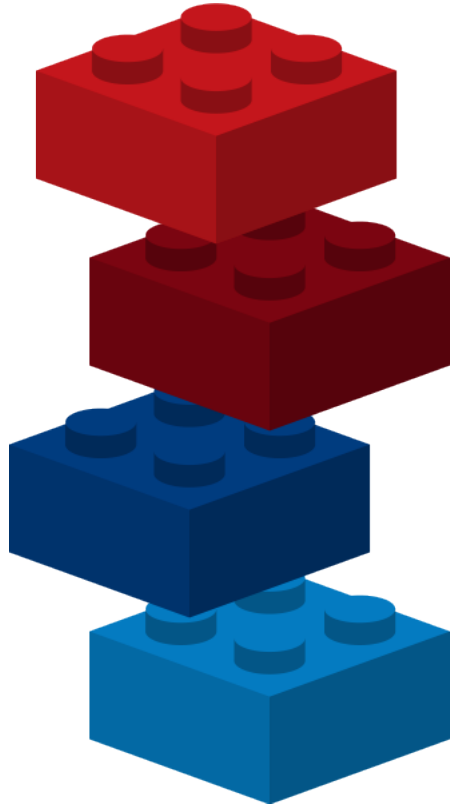
- **Restantes Épocas:**
  - Componente teórica (CT): Prova Escrita (PE) de escolha múltipla e desenvolvimento sem mínimos (modalidade online)
  - Componente prática (CP): Teste Prático (TP) com mínimos de 9.5 valores (modalidade presencial)
  - Classificação final: PE*0.3 + TP*0.7
- A melhoria apenas pode ser feita para ambas as componentes ao mesmo tempo, não se guardando classificações de outras épocas.
- Caso não seja possível realizar a componente prática (CP) das épocas de exame na modalidade presencial, estas serão realizadas na modalidade online.

The Software-Defined Networking (SDN) movement

The modern SDN movement grew out of a simple question: *why shouldn't networking devices be programmable just as other computing platforms are?*

What is SDN?

The physical separation of the network **control plane** from the **forwarding plane**, and where a control plane controls several devices.



What is SDN?

An emerging architecture that is **dynamic**, **manageable**, **cost-effective**, and **adaptable**, making it ideal for the high-bandwidth, dynamic nature of today's applications.

This architecture **decouples** the **network control** and **forwarding functions** enabling the network control to become directly programmable and the **underlying infrastructure to be abstracted** for applications and network services.

The **OpenFlow (OF)** protocol is a foundational element for building SDN solutions.

SDN Architecture

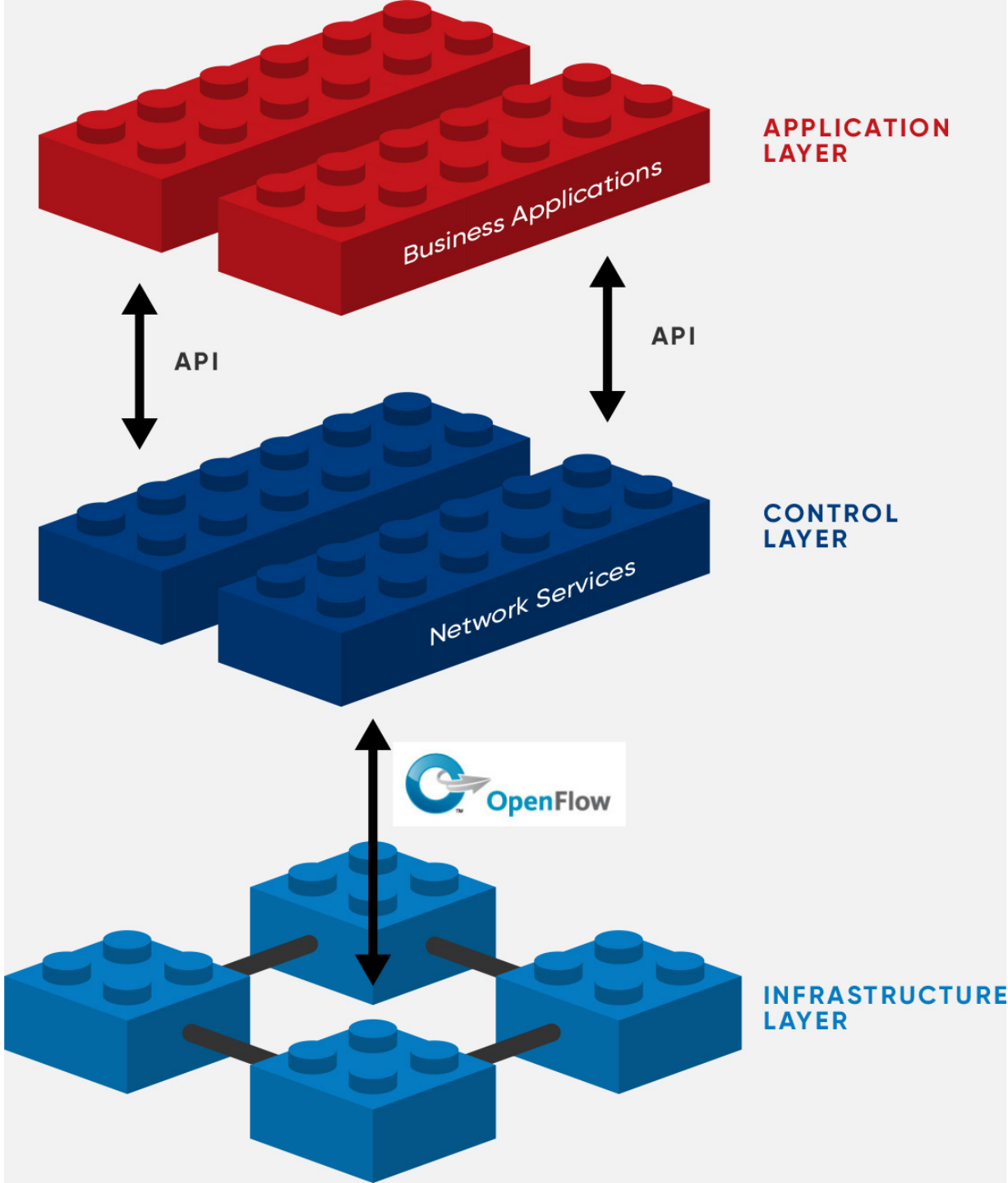
DIRECTLY PROGRAMMABLE
Network control is directly programmable because it is decoupled from forwarding functions.

AGILE
Abstracting control from forwarding lets administrators dynamically adjust network-wide traffic flow to meet changing needs.

CENTRALLY MANAGED
Network intelligence is (logically) centralized in software-based SDN controllers that maintain a global view of the network, which appears to applications and policy engines as a single, logical switch.

PROGRAMMATICALLY CONFIGURED
SDN lets network managers configure, manage, secure, and optimize network resources very quickly via dynamic, automated SDN programs, which they can write themselves because the programs do not depend on proprietary software.

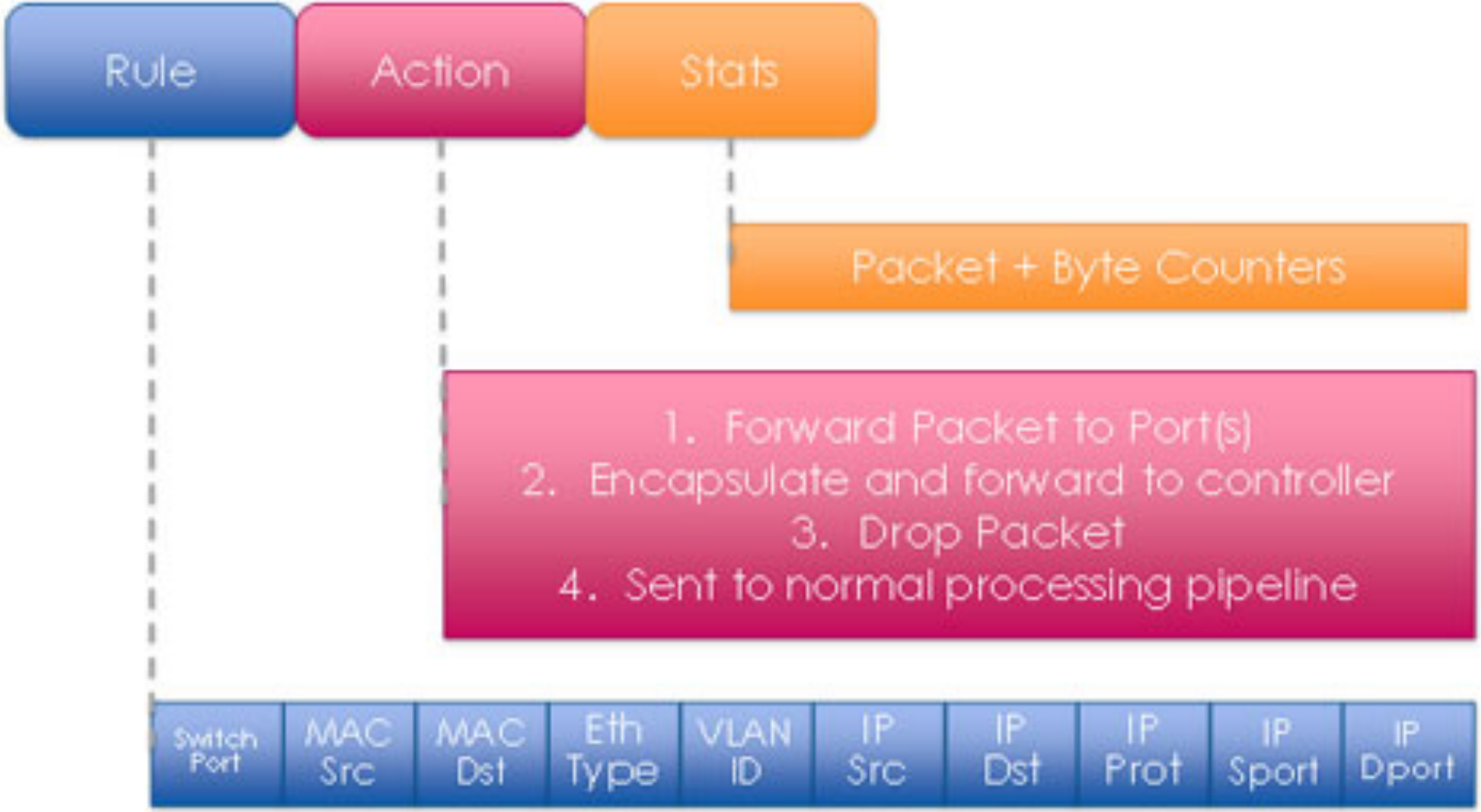
OPEN STANDARDS-BASED AND VENDOR-NEUTRAL
When implemented through open standards, SDN simplifies network design and operation because instructions are provided by SDN controllers instead of multiple, vendor-specific devices and protocols.



OpenFlow (OF)

- Original concept @ Stanford University (2008)
- One of the first SDN standards
- Managed by the Open Networking Foundation (ONF)
- Enables the SDN controller to directly interact with the forwarding plane of network devices
- Since it’s release, multiple companies and open source projects, like the OpenDaylight Project, support OF

Flow-Table Manipulation in an OF Switch



Benefits of OF

Programmability

- Enable innovation/differentiation
- Accelerate new features and services introduction

Centralized Intelligence

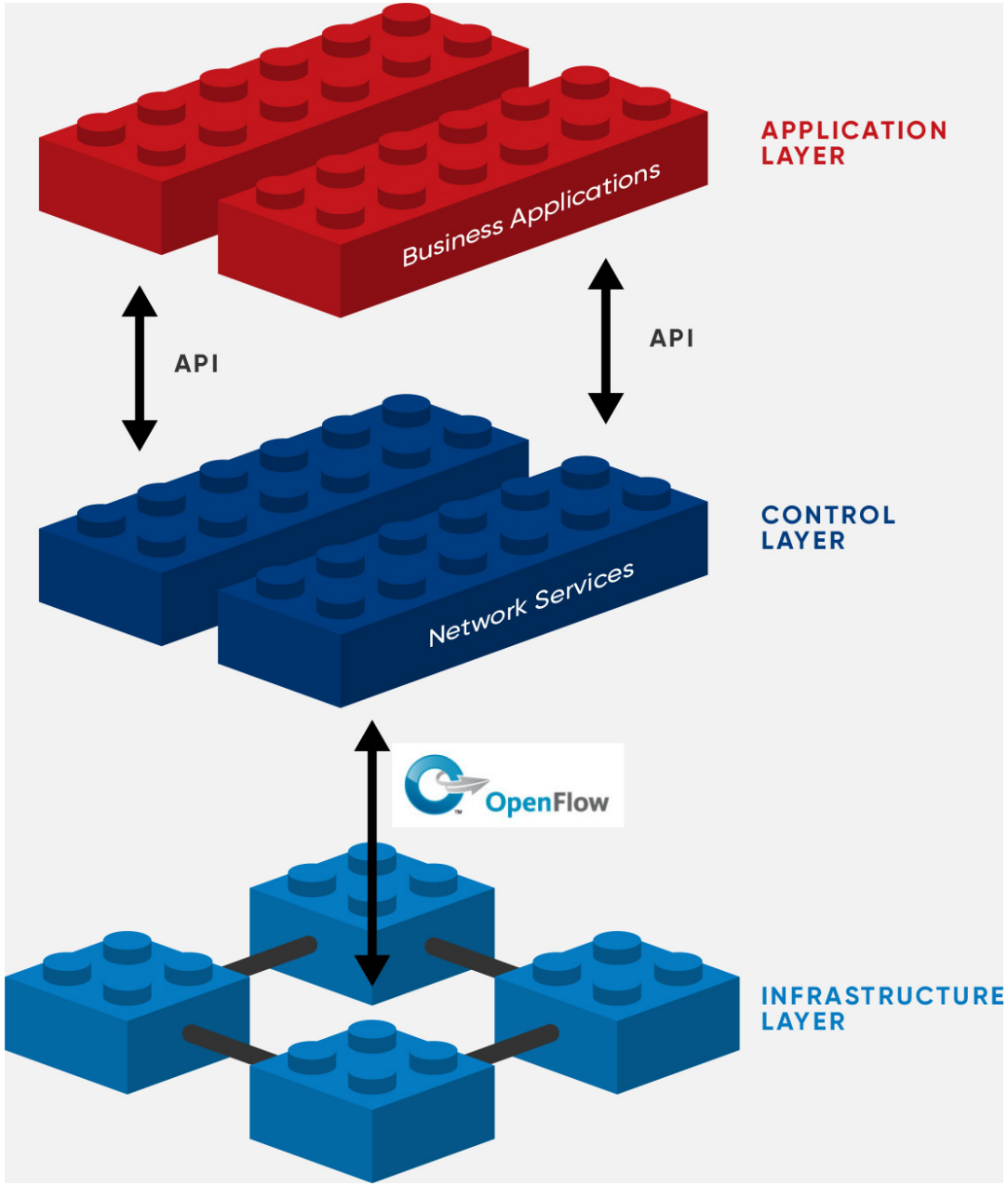
- Simplify provisioning
- Optimize performance
- Granular policy management

Abstraction

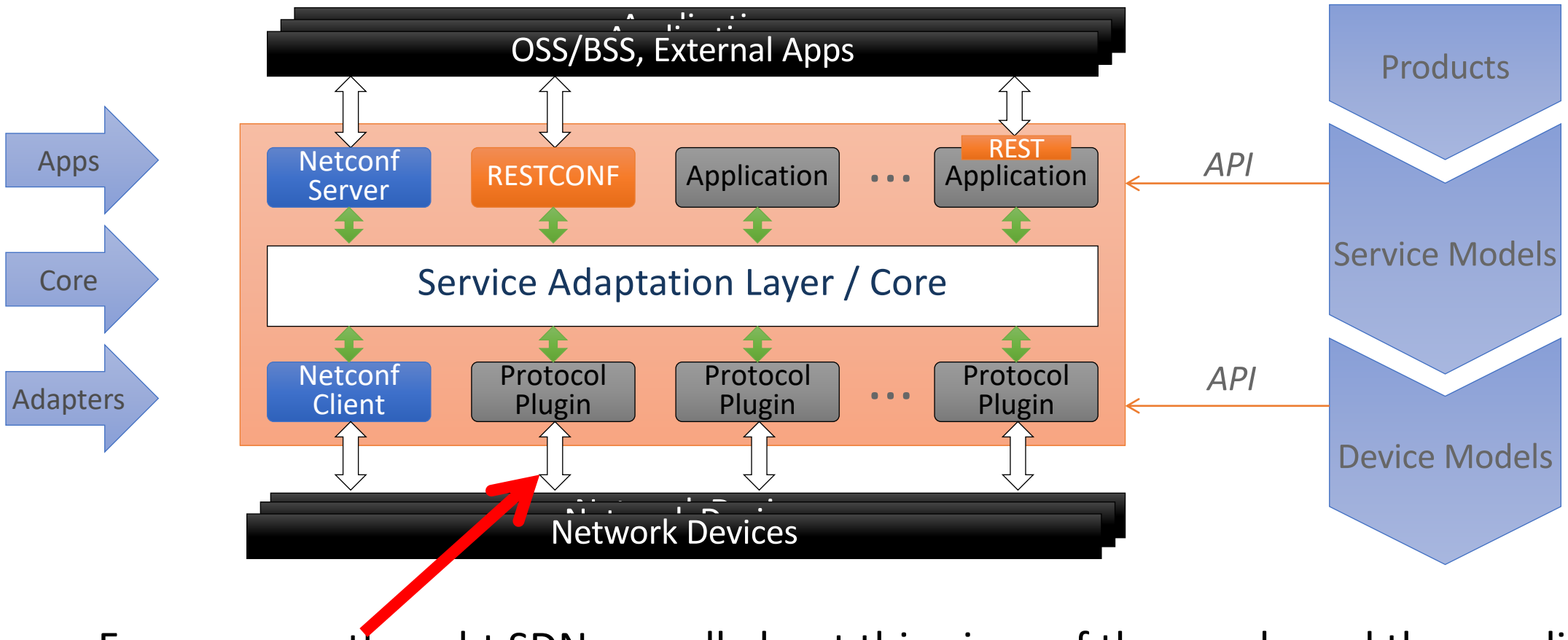
- Decoupling of Hardware & Software, Control plane & forwarding, and Physical & logical config

SDN Controller

SDN Controller positioning in the SDN Architecture



SDN Controllers: Modularity & Layering



For years we thought SDN was all about this piece of the puzzle and then realized the protocols were limiting innovation

OpenDaylight (ODL) Platform Overview

On April 8, 2013, the open-source foundation ODL, which is part of the Linux Foundation, was announced.

ODL is a SDN Controller based on Java code

ODL is a modular open platform for customizing and **automating networks** of any size and scale. The ODL Project arose out of the SDN movement, with a clear focus on **network programmability**.

It supports OF and other southbound APIs and includes critical features, such as high-availability and clustering

ODL Today

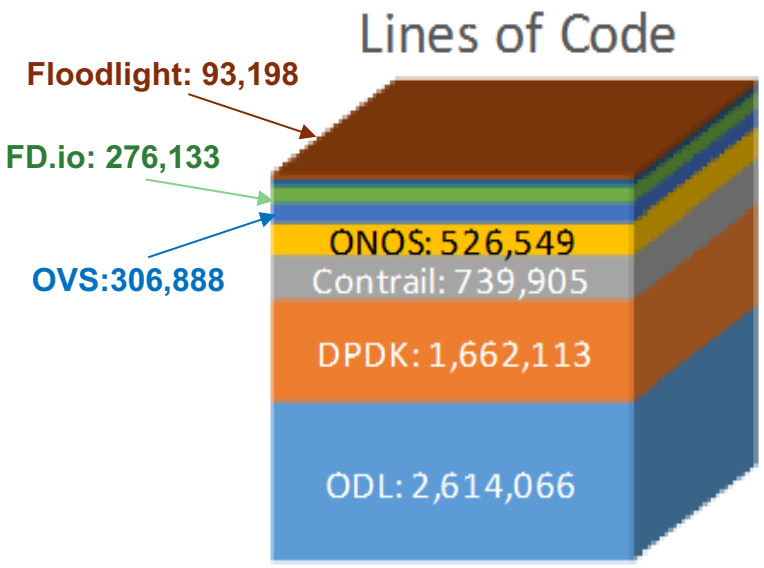
As part of Linux Foundation Networking (LFN), ODL is driven by a global, collaborative community of **vendor and user organizations** that continuously adapts to support the industry's broadest set of SDN and NFV use cases.

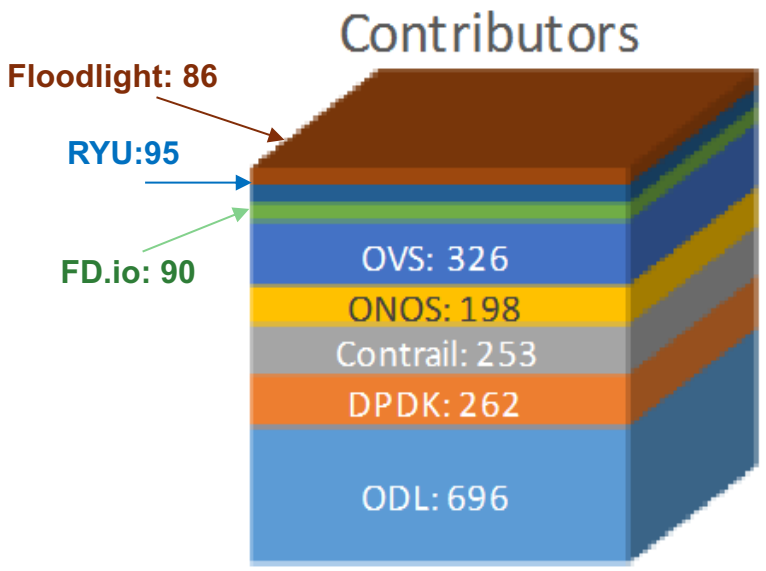
With over 1000 developers, and supporting approximately 1 billion subscribers around the world, ODL is quickly evolving integrated toolchains for leading use cases.

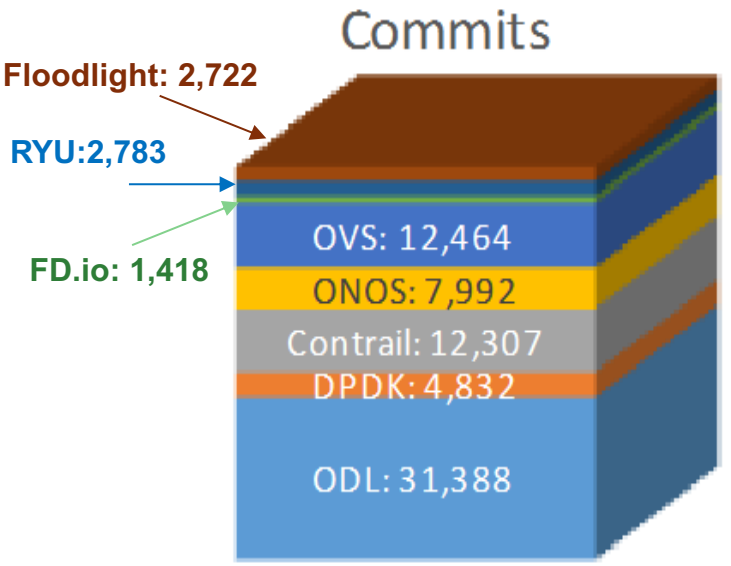
ODL code has been integrated or embedded in **more than 35 vendor solutions and apps**, and can be utilized within a range of services. It is also at the core of broader open source frameworks, including **ONAP**, **OpenStack**, and **OPNFV**.

ODL Today

- The biggest networking OSS project by any measure (www.openhub.net):

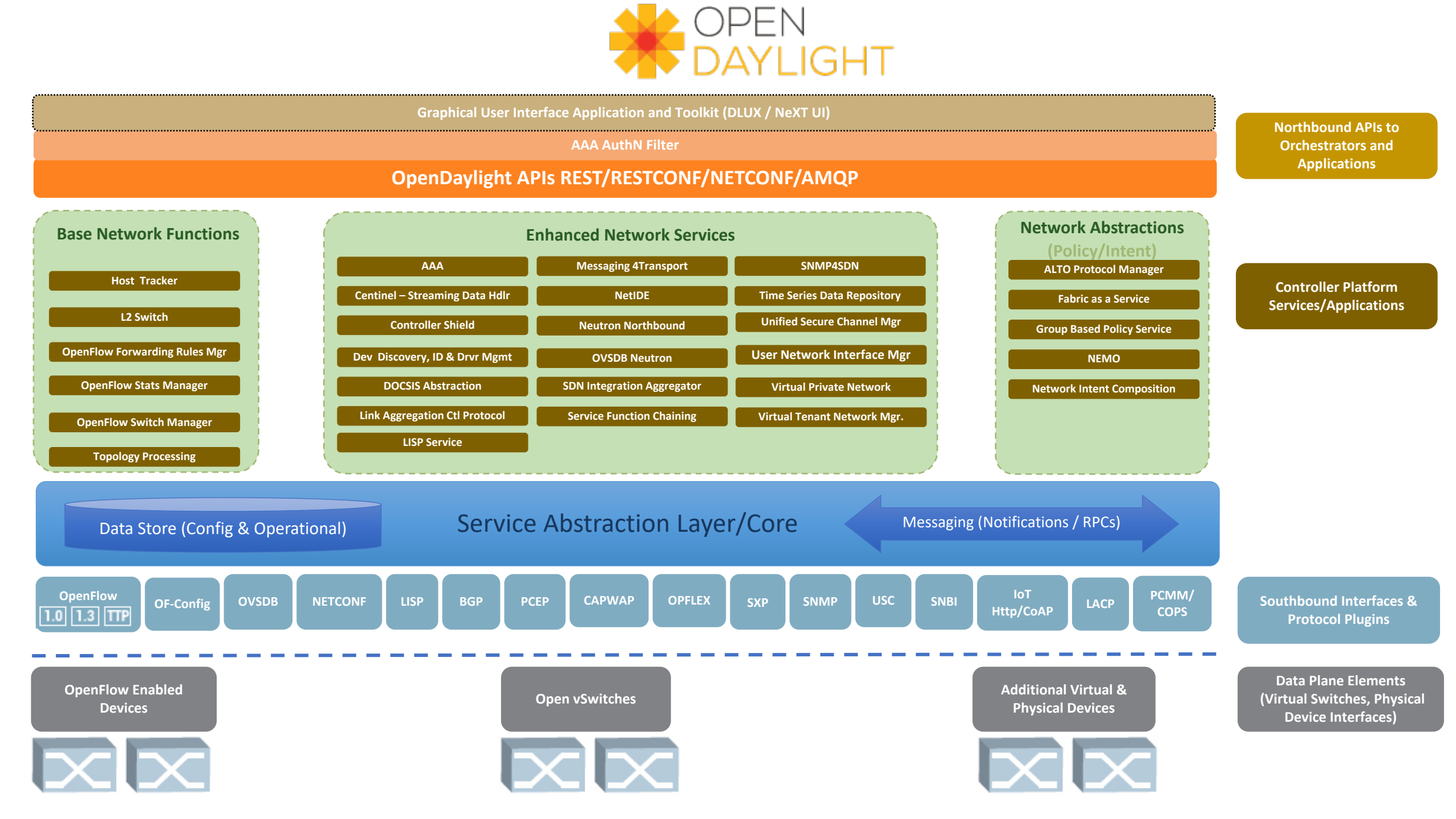




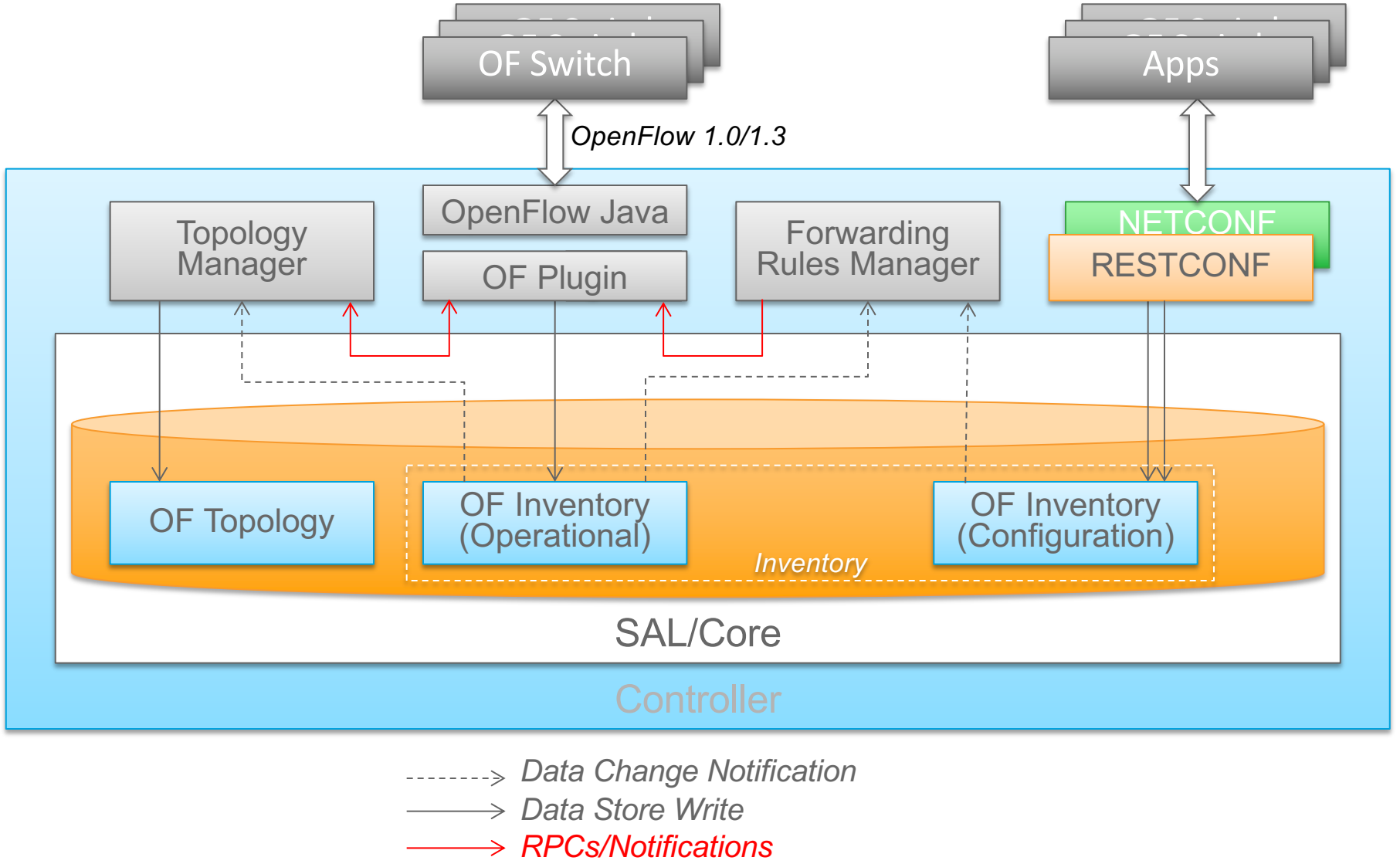


- 150+ known* deployments by 20+ companies
  - SW / Equipment vendors, SPs, …
  - Networking, entertainment, energy management, …

* = reported to Linux Foundation

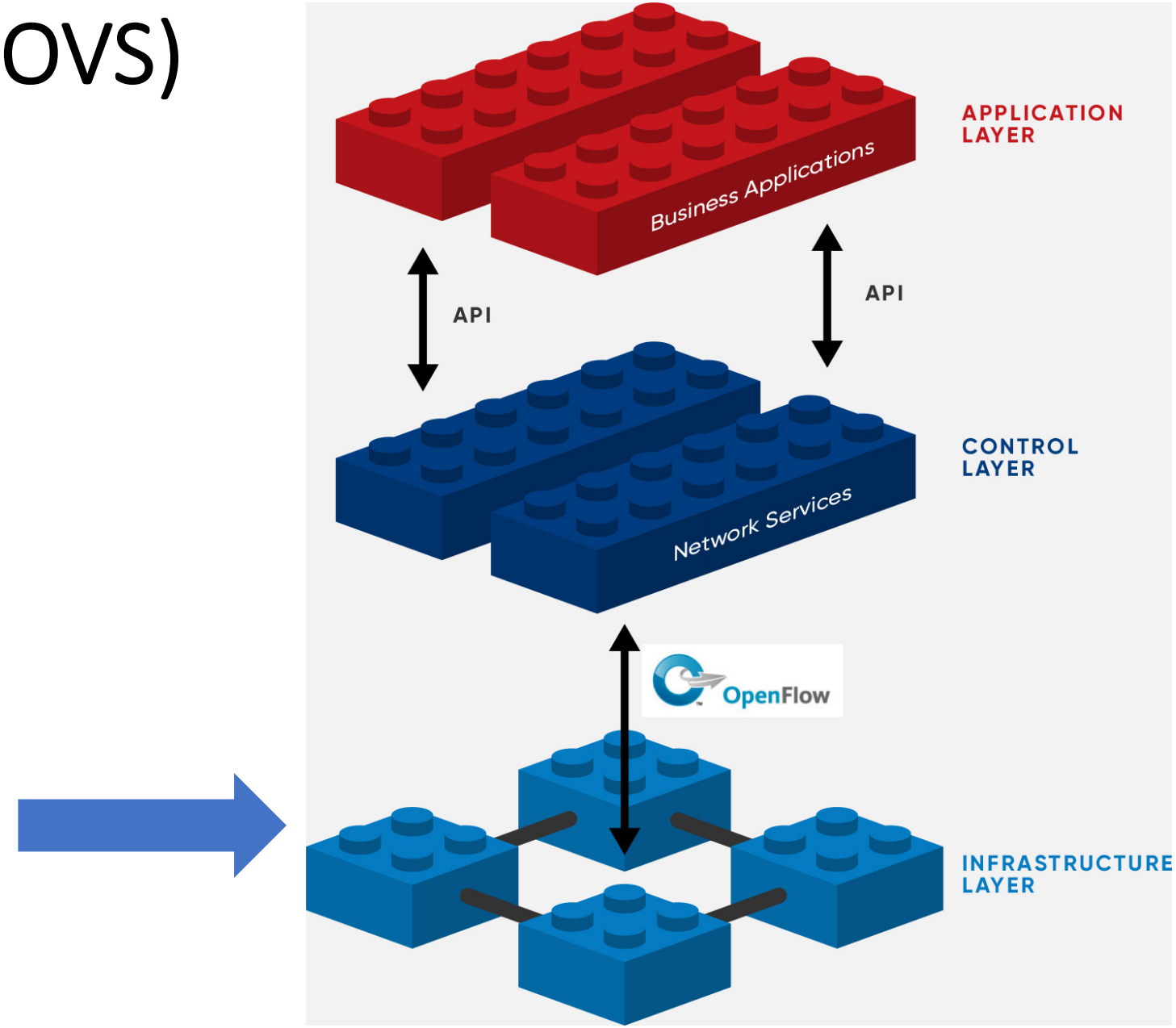

OPEN DAYLIGHT
Graphical User Interface Application and Toolkit (DLUX / NeXT UI)
AAA AuthN Filter
OpenDaylight APIs REST/RESTCONF/NETCONF/AMQP
Northbound APIs to Orchestrators and Applications
Base Network Functions
Host Tracker
L2 Switch
OpenFlow Forwarding Rules Mgr
OpenFlow Stats Manager
OpenFlow Switch Manager
Topology Processing
Enhanced Network Services
AAA
Centinel – Streaming Data Hdlr
Controller Shield
Dev Discovery, ID & Drvr Mgmt
DOCSIS Abstraction
Link Aggregation Ctl Protocol
LISP Service
Messaging 4Transport
NetIDE
Neutron Northbound
OVSDB Neutron
SDN Integration Aggregator
Service Function Chaining
SNMP4SDN
Time Series Data Repository
Unified Secure Channel Mgr
User Network Interface Mgr
Virtual Private Network
Virtual Tenant Network Mgr.
Network Abstractions (Policy/Intent)
ALTO Protocol Manager
Fabric as a Service
Group Based Policy Service
NEMO
Network Intent Composition
Controller Platform Services/Applications
Data Store (Config & Operational)
Service Abstraction Layer/Core
Messaging (Notifications / RPCs)
OpenFlow 1.0 1.3 TTP
OF-Config
OVSDB
NETCONF
LISP
BGP
PCEP
CAPWAP
OPFLEX
SXP
SNMP
USC
SNBI
IoT Http/CoAP
LACP
PCMM/ COPS
Southbound Interfaces & Protocol Plugins
OpenFlow Enabled Devices
Open vSwitches
Additional Virtual & Physical Devices
Data Plane Elements (Virtual Switches, Physical Device Interfaces)

Example Processing Pipeline: OpenFlow



Open vSwitch (OVS)

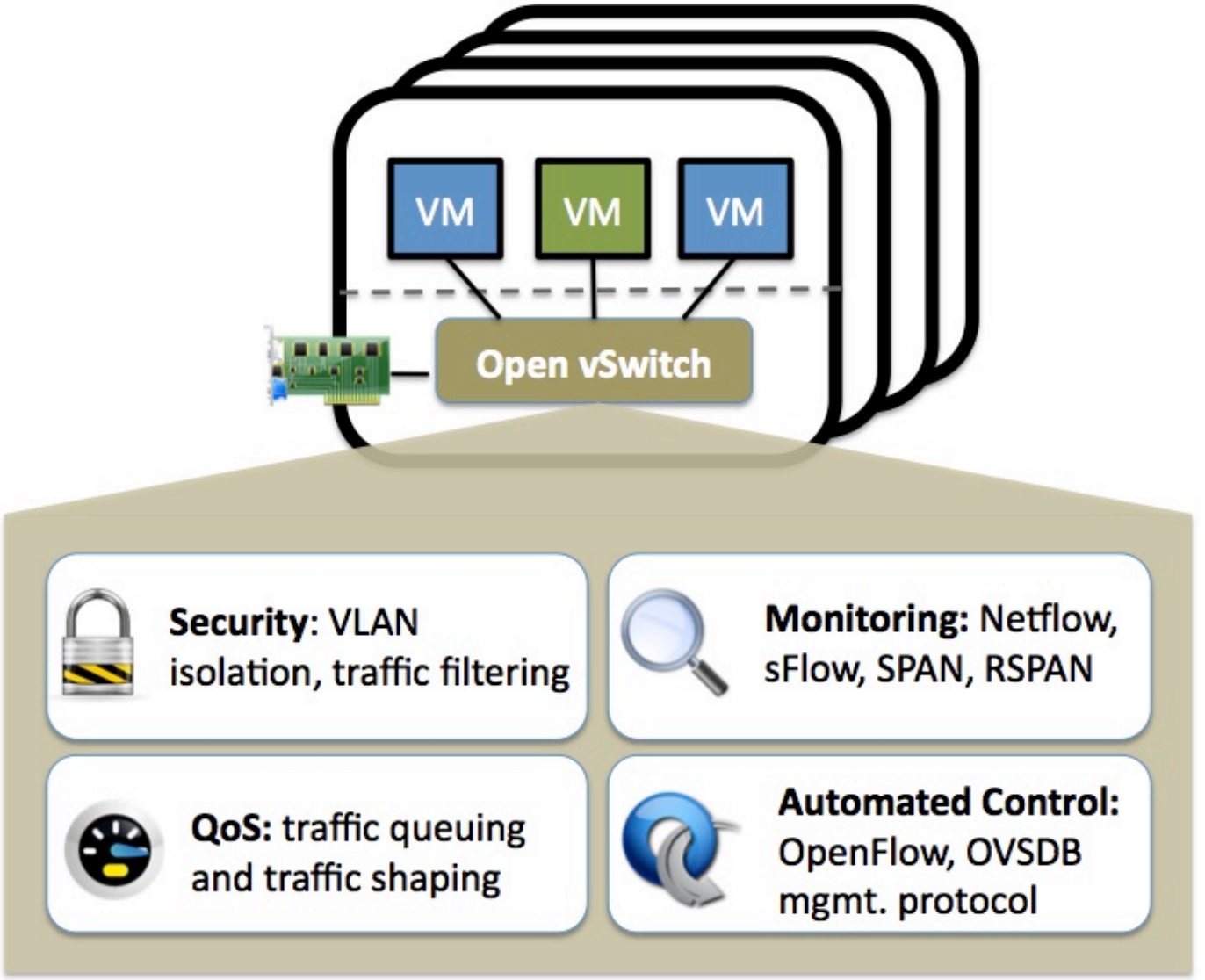
OVS positioning in the SDN Architecture



OVS Switch

- OVS is a production quality, multilayer virtual switch
- Designed to enable massive network automation through programmatic extension
- Still supporting standard management interfaces and protocols (e.g. NetFlow, sFlow, IPFIX, RSPAN, CLI, LACP, 802.1ag)
- Code written in platform-independent C and is easily ported to other environments

OVS Architecture



OVS Concepts

- A switch contains ports
- A port may have one or more interfaces (bonding allows more than once interface per port)
- Packets are forward by flow

Identifying Flows

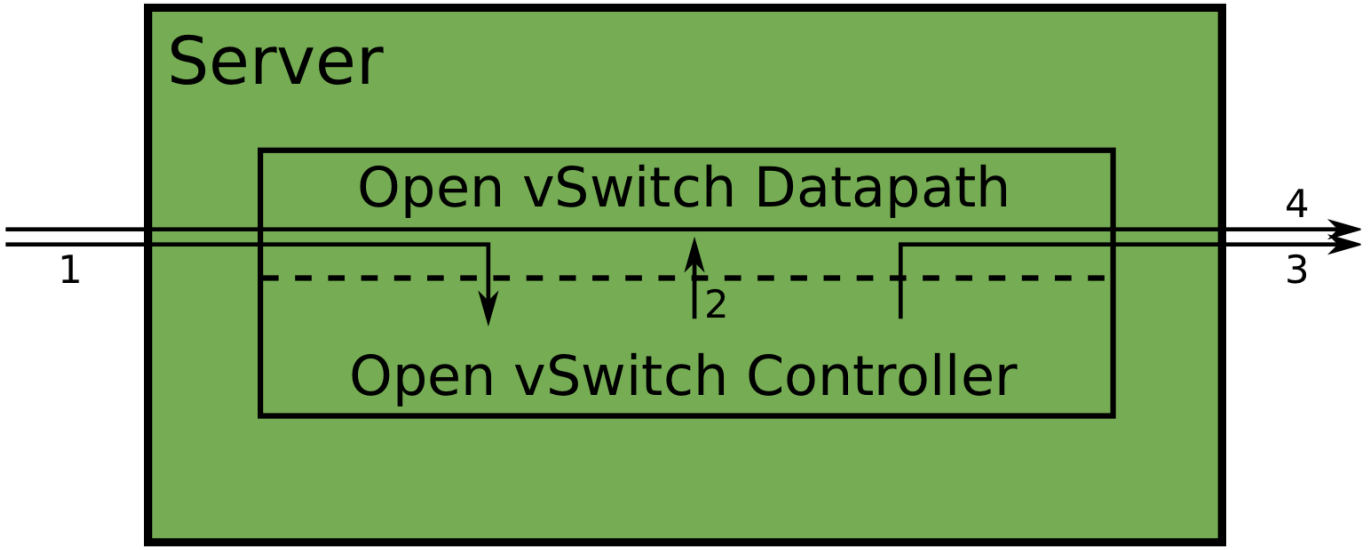
A flow may be identified by any combination of Tunnel ID

- IPv6 ND target
- IPv4 or IPv6 source address
- IPv4 or IPv6 destination address
- Input port
- Ethernet frame type
- VLAN ID (802.1Q)
- TCP/UDP source port
- TCP/UDP destination port
- Ethernet source address
- Ethernet destination address
- IP Protocol or lower 8 bits of ARP ppcode
- IP ToS (DSCP field)
- ARP/ND source hardware address
- ARP/ND destination hardware address

Forwarding Flows

1. The first packet of a flow is sent to the controller
2. The controller programs the datapath's actions for a flow
   - Usually one, but may be a list
   - Actions include:
     - Forward to a port or ports, mirror
     - Encapsulate and forward to controller
     - Drop
3. And returns the packet to the datapath
4. Subsequent packets are handled directly by the datapath

Forwarding Flows (Graphical Representation)



OVS Base Commands

- **ovs-vsctl:** Used for configuring the ovs-vswitchd configuration database (known as ovs-db)
- **ovs-ofctl:** A command line tool for monitoring and administering OpenFlow switches
- **ovs-dpctl:** Used to administer Open vSwitch datapaths
- **ovs–appctl:** Used for querying and controlling Open vSwitch daemons

OVS Show Commands

ovs-vsctl –V: Prints the current version of openvswitch.

ovs-vsctl show: Prints a brief overview of the switch database configuration.

ovs-vsctl list-br: Prints a list of configured bridges

ovs-vsctl list-ports <bridge>: Prints a list of ports on a specific bridge.

ovs-vsctl list interface: Prints a list of interfaces.

Use Case: IoT Controller & Data Collector