Software Defined Networking

Overview

- Planes of Networking
- What is SDN?
- Why do we need SDN?
- SDN Use Cases
- SDN Controllers

Planes of Networking

- **Data Plane**: All activities involving as well as resulting from data packets sent by the end user
  - Forwarding
  - Fragmentation and reassembly
  - Replication for multicasting
- **Control Plane**: All activities that are necessary to perform data plane activities but do not involve end-user data packets
  - Making routing tables
  - Setting packet handling policies (e.g., security)
  - Base station beacons announcing availability of services
  - IP address assignment (e.g., DHCP)

Source: D. Kreutz et al., "Software-Defined Networking: A Comprehensive Survey“, in Proc. of the IEEE, vol. 103, no. 1, pp. 14-76, Jan. 2015. doi: 10.1109/JPROC.2014.2371999

Planes of Networking

- **Management Plane**: All activities related to provisioning and monitoring of the networks
  - Fault, Configuration, Accounting, Performance, and Security (FCAPS)
  - Instantiate new devices and protocols (Turn devices on/off)
- **Role of each Plane**
  - Management Plane → **definition** of network policy
  - Control Plane → **enforcement** of network policy
  - Data Plane → **execution** of network policy by forwarding data accordingly

  Network policy is a broad term that describes the operation rules (network constraints, configurations, and settings)

    - E.g., Access Control Lists (ACLs), Quality of Service (QoS)

Source: D. Kreutz et al., "Software-Defined Networking: A Comprehensive Survey“, in Proc. of the IEEE, vol. 103, no. 1, pp. 14-76, Jan. 2015. doi: 10.1109/JPROC.2014.2371999

Traditional IP Networks

- **Control plane** and **Data plane** embedded in the networking devices
  - reducing flexibility and hindering innovation and evolution of the networking infrastructure
- Highly decentralized
  - Important aspect in the early days of the Internet
  - Was the best way to guarantee network resilience
- Examples of transition inertia
  - Transition from IPv4 to IPv6 → merely a protocol update
  - New routing protocol can take 5 to 10 years to be fully designed, evaluated and deployed

Software Defined Networking (SDN)

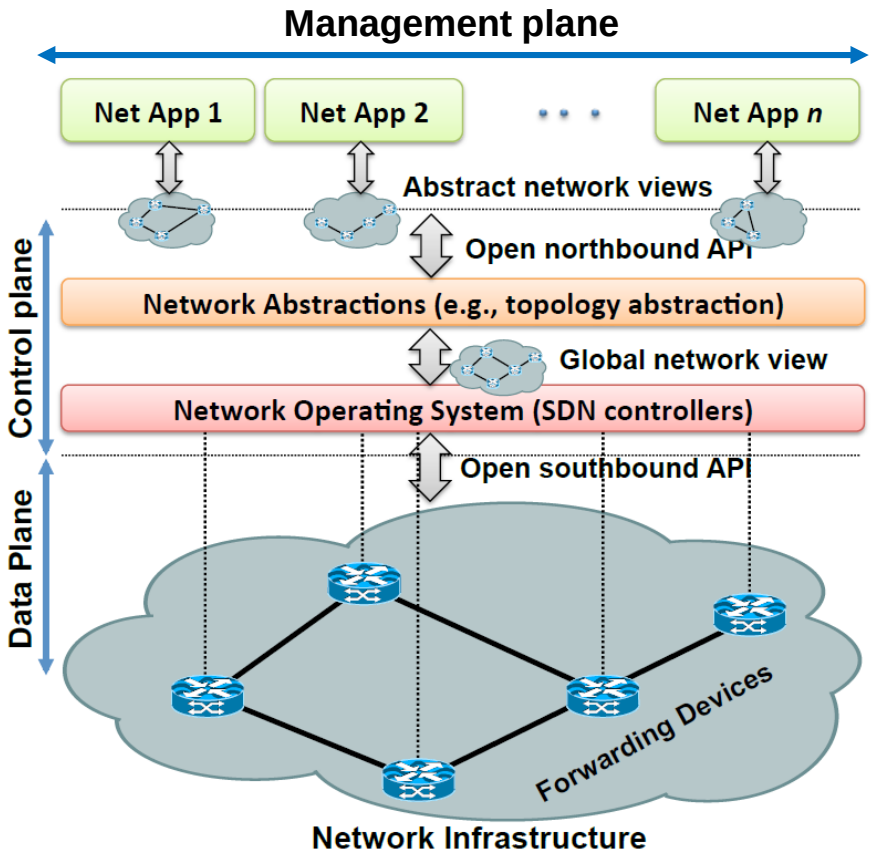
Open Networking Foundation definition[1]:

SDN is 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 protocol is a foundational element for building SDN solutions.*

[1] Open Networking Forum, https://www.opennetworking.org [Accessed: 14th May 2021]

Origins of SDN

- SDN originated from OpenFlow
- Centralized Controller
  - Easy to program
  - Change routing policies on the fly
- Initially, SDN equal to:
  - Separation of Control and Data Plane
  - Centralization of Control
  - OpenFlow to talk to the data plane
- Then, the definition has evolved



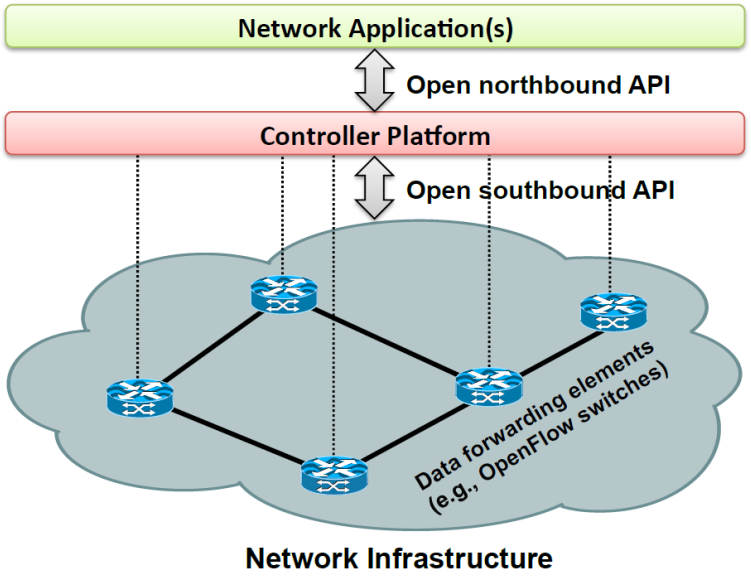
Example:
Net App 1 – Security
Net App 2 – Network Monitoring
Net App 3 – Bandwidth Management

Features that Define SDN

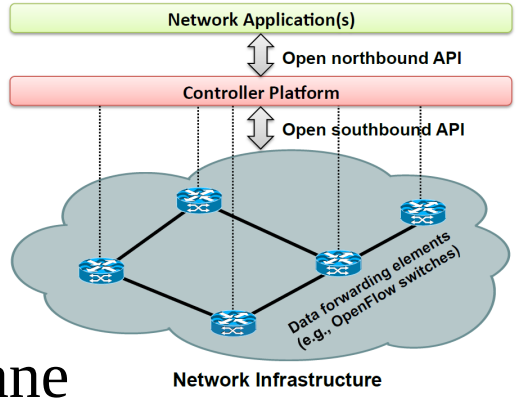
- **Abstract the Hardware**: No dependence on physical infrastructure
  - software Application Programming Interface (API)
- **Programmable**: Shift away from static manual operation to fully configurable and dynamic
- **Centralized Control of Policies**: Policy delegation and management

Software Defined Networking (Zoom In)

- Separation of network's control logic (control plane) from underlying routers and switches forwarding traffic (the data plane)
- With the separation of control and data planes
  - Switches become simple forwarding devices (white boxes)
  - Control logic implemented in logically centralized controller (or network operating system)
- Simplifies policy enforcement and network (re)configuration and evolution



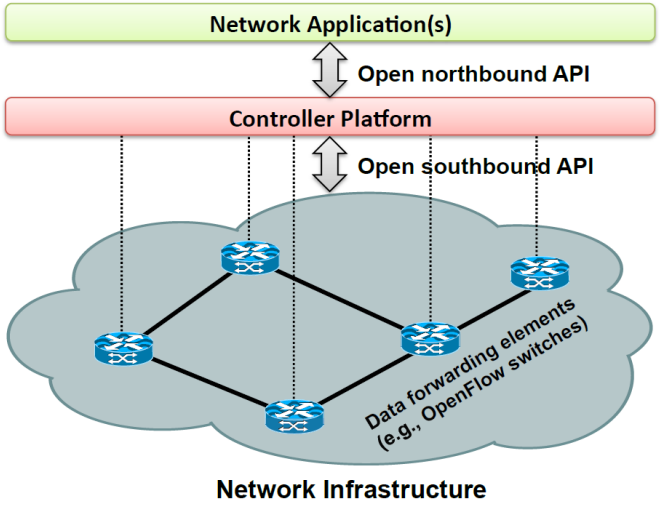
Software Defined Networking (Zoom In)



- Separation of the control plane and the data plane
  - Realized using well-defined programming interface between the switches and the SDN controller – e.g., OpenFlow API (supported by most vendors)
- OpenFlow Switch
  - One or more tables of packet-handling rules (flow table)
  - Each rule matches a subset of the traffic and performs certain actions (dropping, forwarding, modifying, etc.) on the traffic
  - Depending on the rules defined by Network Application, forwarding device can act as router, switch, firewall, NAT, etc.

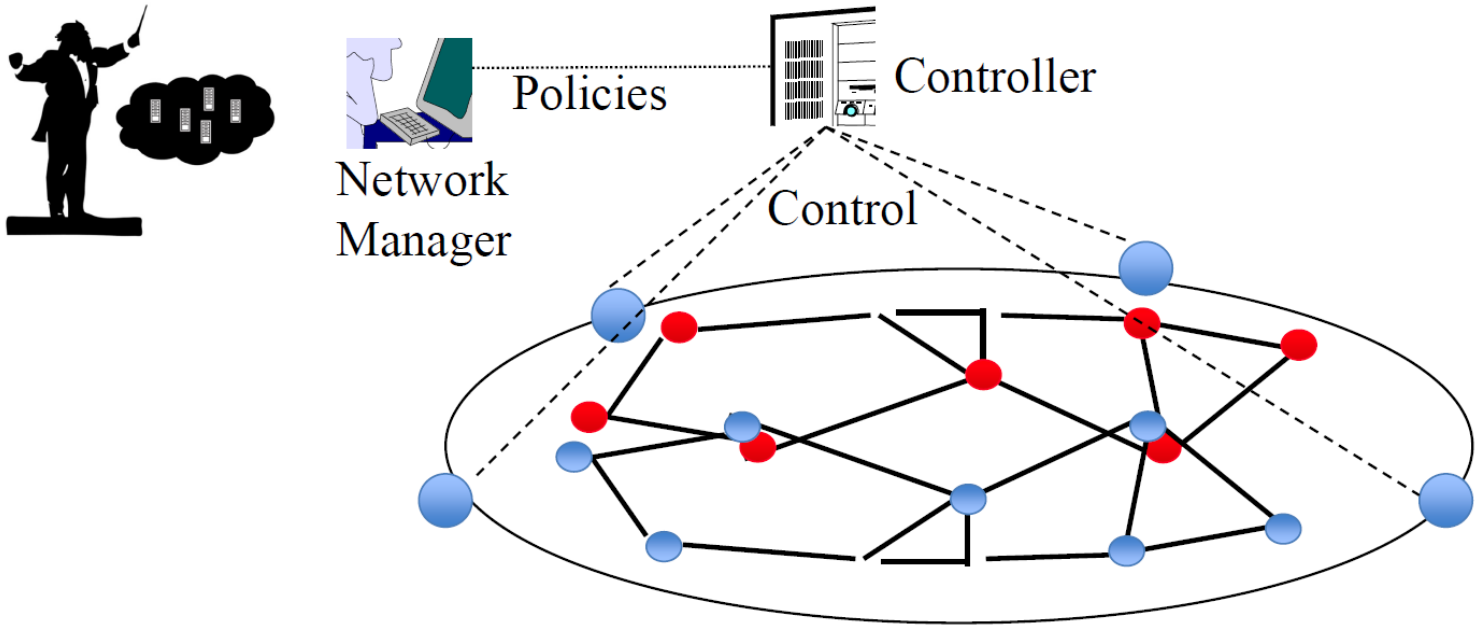
Source: D. Kreutz et al., "Software-Defined Networking: A Comprehensive Survey“, in Proc. of the IEEE, vol. 103, no. 1, pp. 14-76, Jan. 2015. doi: 10.1109/JPROC.2014.2371999

Software Defined Networking (Zoom In)



- Separation of concerns
  - Definition of network policies – **management** plane
  - Implementation of network policies in forwarding hardware – **control** plane
  - Forwarding of traffic – **data** plane
- This is key to the desired flexibility
  - Breaks network control problem into simpler components
  - Simplifies network management
  - Facilitates network evolution and innovation

Software Defined Networking



- Centralized Programmable Control Plane
- Allows automated orchestration (provisioning) of a large number of virtual resources (machines, networks, storage)
- Large network topologies can be created on demand

Why do we need SDN?

1. Virtualization: Use network resource without worrying where it is physically located, how much it is, how it is organized, etc.

- Abstraction → Virtualization

2. Orchestration: Be able to control and manage thousands of devices with one command

- e.g., in data centers, 4G/5G mobile networks

3. Programmable: Be able to change behavior on the fly

4. Dynamic Scaling: Be able to change size, quantity

- Virtualization → Scaling

5. Automation: To lower OpEx → minimize manual involvement

- Troubleshooting, reduce downtime, policy enforcement
- Provisioning/Re-provisioning/Segmentation of resources
- Add new workloads, sites, devices, and resources

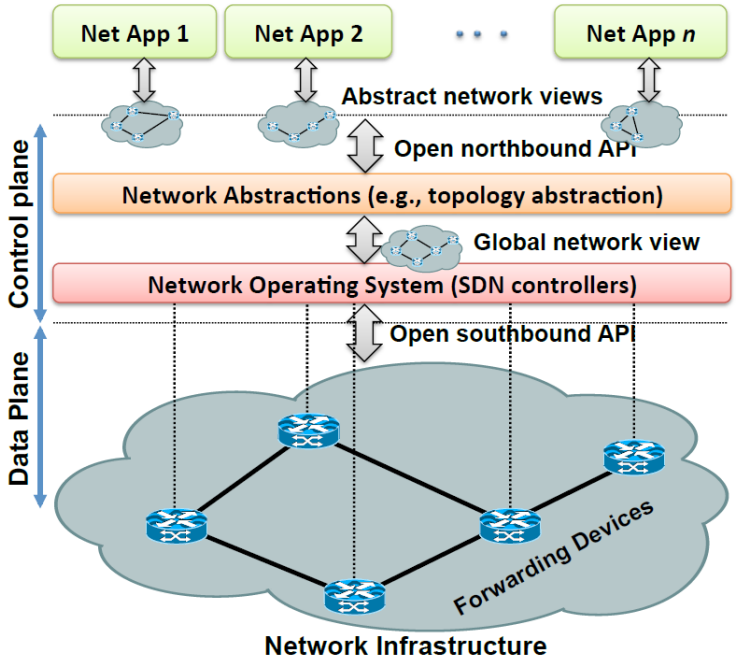
Why do we need SDN?

6. **Visibility**: Monitor resources, connectivity
7. **Performance**: Optimize network device utilization
    - Bandwidth management
    - Load balancing
    - High utilization
    - Fast failure handling
8. **Multi-tenancy/Slicing**: Tenants need complete control over their addresses, topology, routing, security, Quality of Service (QoS), etc.
9. **Service Integration**: Load balancers, firewalls, Intrusion Detection Systems (IDS)
    - provisioned on demand and placed appropriately on the traffic path

Why do we need SDN?

10. Openness: Full choice of "How" mechanisms

- Abstraction: opposite of concrete → define tasks by APIs and not by how it should be done
  - E.g., send from A to B. Not use OSPF.



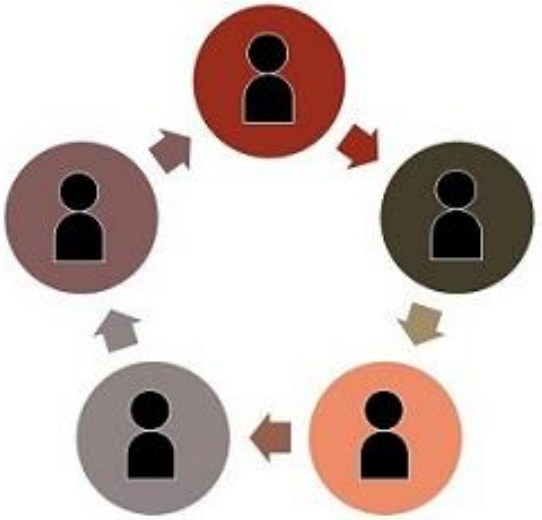
Source: Open Data Center Alliance, Open Data Center Alliance Master Usage Model: Software-Defined Networking Rev. 2.0, White Paper, 2014.

Software Defined Anything (SDx)

- Software Defined Things
  - Software Defined Networking (SDN)
  - Software Defined Datacenter (SDDC)
  - Software Defined Storage (SDS)
  - Software Defined Compute (SDC)
  - Software Defined Infrastructure (SDI)
  - Software Defined Radio (SDR)
  - ...

Source: http://www.vdicloud.nl/2015/04/06/software-defined-dockerized-springpath-halo-at-sfd7 [Accessed: 14th May 2021]

Centralized vs. Distributed



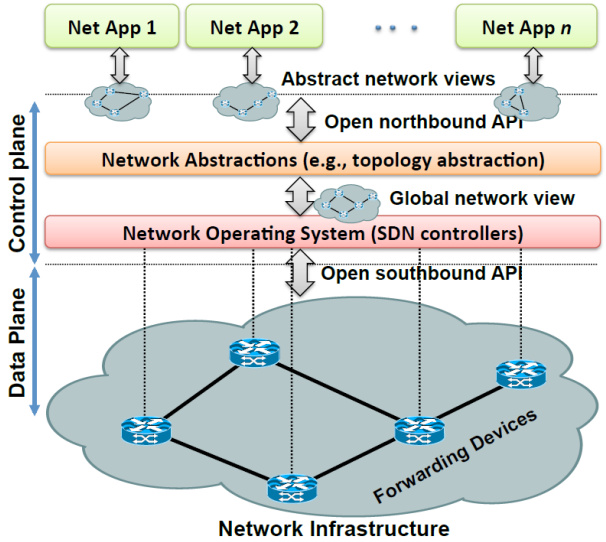
- Fast Response to changes
- Fast Consistency
- Less overhead ⇒ Scalable
- “Single Point of Failure”

- Time to converge
- Slow consistency
- “Not scalable”
- Fault Tolerant

Source: https://keydifferences.com/difference-between-centralization-and-decentralization.html [Accessed: 14th May 2021]

What SDN is not?

- SDN = OpenFlow
- SDN = Standard Southbound API
- SDN = Centralization of control plane
- SDN = Separation of Control and Data Planes



- All of these are mechanisms
- SDN is *not* about a mechanism
- SDN is a framework/paradigm → Many solutions

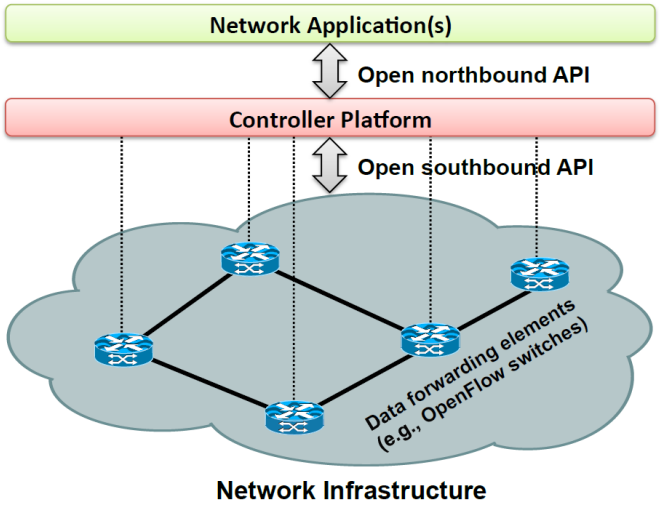
Confusions about SDN

- **Policies (What) vs. Control (How)**
  - Control = All bits and messages not sent by the user
  - In IP, control includes all header bits and all routing messages
- **Separation of Control Plane**
  - Elements have only data plane and have no brains
- **SDN vs. OpenFlow**
  - OpenFlow is the father of SDN but not SDN
- **SDN needs OpenFlow**
  - Other options can be used (e.g., OpFlex from Cisco)

Current SDN Debate: What vs. How?

- **SDN is easy if control is centralized but not necessary**
  - Distributed/hierarchical solutions may be required for fail-safe operation
- **Complete removal of control plane may be harmful**
  - Exact division of control plane between centralized controller and distributed forwarders is yet to be worked out
- **SDN is easy with a standard southbound protocol like OpenFlow**
  - OpenFlow is optional element of the SDN-like architectures
  - Many modern SDN-like solutions do not rely on OpenFlow
    - BGP and MPLS gaining momentum as SDN-enablers

SDN Controller



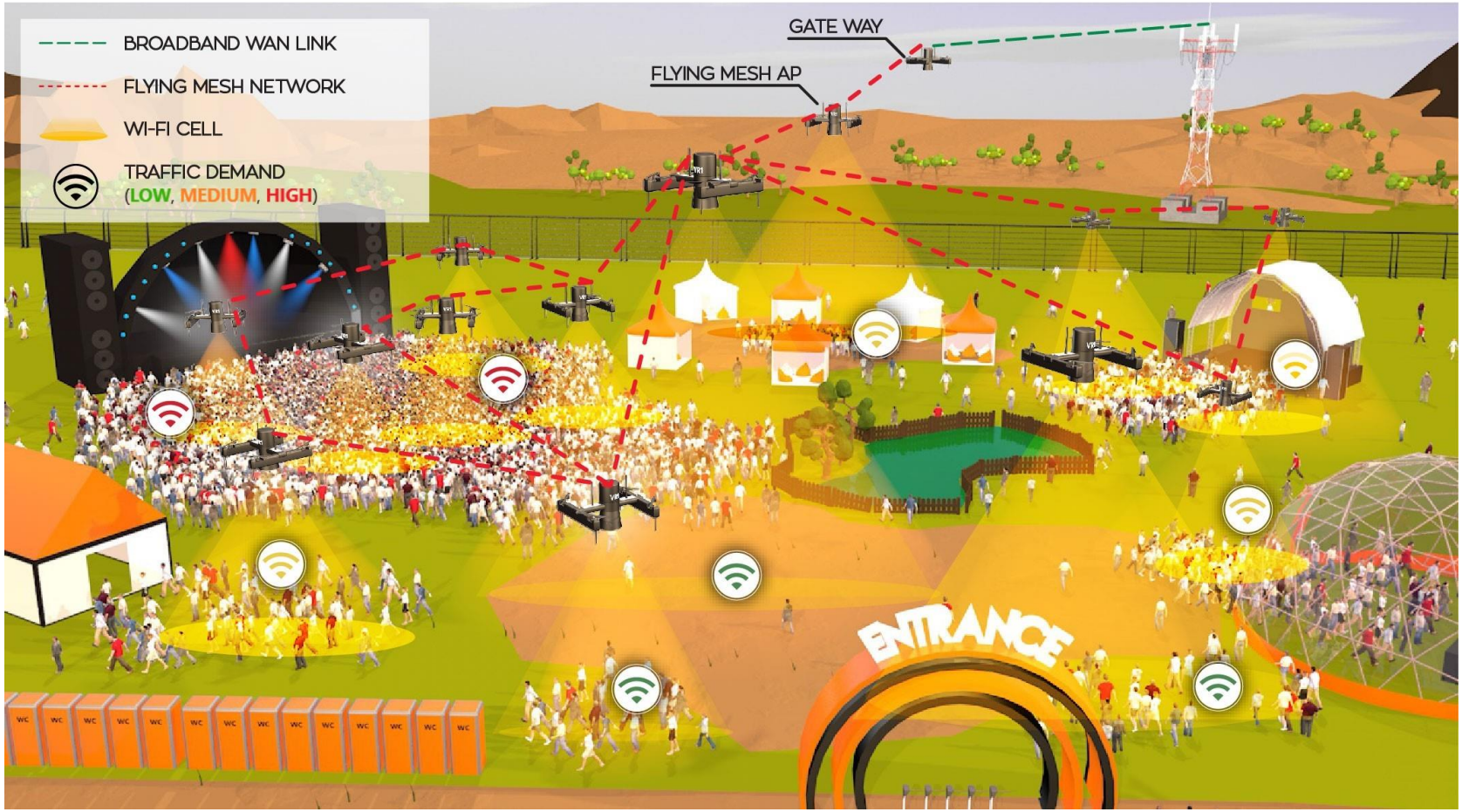
- Acts as the “network brain”
- Creates and manages policies and rules through Northbound API
- Relays information to switches and routers via Southbound API
- Easy evolution to Machine Learning oriented networking solutions
  - Centralized intelligence → high performance / quantum computing
  - Very active research work along these lines (e.g., 6G)

SDN Use Cases

- Data Center Virtual Networks
- Campus Virtual Networks
- Mobile and Wireless Networks – 5G, Wi-Fi
- Virtualized Customer Premises Equipment (vCPE)
- Content Acquisition / Video Streaming
  - Establish multicast forwarding from a sender to set of receivers
- Virtual Network Gateway (vBNG)
  - Provide connectivity between a private host and the Internet
- Bandwidth Calendaring
  - Establish tunnels with bandwidth guarantees between two points at a given time

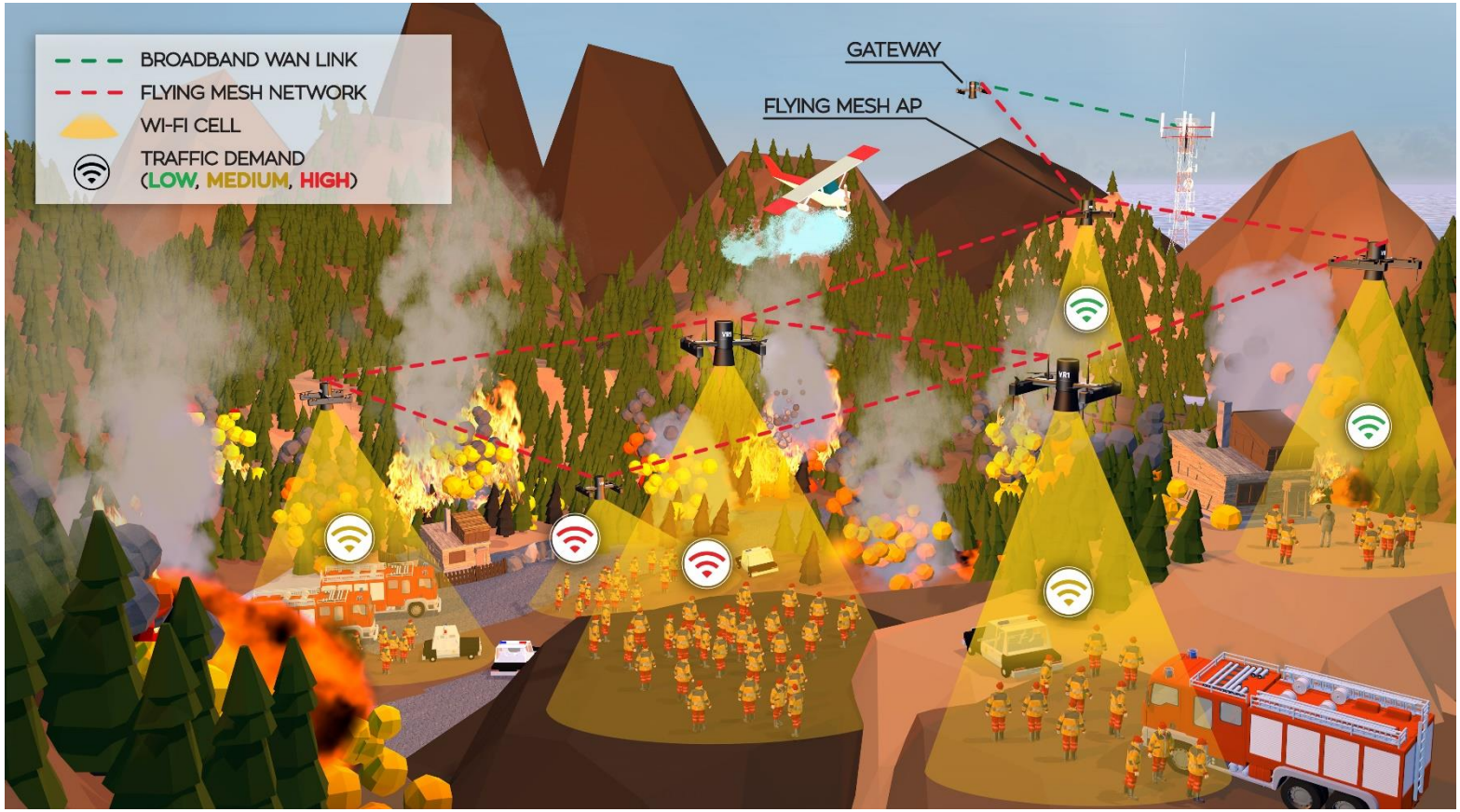
SDN Use Cases

- Flying Wireless Networks
  - drone positioning, routing, mobility management



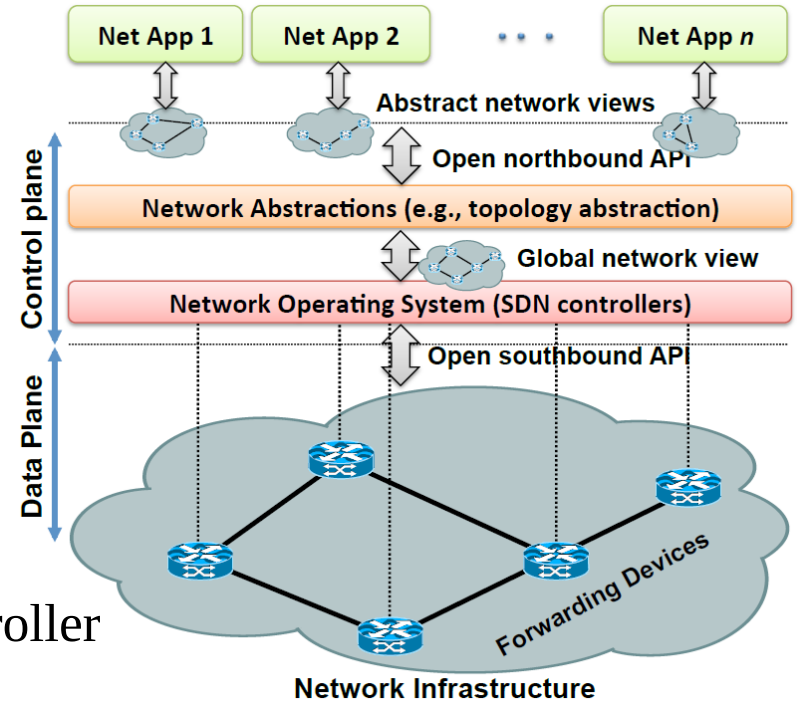
SDN Use Cases

- Flying Wireless Networks
  - drone positioning, routing, mobility management



SDN Controllers

- **Floodlight**
  - Java-based
    - Northbound API → REST API
    - Southbound API → OpenFlow API
- **ONOS**
  - Java-based, leading open source controller
    - Northbound API → REST, GUI, CLI
    - Southbound API → initially OpenFlow-only, now multi-protocol (REST, BGP, OSPF, ...)



SDN Controllers

- **OpenDaylight**
  - Multi-company collaboration under Linux foundation
  - Java-based
    - Northbound API → multiple APIs, including REST API
    - Southbound API → multiple APIs via plug-ins, including OpenFlow API

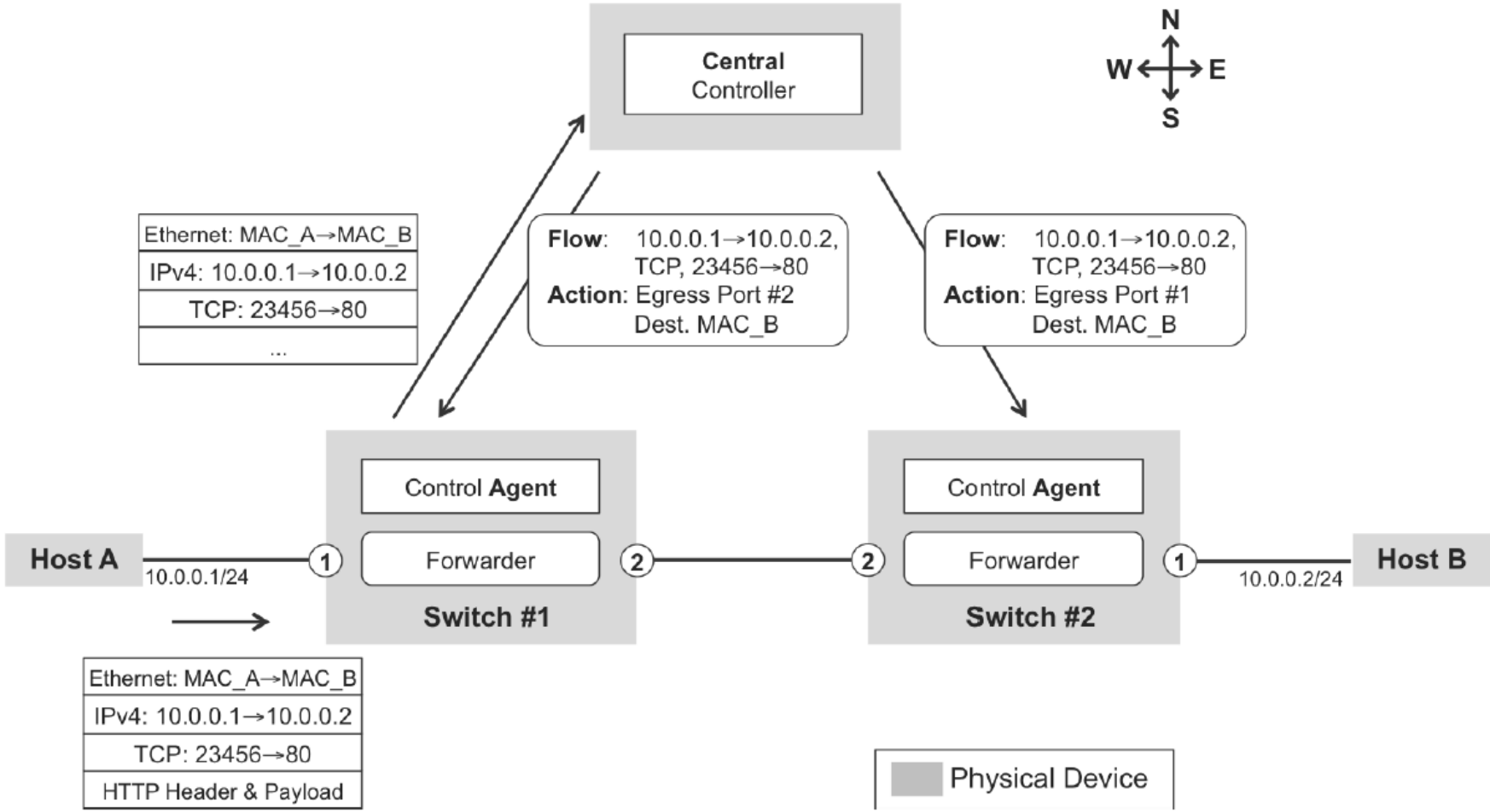
OpenFlow

- OpenFlow enables flow-based programmability of a forwarding engine
- Key ideas
  - Separation of control and data planes
  - Centralization of control
  - Flow-based control

OpenFlow – How it works

- Assumes that there is a central controller software running as a virtual machine (VM), container, or directly on the host OS of a server
- Controller must have IP connectivity to the switches by using either:
  - Out-of-band network that is *not* under the command of the controller
  - In-band network connection that relies on some pre-existing forwarding state (e.g., running some distributed routing protocol)
- The switches' control plane is connected to the central controller via an OpenFlow TCP session, whose purpose is to exchange OpenFlow messages

OpenFlow – How it works



The SDN Era

- Focus has typically been on the **“How”**
  - How to program low-level flows or how to configure a device
- However, what the industry really needs is a **focus on the “What”**
  - What is the intent
  - Intent-based networking is next logical progression of SDN
  - Network as a Service (NaaS)
    - network administrator defines desired state of the network and network orchestration software implements those policies

Summary

- SDN =**Abstraction** + **Programmability** + **Centralization**
- SDN = Disaggregation of hardware and software
- Many hardware and software based switches including Open vSwitch
- OpenFlow originated SDN but now many different Southbound and Northbound APIs, intermediate services and tools are being discussed and implemented by industry
- Myriad of SDN use cases
- Evolving from SDN to Intent-based Networking
  - Ongoing research work combining SDN and Machine Learning for smart networking solutions → 6G