



Introduction to Software-Defined Networking (SDN) and Network Programmability

By Minh Dang

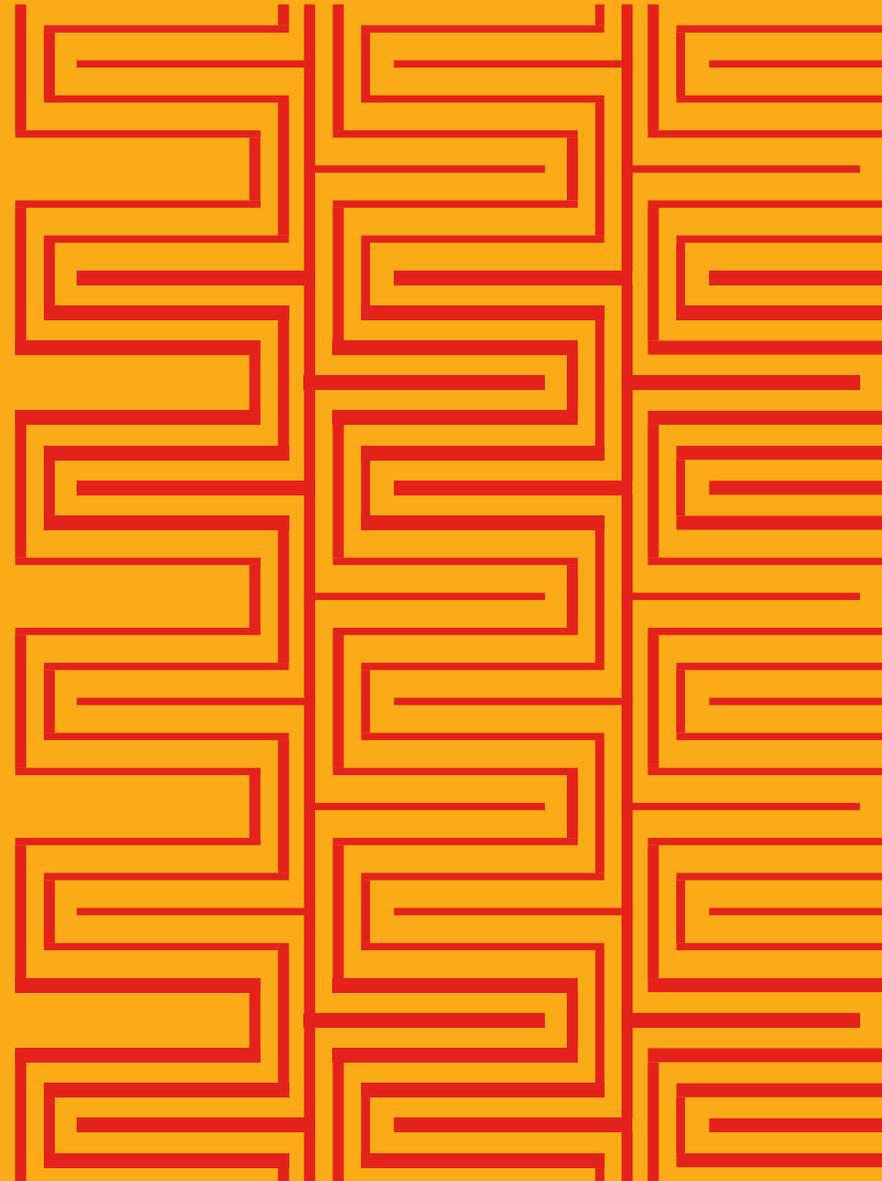


INTUITIVE

Agenda

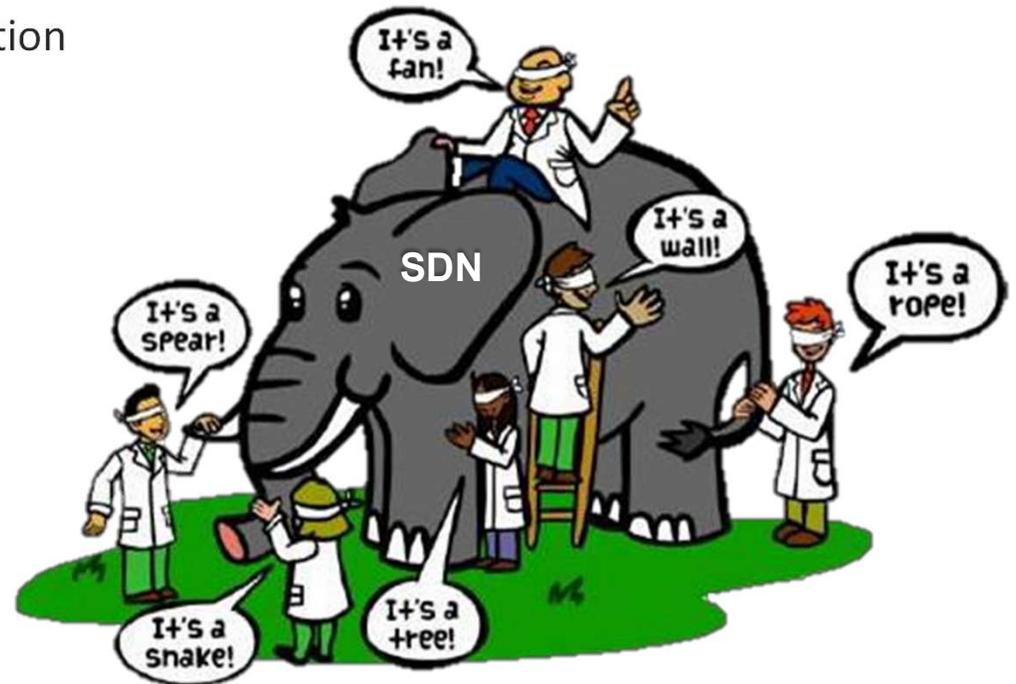
- Let's define SDN
- Key Concepts and Technologies
 - APIs
 - RESTCONF, NETCONF & YANG
 - Model Driven Telemetry
 - NFV

Let's Define
Software-Defined
Networking (SDN)



What is Software-Defined Networking?

- What is SDN about? Automation/Simplification
- Why? Savings!
- SDN is not a protocol
- SDN is not single industry standard
- SDN is an approach



Developers & Infrastructure Engineers Must Collaborate

Network APIs

DevOps

Infrastructure Management

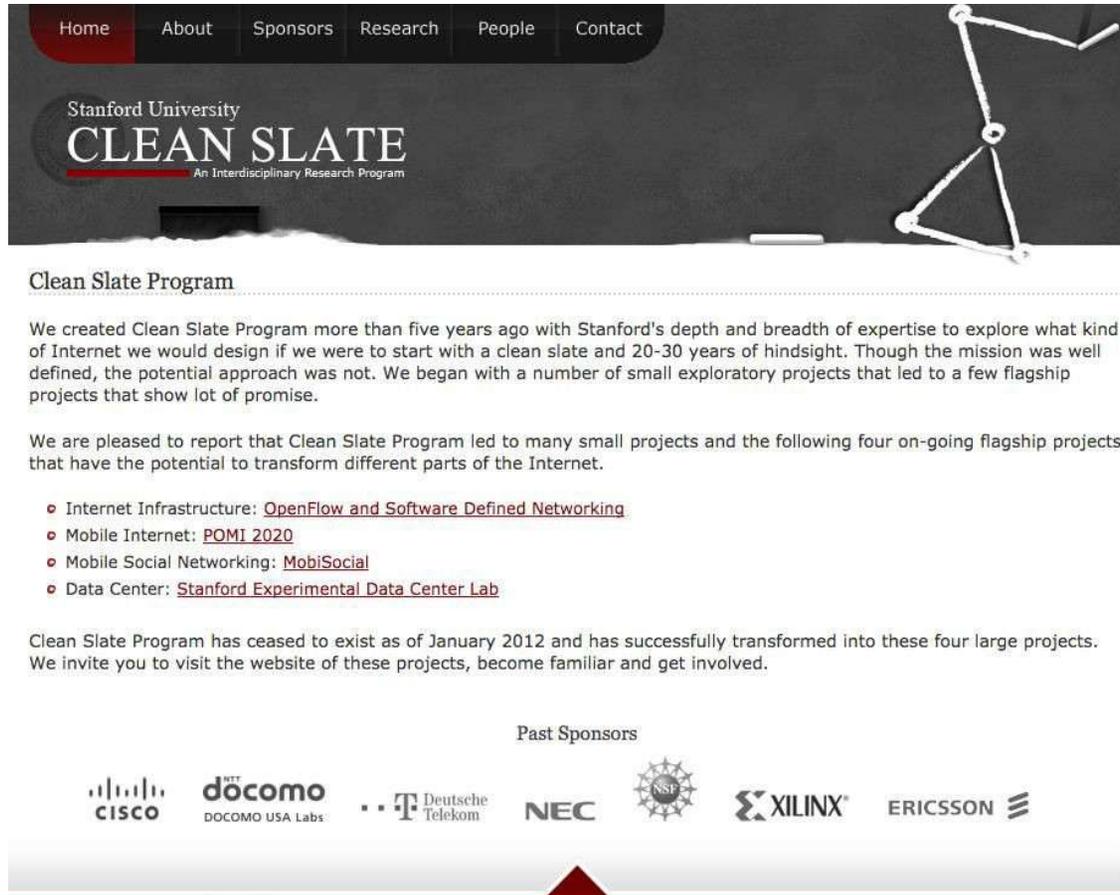


Applications

Network Requirements

Security

Where Did It All Begin?



Home About Sponsors Research People Contact

Stanford University
CLEAN SLATE
An Interdisciplinary Research Program

Clean Slate Program

We created Clean Slate Program more than five years ago with Stanford's depth and breadth of expertise to explore what kind of Internet we would design if we were to start with a clean slate and 20-30 years of hindsight. Though the mission was well defined, the potential approach was not. We began with a number of small exploratory projects that led to a few flagship projects that show lot of promise.

We are pleased to report that Clean Slate Program led to many small projects and the following four on-going flagship projects that have the potential to transform different parts of the Internet.

- Internet Infrastructure: [OpenFlow and Software Defined Networking](#)
- Mobile Internet: [POMI 2020](#)
- Mobile Social Networking: [MobiSocial](#)
- Data Center: [Stanford Experimental Data Center Lab](#)

Clean Slate Program has ceased to exist as of January 2012 and has successfully transformed into these four large projects. We invite you to visit the website of these projects, become familiar and get involved.

Past Sponsors



OPEN NETWORKING
FOUNDATION

Dynamically apply state and control to network infrastructure using globally aware **software controls**

Decoupled control and **data planes** and **centralised** intelligence

Abstraction of underlying **network infrastructure**

SDN is...

...a new approach at network transformation

...impacting the networking industry

...providing new methods to interact with equipment/services via **controllers, APIs**

...enabling high-scale, rapid network and service provisioning/management

...generating a LOT of attention

...providing a catalyst for traditional Route/Switch engineers to branch-out

SDN is not...

...an easy button

...an end-state

...narrowly defined

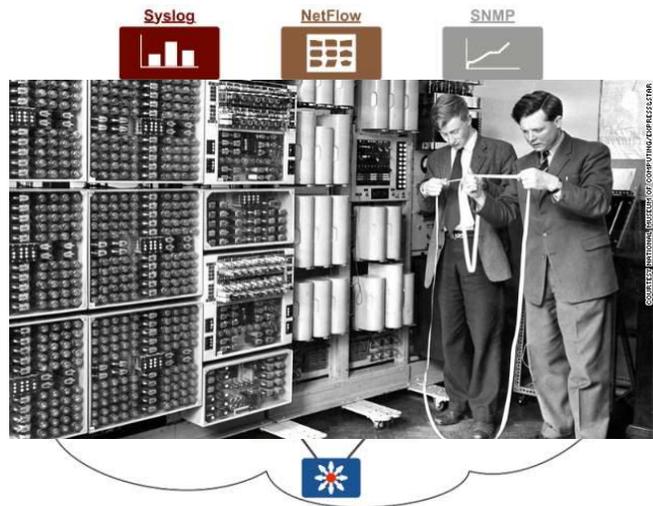
...meaning the death of network engineers

...a mandate for all network engineers to become programmers

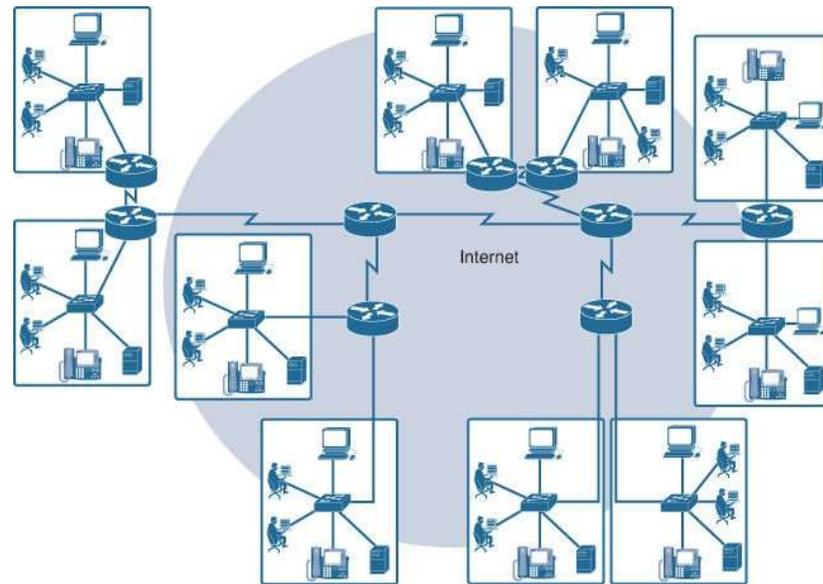
...a new attempt at network evolution

Evolution of Network Configuration

1990s



Today



What is the most used automation tool for network engineers?

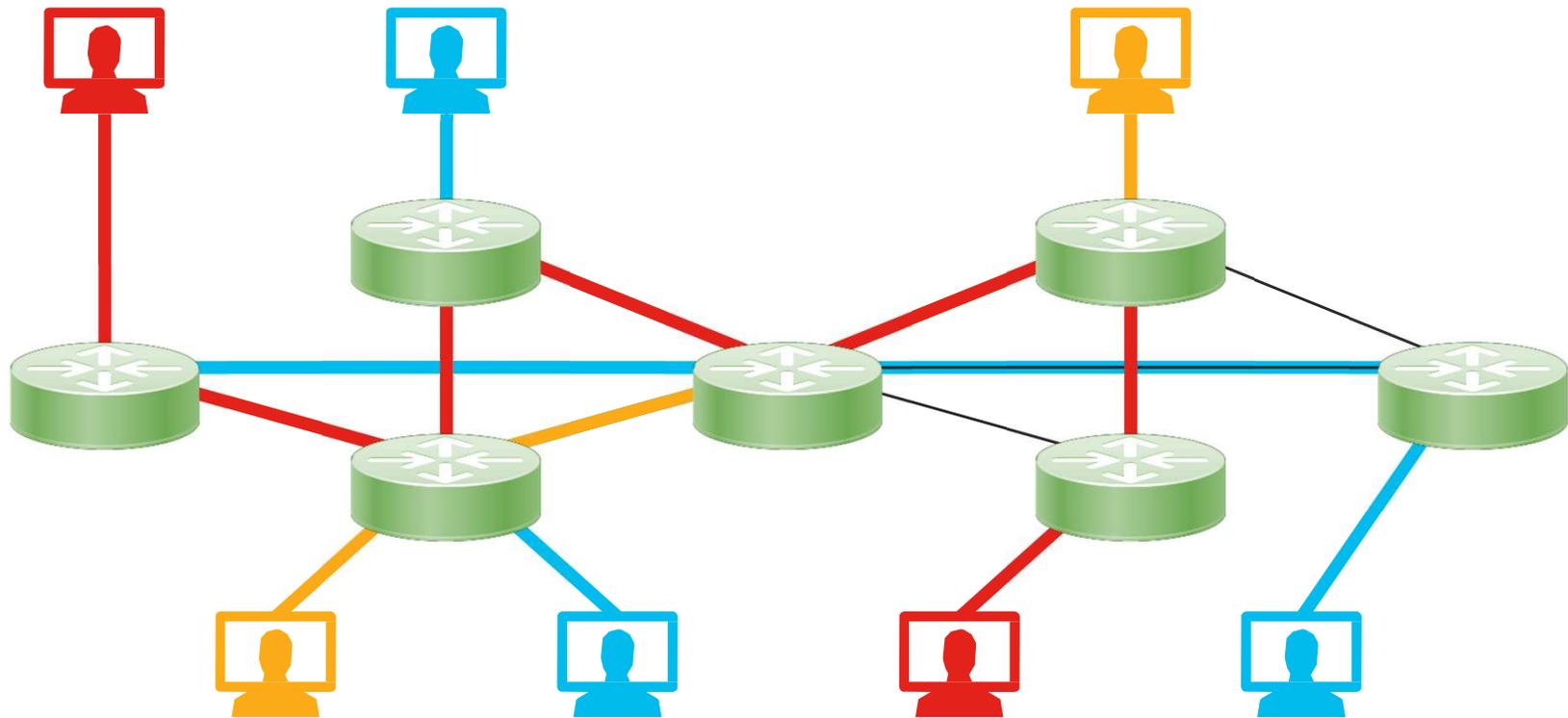
Notepad

```
Untitled - Notepad
File Edit Format View Help
version 16.3
no service pad
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-recovery
no platform punt-keepalive disable-kernel-core
!
hostname switch1
!
!
interface GigabitEthernet0/0
vrf forwarding Mgmt-vrf
ip address 10.1.1.11 255.255.255.0
negotiation auto
!
```

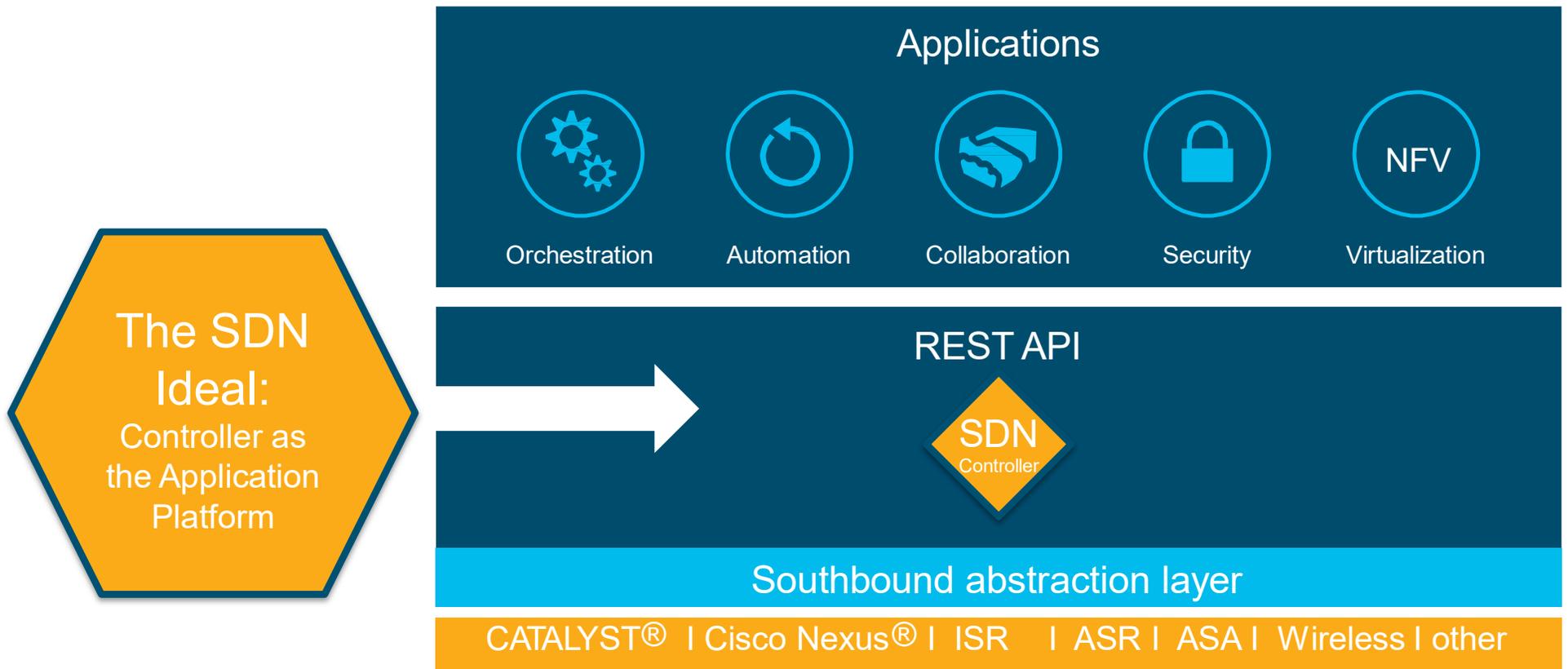
```
Untitled - Notepad
File Edit Format View Help
version 16.3
no service pad
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-recovery
no platform punt-keepalive disable-kernel-core
!
hostname switch2
!
!
interface GigabitEthernet0/0
vrf forwarding Mgmt-vrf
ip address 10.1.1.12 255.255.255.0
negotiation auto
!
```

```
Untitled - Notepad
File Edit Format View Help
version 16.3
no service pad
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-recovery
no platform punt-keepalive disable-kernel-core
!
hostname switch6
!
!
interface GigabitEthernet0/0
vrf forwarding Mgmt-vrf
ip address 10.1.1.16 255.255.255.0
negotiation auto
!
```

Need More than Centralised Management



Network-Wide Abstractions Simplify the Network



Automation Makes Life Easier & Prettier!

You 1:35 pm

dnac show networkDevices

dnac 1:35 pm

```
1 | DNAC>asr1001-x.abc.inc(ASR1001-X: ip 10.10.22.74: serial FXS1932Q1SE: version 16.6.1: statu
2 | cat_9k_1.abc.inc(C9300-24UX: ip 10.10.22.66: serial FCW2136L0AK: version 16.6.1: status Mar
3 | cat_9k_2.abc.inc(C9300-24UX: ip 10.10.22.70: serial FCW2140L039: version 16.6.1: status Mar
4 | cs3850.abc.inc(WS-C3850-48U-E: ip 10.10.22.69: serial FOC1833X0AR: version 16.6.2s: status
```

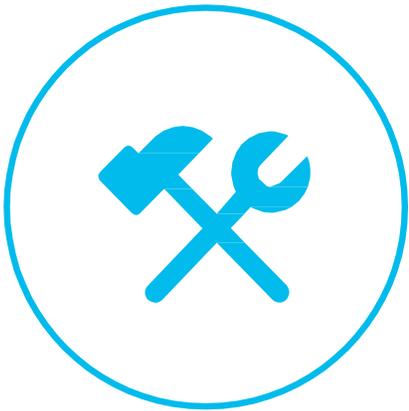
You Yesterday, 11:47 am

dnac find host [10.10.22.114](#)

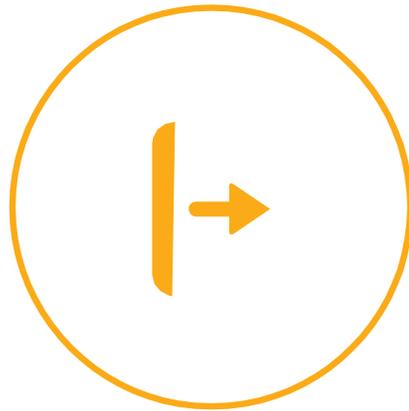
dnac Yesterday, 11:47 am

```
1 | DNAC>Found HOST 10.10.22.114:00:1e:13:a5:b9:40(wired) on ->
2 | DEVICE:10.10.22.70|TenGigabitEthernet1/0/24 vlan:1
```

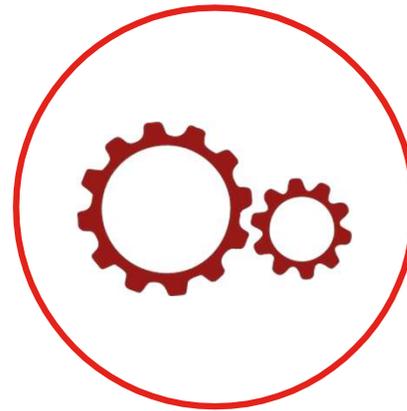
SDN Use Cases



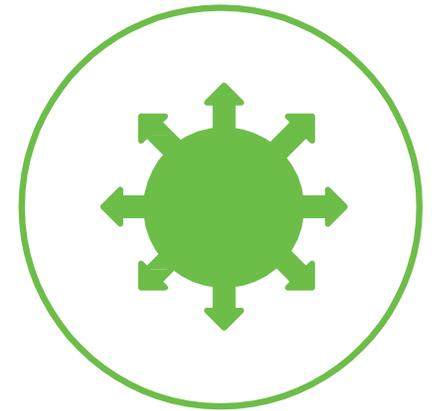
Network
Abstraction



Service
Deployment



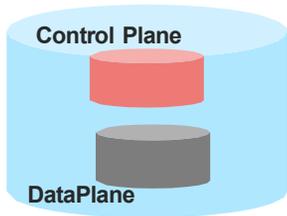
Traffic
Engineering



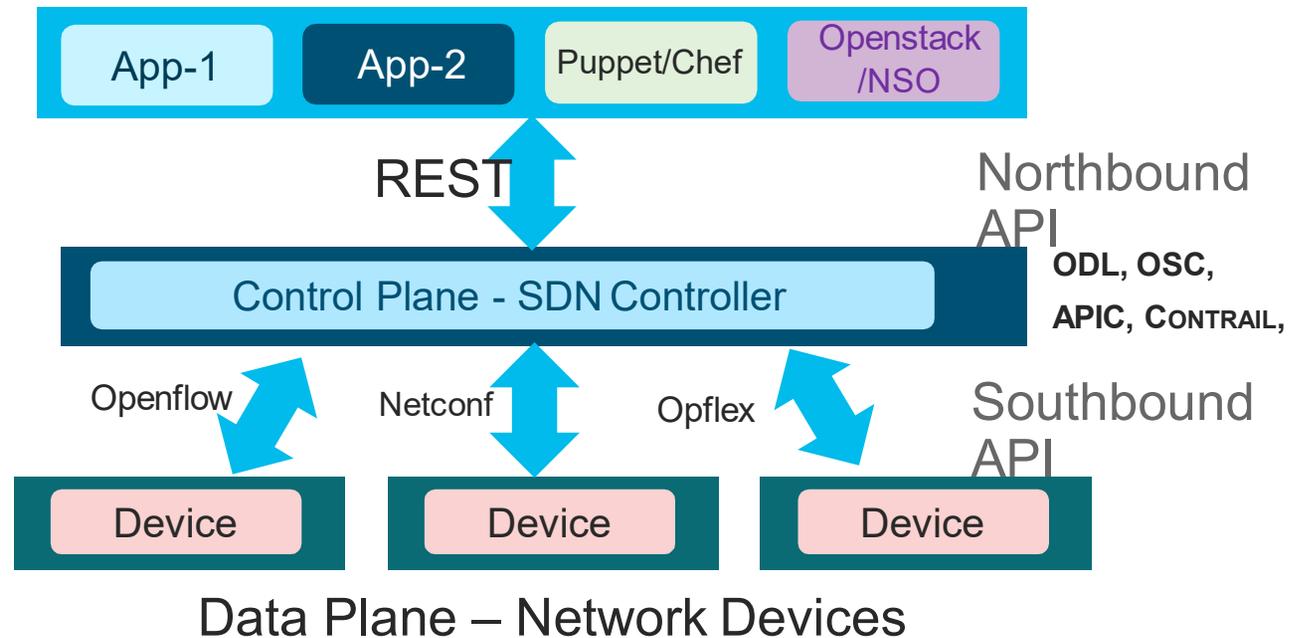
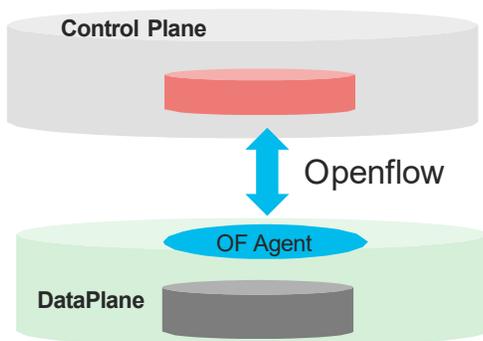
Network Function
Virtualisation

Software Defined Networking

SDN Definition (ONF): The physical separation of the network control plane from the forwarding plane, and where a control plane controls several devices.

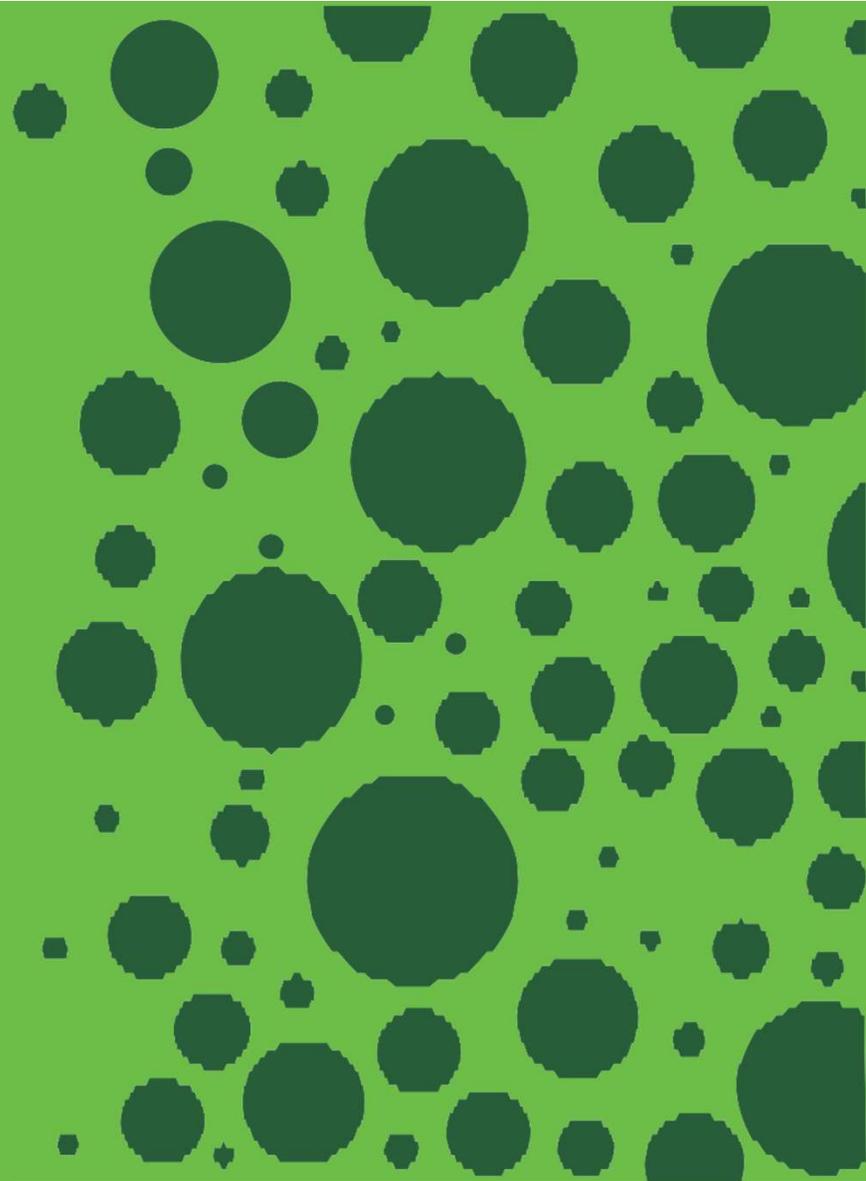


In SDN, Not All Processing Happens Inside Device



Data Plane – Network Devices

API



CLI to API

- Familiar Manual, CLI-driven, device-by-device approach is inefficient
- Increased need for programmatic interfaces which allow faster and automated execution of processes and workflows with reduced errors
- Need for programmatically readable data structures

```
davisnet-3560CG>sh int
Vlan1 is up, line protocol is up
  Hardware is EtherSVI, address is 381c.1a72.1bc0 (bia 381c.1a72.1bc0)
  Internet address is 192.168.1.10/24
  MTU 1500 bytes, BW 1000000 kbit, DLY 10 usec,
    reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation ARPA, loopback not set
  Keepalive not supported
  ARP type: ARPA, ARP Timeout 04:00:00
  Last input 00:00:00, output 00:00:00, output hang never
  Last clearing of "show interface" counters never
  Input queue: 1/75/0/0 (size/max/drops/flushes); Total output drops: 0
  Queueing strategy: fifo
  Output queue: 0/40 (size/max)
  5 minute input rate 4000 bits/sec, 2 packets/sec
  5 minute output rate 1000 bits/sec, 1 packets/sec
  569740 packets input, 41609077 bytes, 0 no buffer
    Received 0 broadcasts (0 IP multicasts)
    0 runts, 0 giants, 0 throttles
    0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored
    16520 packets output, 1406421 bytes, 0 underruns
    0 output errors, 2 interface resets
    0 output buffer failures, 0 output buffers swapped out
GigabitEthernet0/1 is up, line protocol is up (connected)
  Hardware is Gigabit Ethernet, address is 381c.1a72.1b81 (bia 381c.1a72.1b81)
  Description: downlink to PapaBear2
  MTU 1500 bytes, BW 10000 kbit, DLY 1000 usec,
    reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation ARPA, loopback not set
  Keepalive set (10 sec)
  Full-duplex, 10Mb/s, media type is 10/100/1000BaseTX
  Input flow-control is off, output flow-control is unsupported
  ARP type: ARPA, ARP Timeout 04:00:00
  Last input never, output 00:00:00, output hang never
  Last clearing of "show interface" counters never
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 0
  Queueing strategy: fifo
```

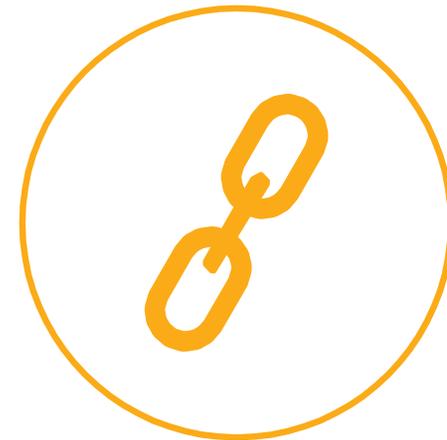


```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="4" xmlns="urn:ietf:params:netconf:base:1.0">
  <data>
    <xml-config-data> Building configuration...
      <Device-Configuration>
        <interface>
          <Param>GigabitEthernet0/0</Param>
          <ConfigIf-Configuration>
            <ip>
              <address><dhcp/></address>
            </ip>
            <duplex><auto/></duplex>
            <speed><auto/></speed>
          </ConfigIf-Configuration>
        </interface>
      <end></end>
    </Device-Configuration>
  </xml-config-data>
</data>
</rpc-reply>]]>>>
```

Application Programming Interface (API)

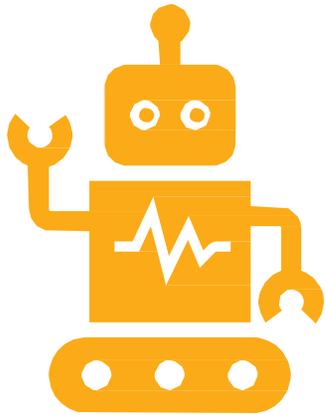


APIs expose pieces of an application to the outside world

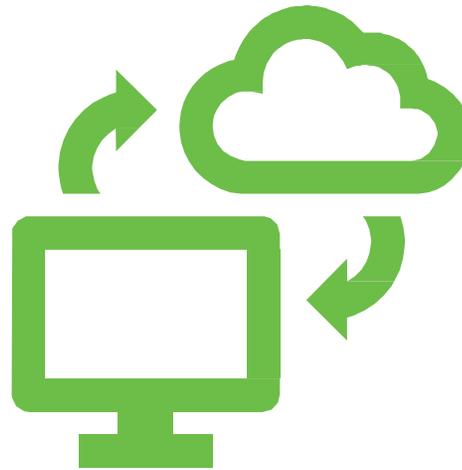


APIs are used to connect pieces of software together

Why APIs?



Automation



Integration



Innovation

RESTful APIs

Well Understood

Easy to Develop Against

Versioned

HTTP or HTTPS
GET, PUT, POST, and DELETE



Client



Server



Response in HTML, JSON / XML

JSON/XML

JSON



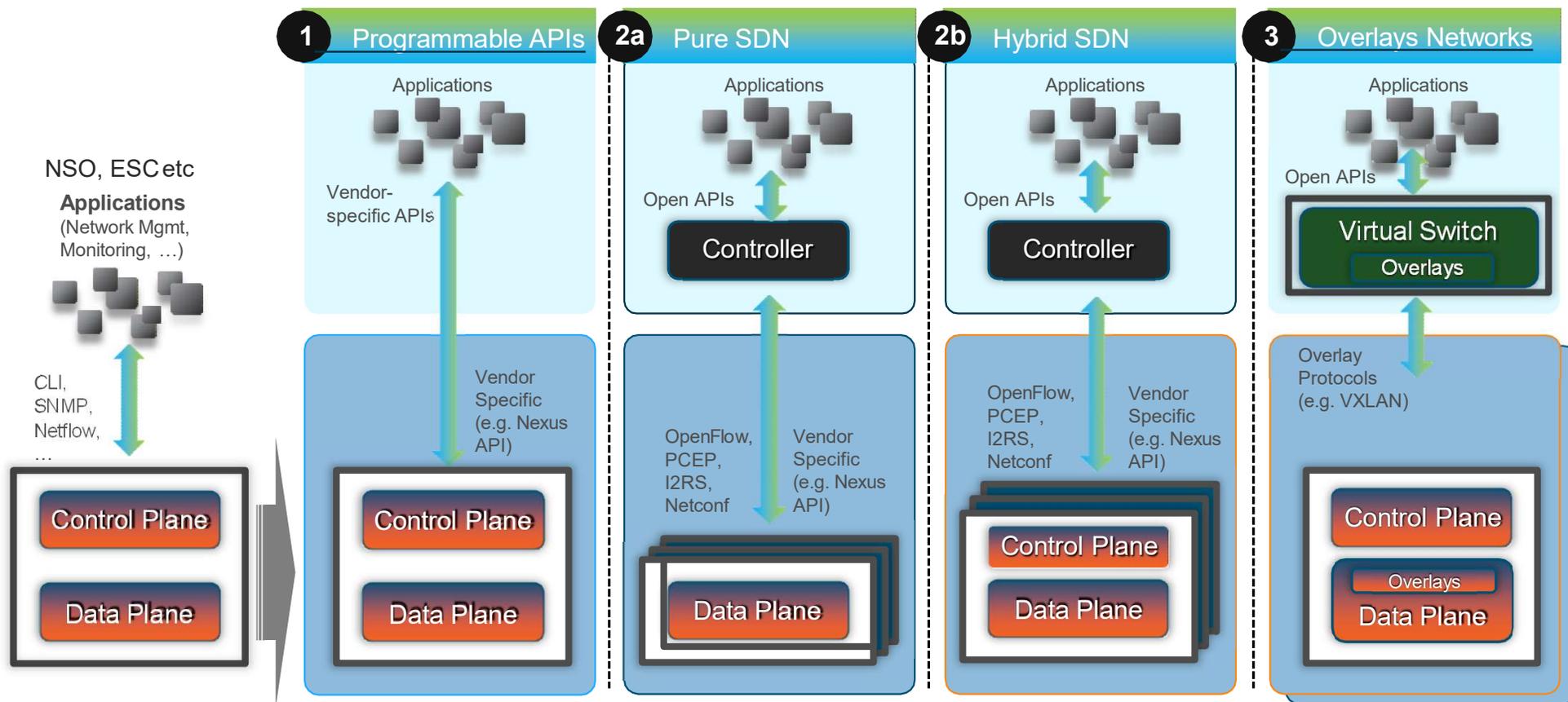
```
{  
  "ipv4Address": "172.16.11.11",  
  "ipv4Mask": "255.255.255.0",  
  "portName": "GigabitEthernet1",  
  "description": " TO_vSWITCH0",  
}
```

XML

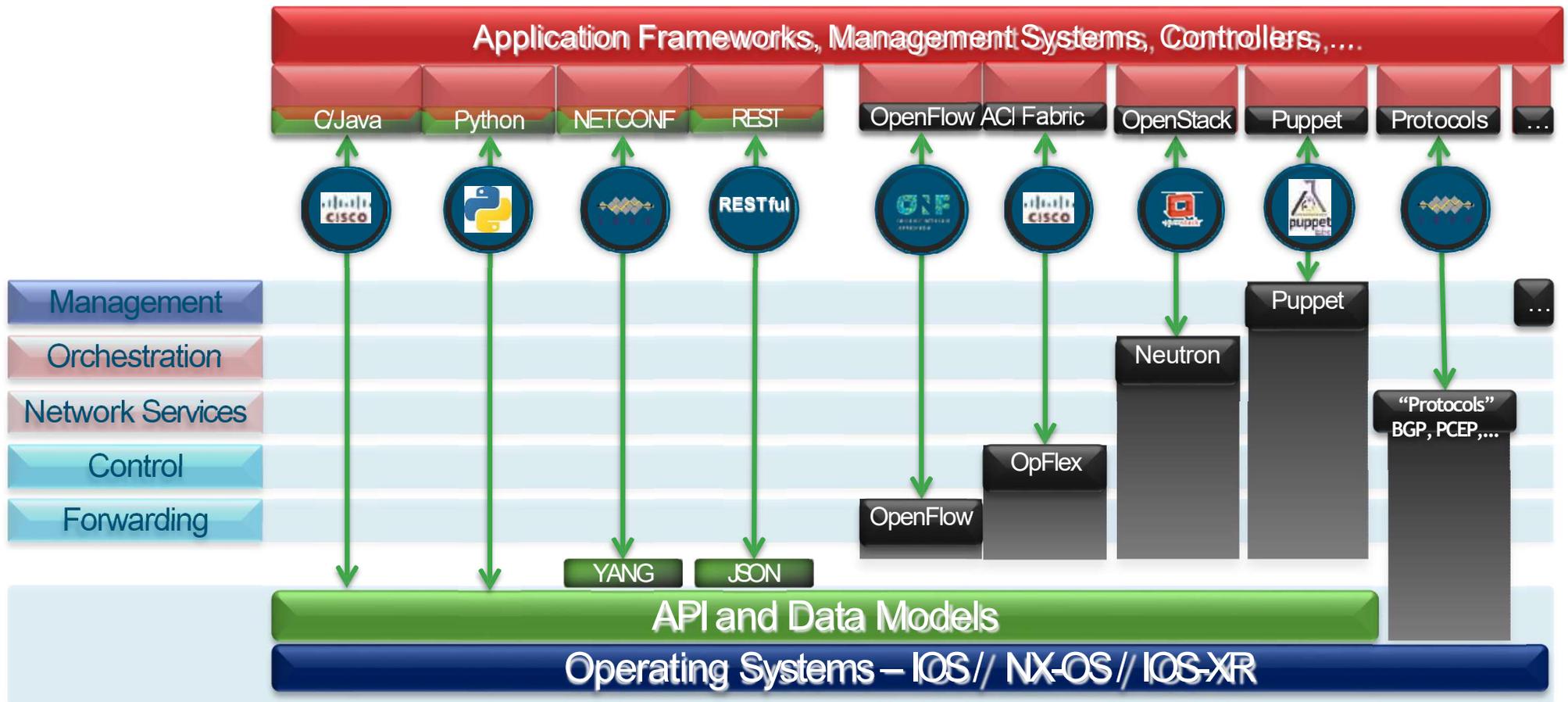


```
{  
  <interface>  
    <name>GigabitEthernet1</name>  
    <description>TO_vSWITCH0</description>  
    <address>  
      <ip>172.16.11.11</ip>  
      <netmask>255.255.255.0</netmask>  
    </address>  
  </interface>  
}
```

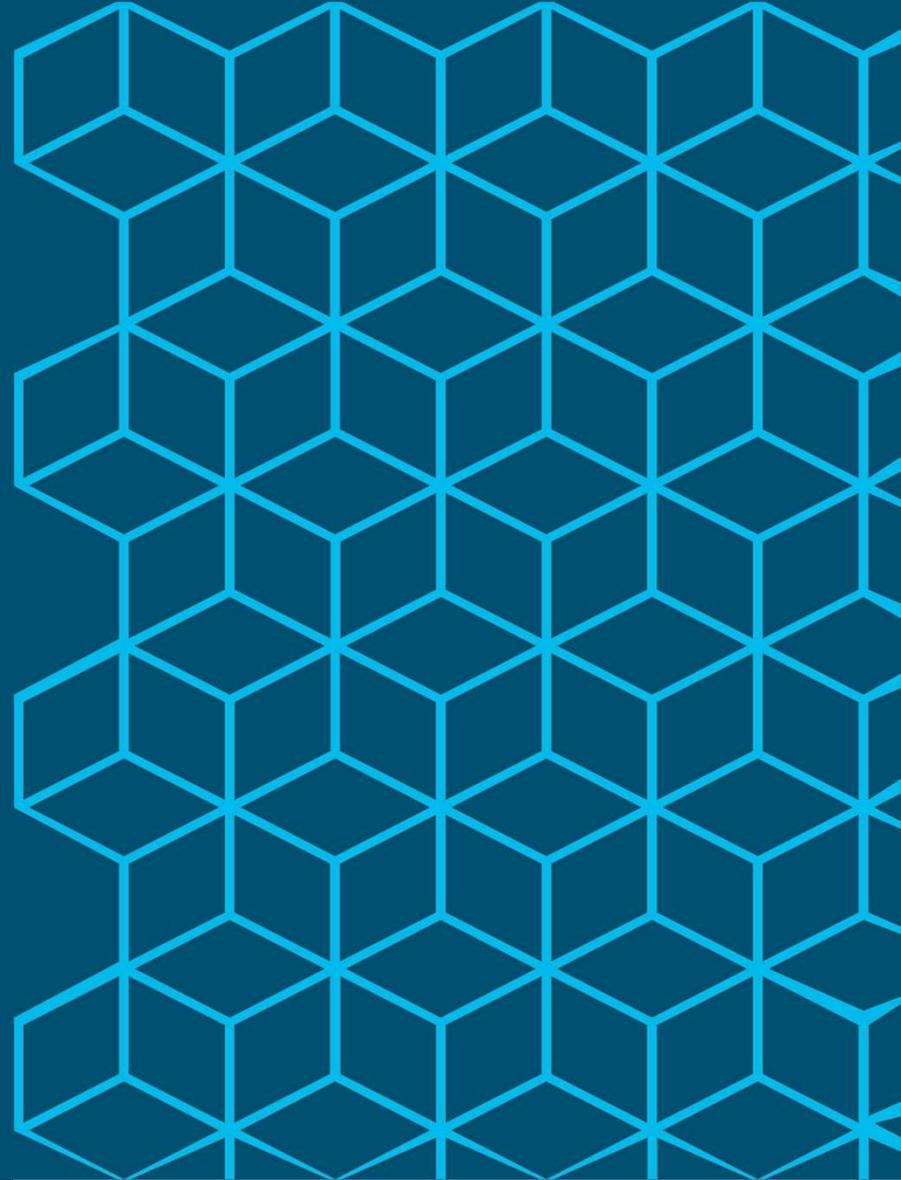
Network Programmability Options



Device Programmability Options – No Single Answer!



Key Concepts and Technologies



Good Old SNMP

SNMP works

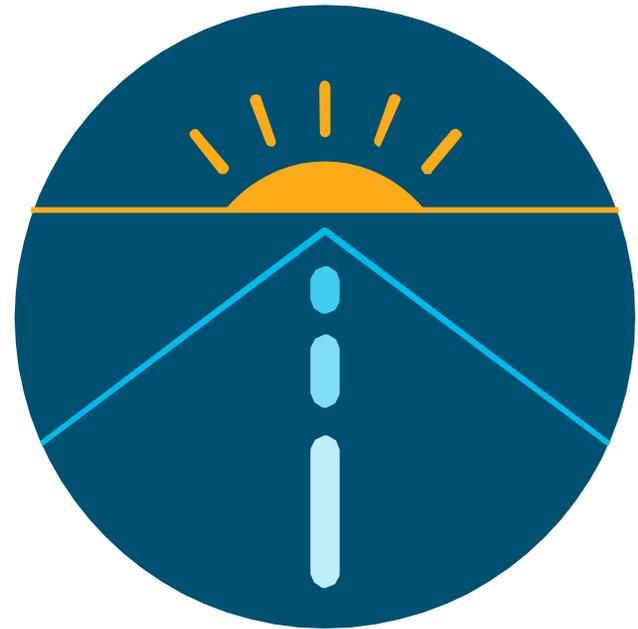
*“reasonably well for
device monitoring”*



- Typical config: SNMPv2 read-only community strings
- Typical usage: interface statistics queries and traps
- Empirical Observation: SNMP is not used for configuration
 - Lack of Writeable MIBs
 - Security Concerns
 - Difficult to Replay/Rollback
 - Special Applications

What is needed?

- A programmatic interface for device configuration
- Separation of Configuration and State Data
- Ability to configure "services" NOT "devices"
- Integrated error checking and recovery



Best Practices Coming Together

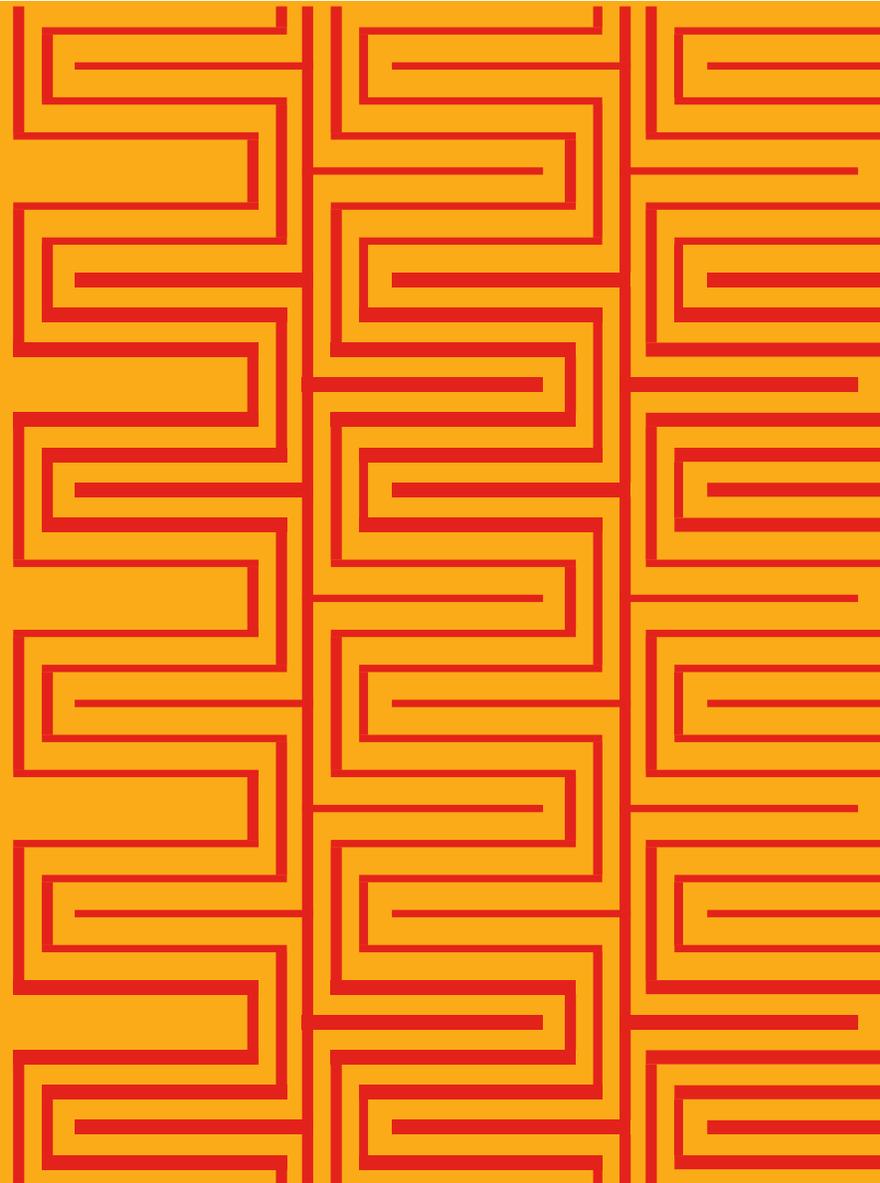
NETCONF, RESTCONF & YANG

Operations Requirements

SNMP Experience

CLI Best Practices

RESTCONF, NETCONF, YANG



Data Model vs. Network Management Protocol

Data Model

- Defines the structure, syntax, and semantics of data
- Consistent and complete

Protocol

- Remote primitives to view and manipulate data
- Encoding of the data

TCP/IP Network Frame Format



- NETCONF
- RESTCONF

- YANG

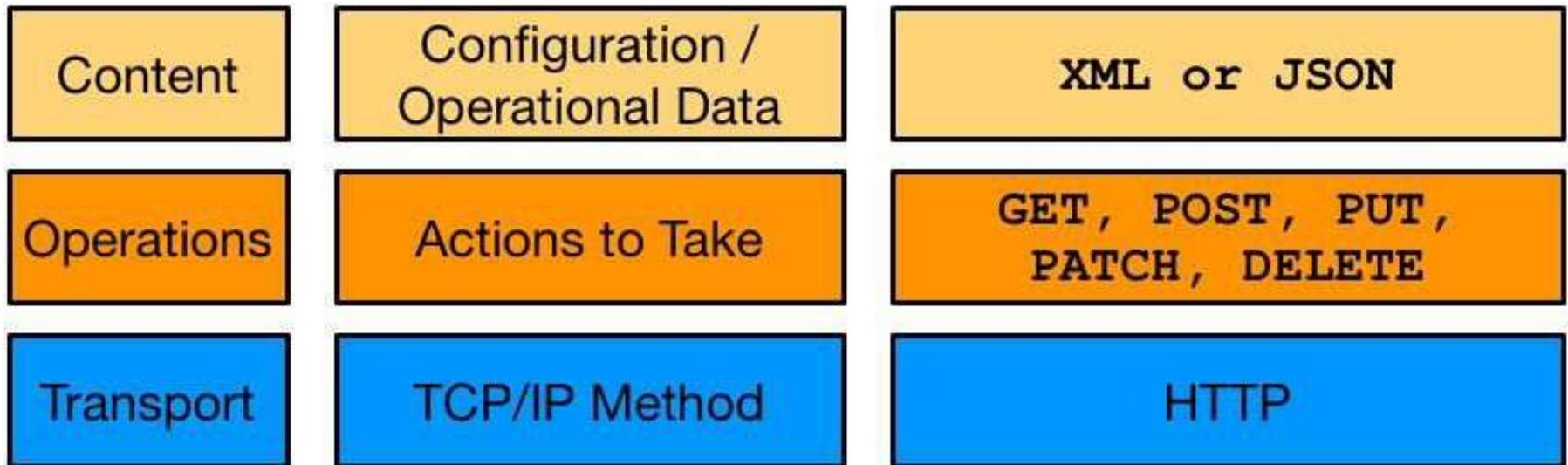
RESTCONF

“an HTTPS-based protocol that provides a programmatic interface for accessing data defined in YANG...”

- Uses HTTP(S) for transport
- Tightly coupled to the YANG data model definitions
- Provides JSON or XML data formats

RESTCONF Protocol Stack & Transport

RESTCONF Protocol Stack



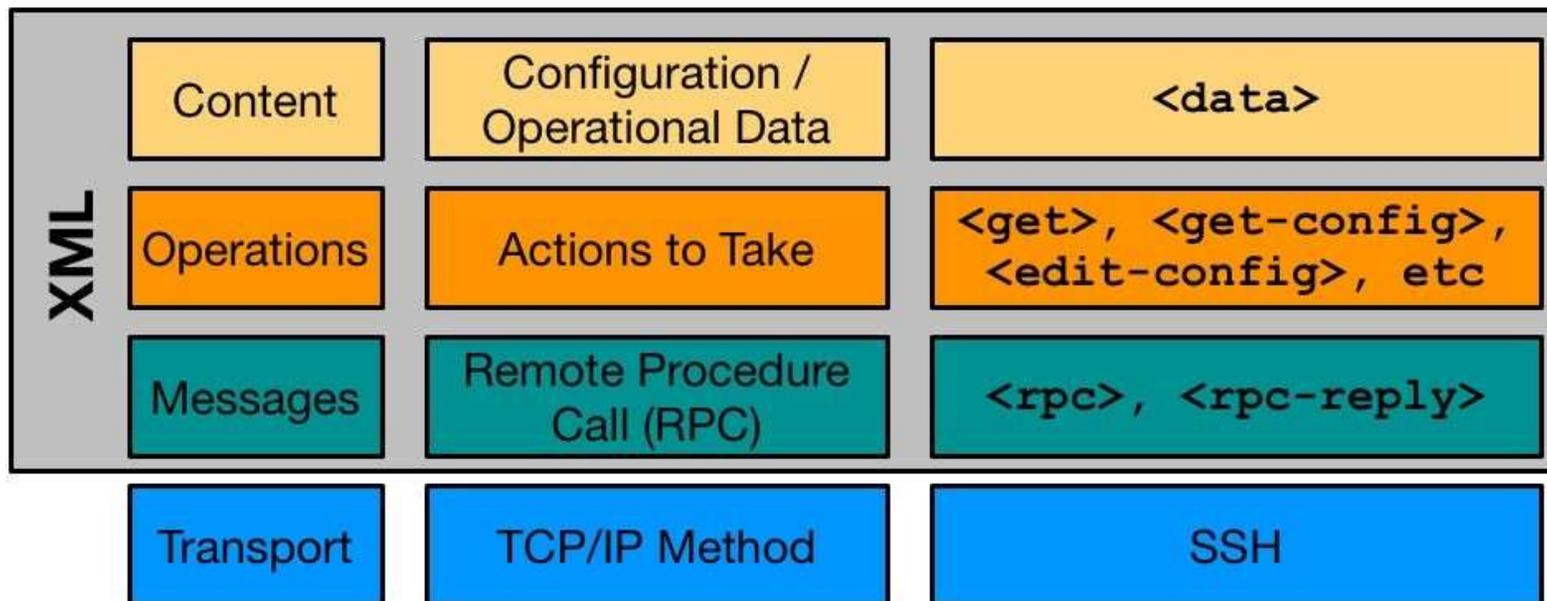
NETCONF

“The Network Configuration Protocol (NETCONF) is a SSH-based network management protocol developed and standardised by the IETF...”

- Uses SSH for transport
- Tightly coupled to XML
- Its operations are realised on top of a simple remote procedure call (RPC) layer

NETCONF Protocol Stack

NETCONF Protocol Stack



YANG

```
DevNet$ pyang -f tree ietf-interfaces.yang
```

```
module: ietf-interfaces
```

```
  +--rw interfaces
```

```
    | +--rw interface* [name]
```

```
    |   +--rw name          string
```

```
    |   +--rw description?  string
```

```
    |   +--rw type          identityref
```

```
    |   +--rw enabled?      boolean
```

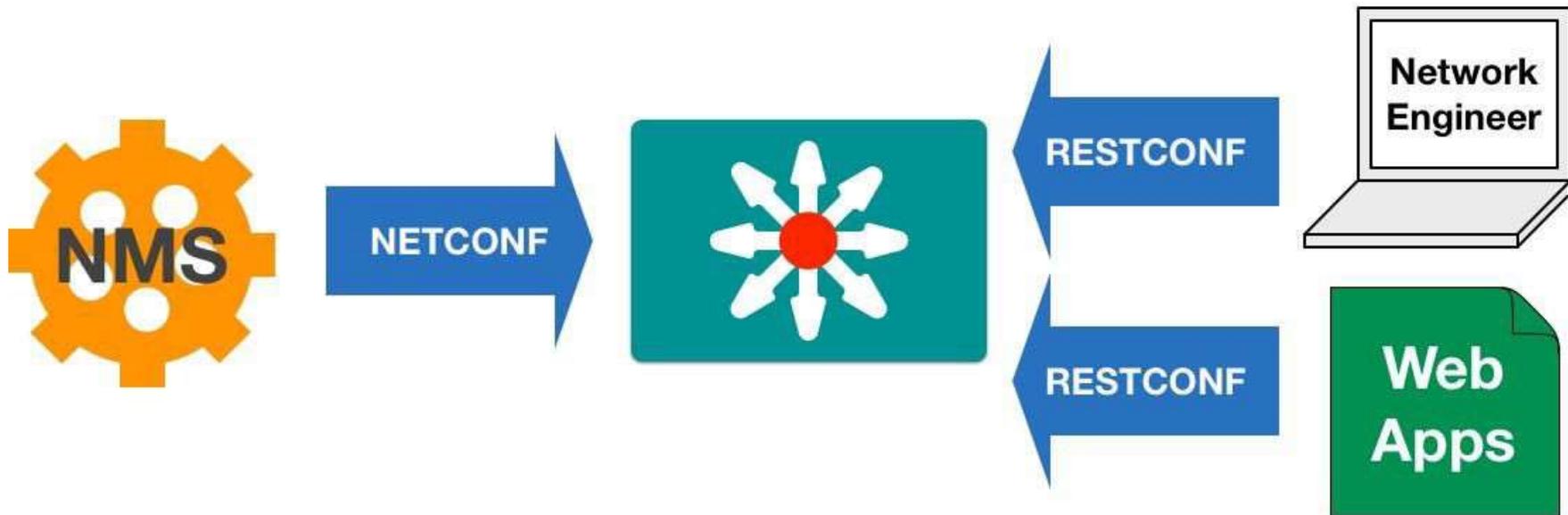
```
    |   +--rw link-up-down-trap-enable? enumeration {if-mib}?
```

Protocol vs Data Model Considerations

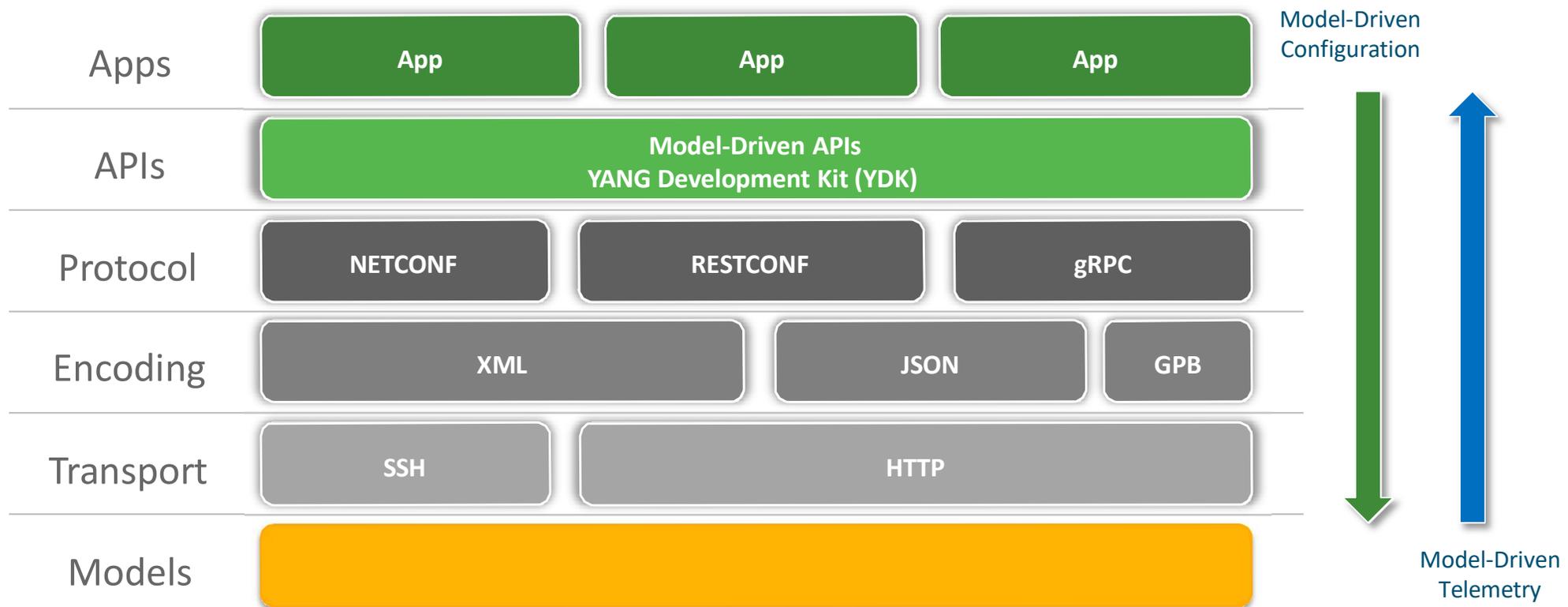
- NETCONF was described and standardised as a transport protocol **several years before YANG** was defined.
- In the time before YANG models existed, NETCONF leveraged **vendor specific data models**.
- Going forward YANG based data models are the primary network data model format used by vendors, but you **may still find devices that offer non-YANG data**
- YANG data models can be leveraged without NETCONF. **RESTCONF** is a great example of this.

RESTCONF And NETCONF?

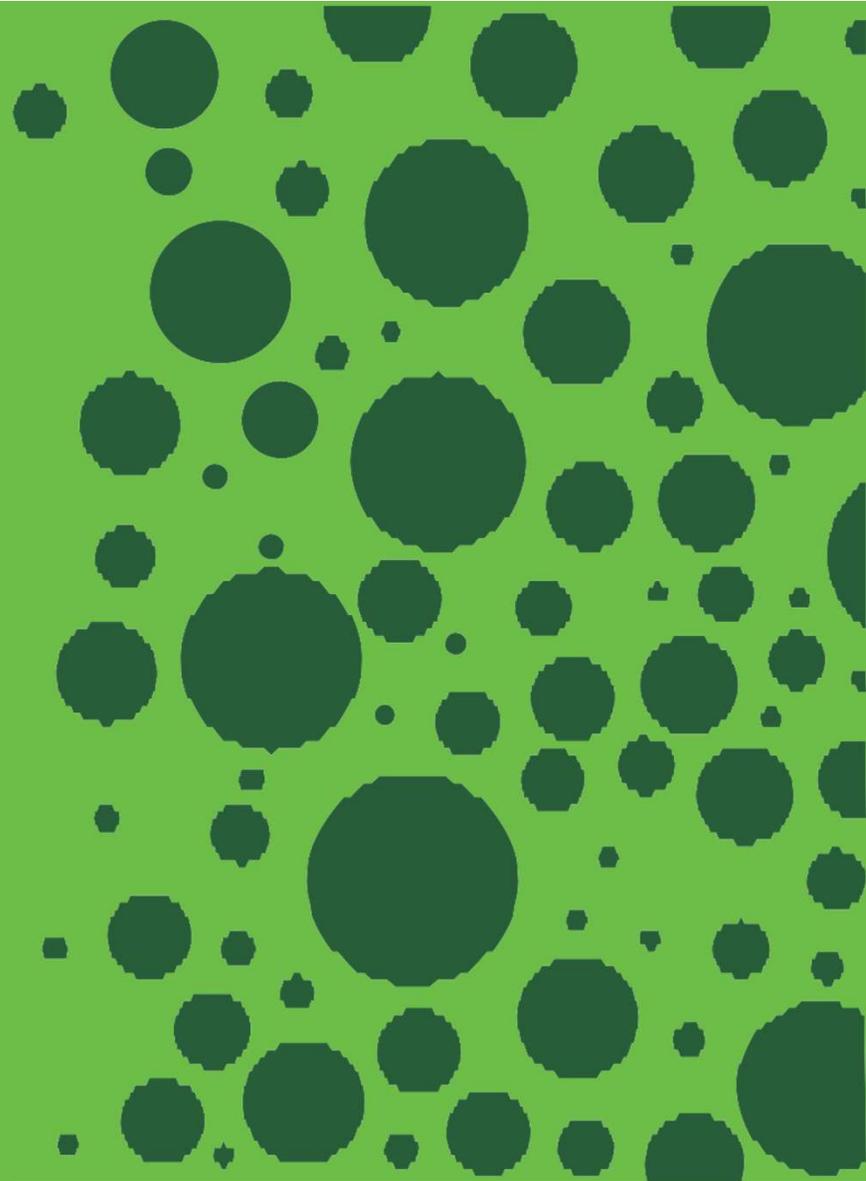
Standard Network Management



Model-Driven Programmability Stack



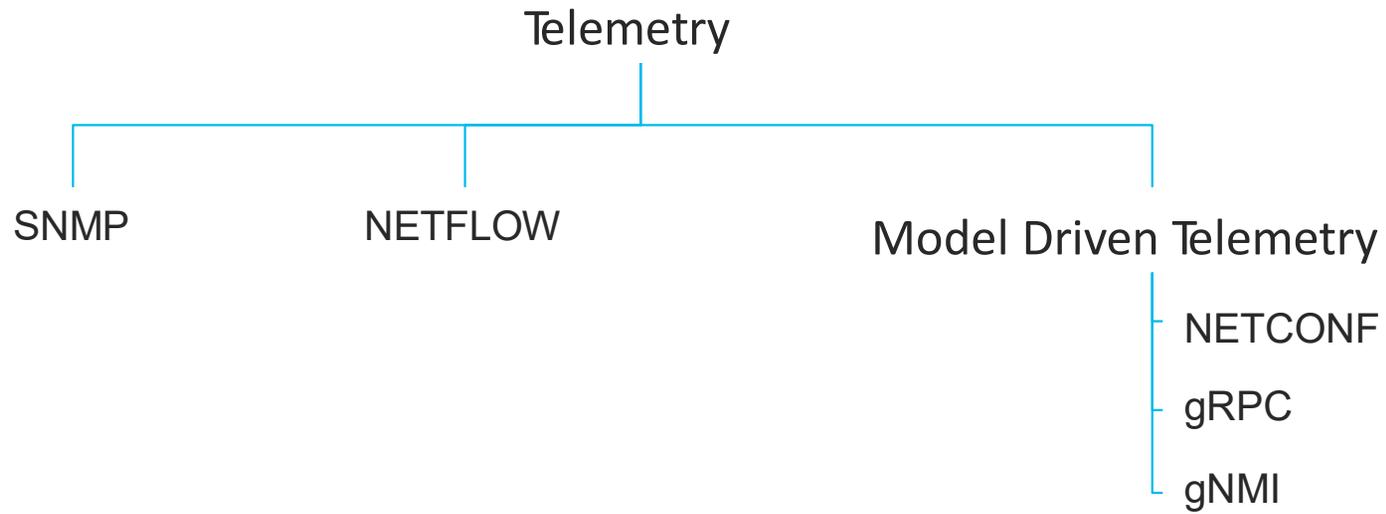
Model Driven Telemetry



***Network Telemetry is a new
revolutionary feature!!***

Is it really?..

Telemetry Hierarchy



Why Model Driven Telemetry is Important

SNMP

UDP transport

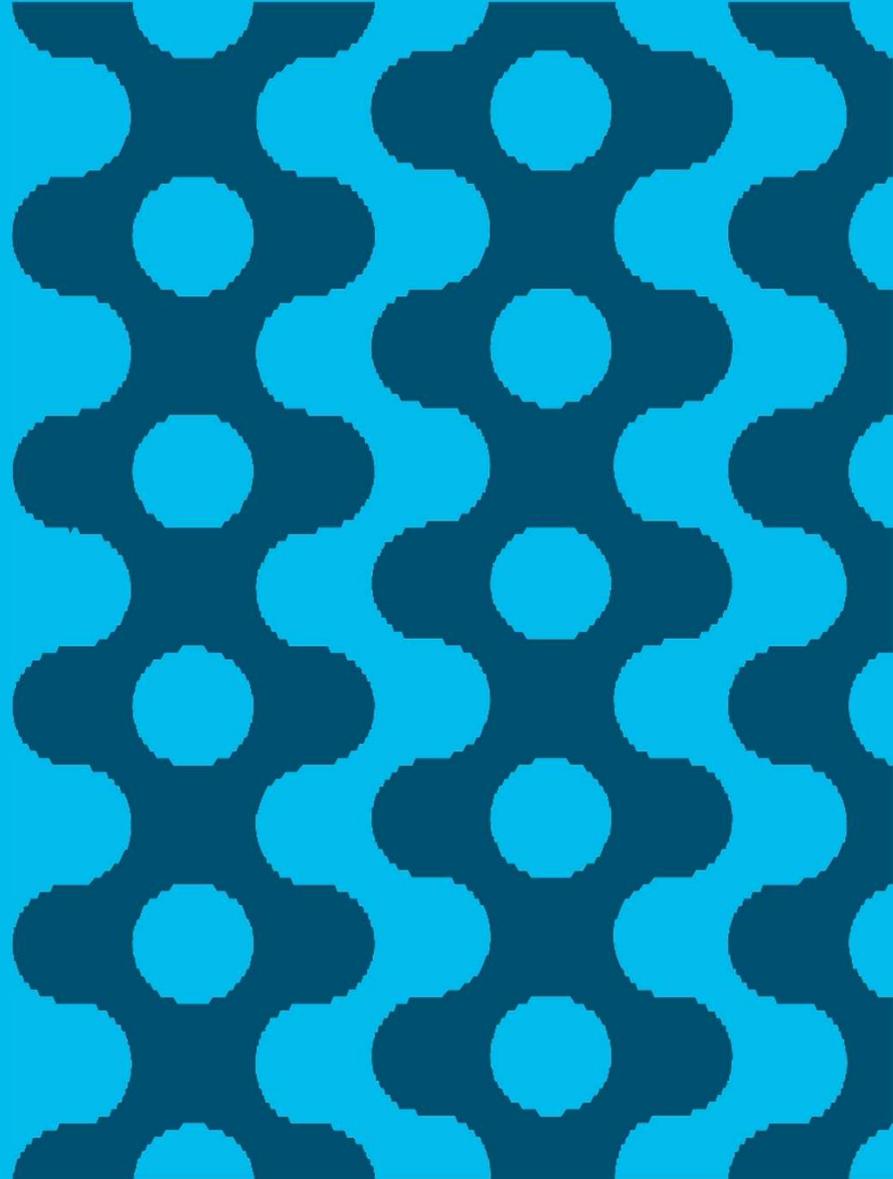
potential to
miss events

**Model Driven
Telemetry**

TCP transport

automate responses to
all events

Network Functions Virtualisation (NFV)



What is Enterprise NFV?

Enterprise Network Functions Virtualisation

Centralised orchestration and management
SDN Applications

Consistent, trusted network services across all the platforms
Virtual Network Functions (VNFs)

Hardware and software independence
Virtualisation Layer

Freedom of choice
Hardware Platform

Customised Network Services for Your Network



IT Agility

1

Select your network functions

- vRouter
- vFirewall
- vWAN optimisation
- vWLAN controller
- Third-party services

2

Select your preferred platform

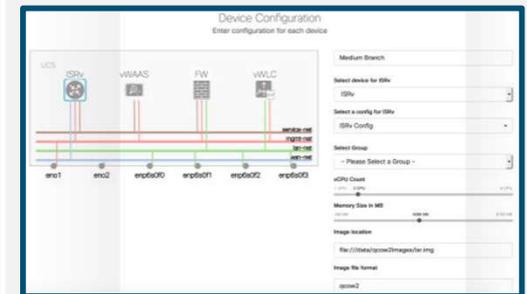
- Cisco® ISR, UCS® E-Series
- Cisco ENCS
- Cisco® UCS C-Series

3

Orchestrate and automate services



Automation Tools



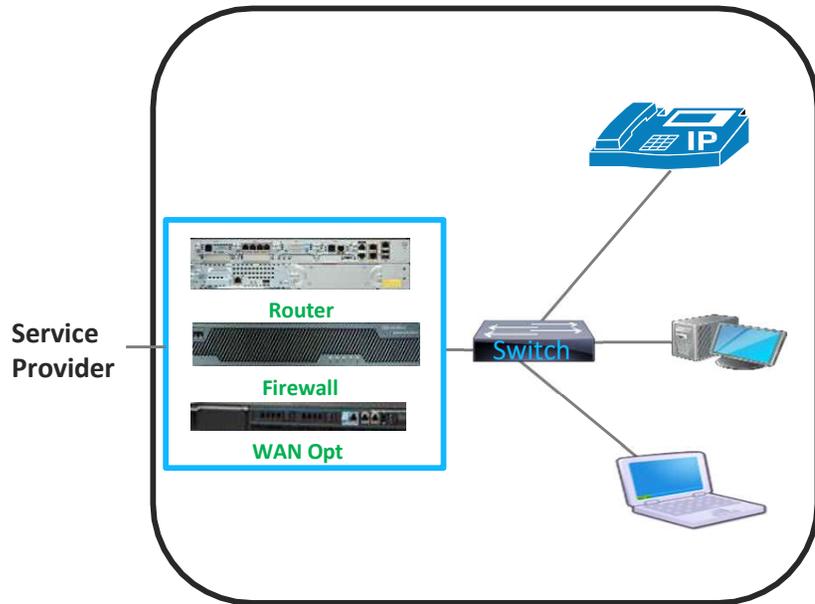
Elastic Services

Run on Any Platform

Deploy in Minutes

In other words...

Traditional Branch



Virtualized Branch

