The bridge to possible

Network Services Orchestration(NSO)

Sofia Athanasiou
Software Engineer – CX AI Engineering
06/12/2024 - GRNOG

Agenda

1 Motivation

2 NSO Overview

3 NSO Architecture

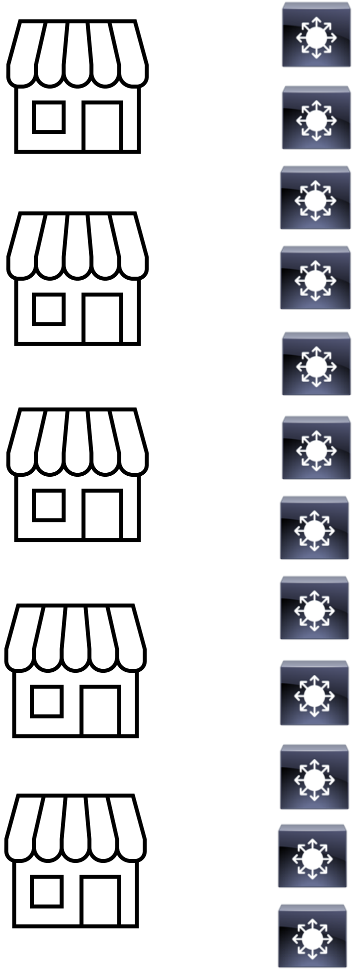
4 Use Cases

Automation

*“**Automation** is the technology by which a process or procedure is performed with minimal human assistance.”*

Wikipedia

The story



- Time consuming manual configuration
- Configuration errors and inconsistency
- Multivendor environment different CLI and APIs

Faster
Errorless
Scalable
Reliable

The automation drive

Complexity

- Increasing number of complex technologies
- Increasing scale
- Skills shortage

Quality

- Faster time to market
- Differentiation demand
- Security focus

Economics

- Reduce Opex
- Reduce human errors
- Account Role of technology investment with business outcomes

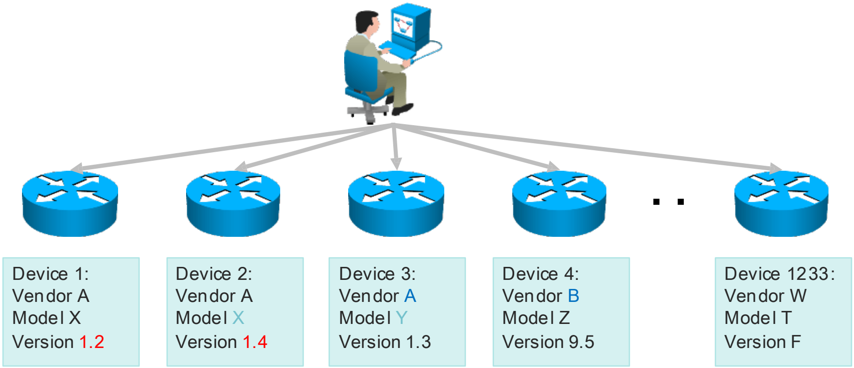
Tie Islands of Automation Together

End-to-End Automation Delivers Business Value

- Multiple Operating Systems and platforms (>10)
- Different Types of Automation Tools (>50)
- Scripts (>10,000)
- Manual Activity (>1,000)
- Orchestration software, cross-team collaboration

Network Management Challenges

- We are configuring **different** devices. - multi-vendor environment.
- There is no real **service** management
- There is no abstract **models** beyond device level
- Understand the capabilities and limitations of each device and device group
- Ensure consistency and reliability of configurations across all devices
- Backup and restore configurations.



Device 1:
Vendor A
Model X
Version 1.2

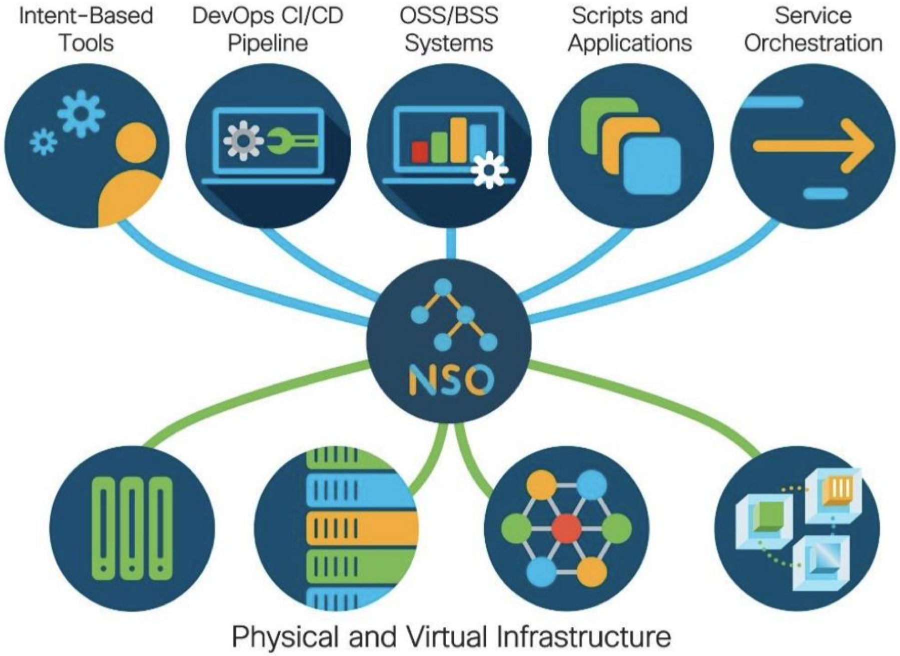
Device 2:
Vendor A
Model X
Version 1.4

Device 3:
Vendor A
Model Y
Version 1.3

Device 4:
Vendor B
Model Z
Version 9.5

Device 1233:
Vendor W
Model T
Version F

Cisco NSO is a bridge

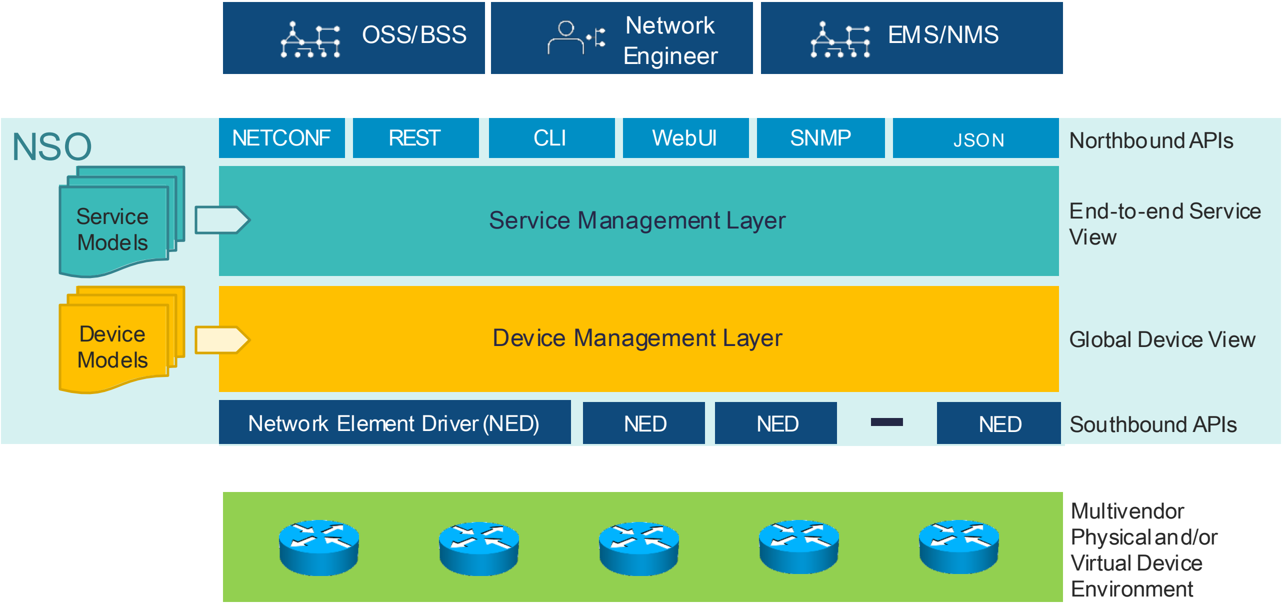


Between people that build services and ones that operate infrastructure

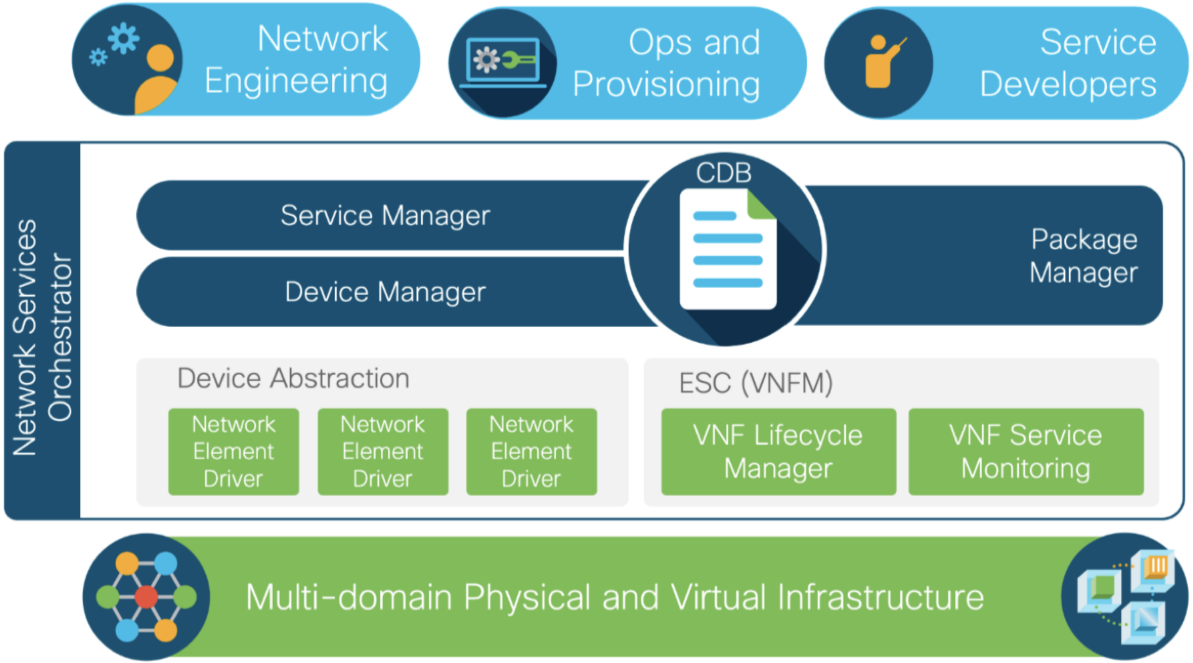
Across different domains and vendors

Over both physical and virtual infrastructure

NSO Network Abstraction



NSO Architecture



Model-driven, end-to-end service lifecycle and customer experience focused

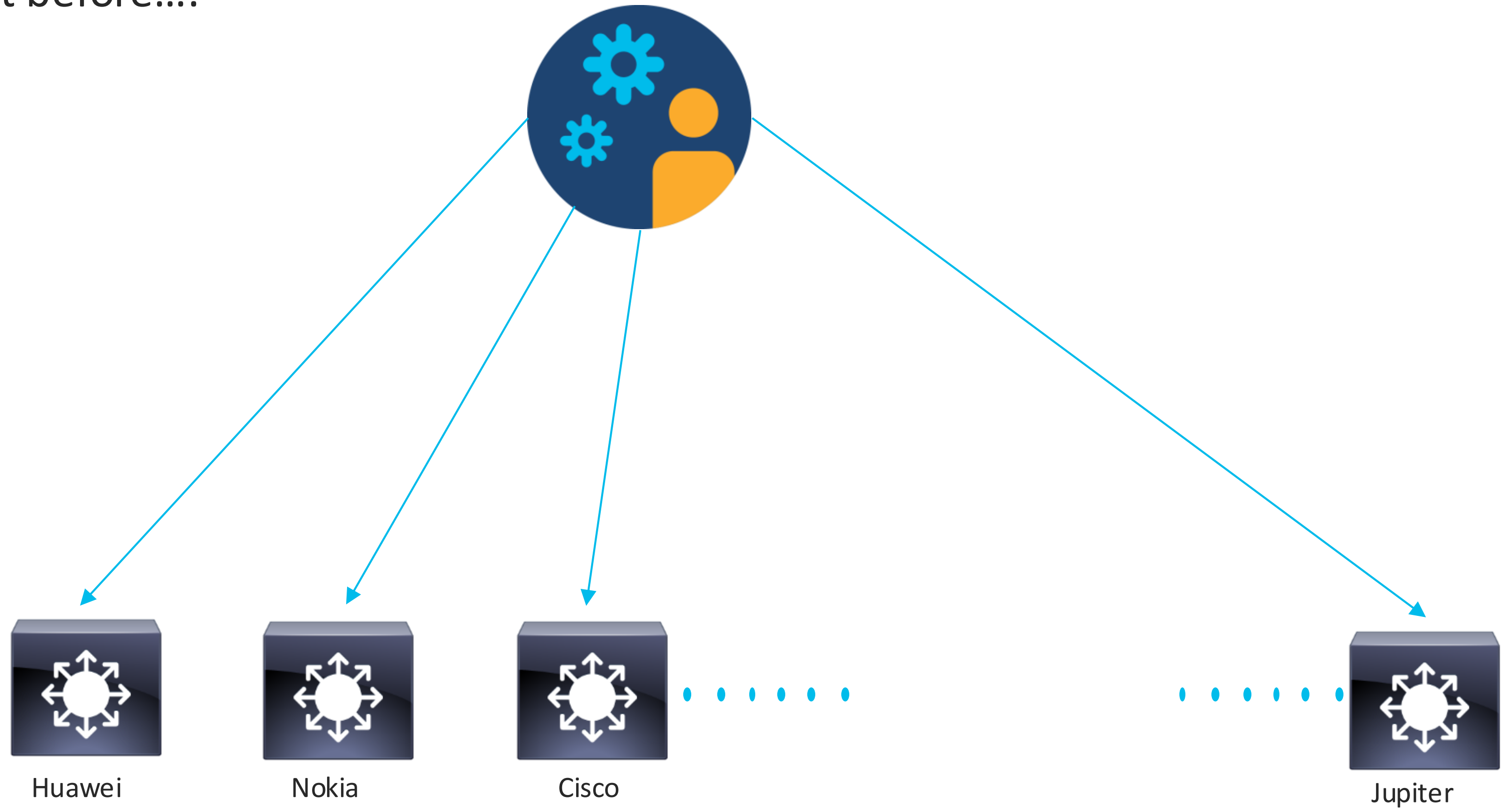
Seamless integration with northbound tooling

Loosely-coupled and modular architecture leveraging open APIs and standard protocols

Orchestration across multi-domain and multi-layer for network-wide, centralized policy and services

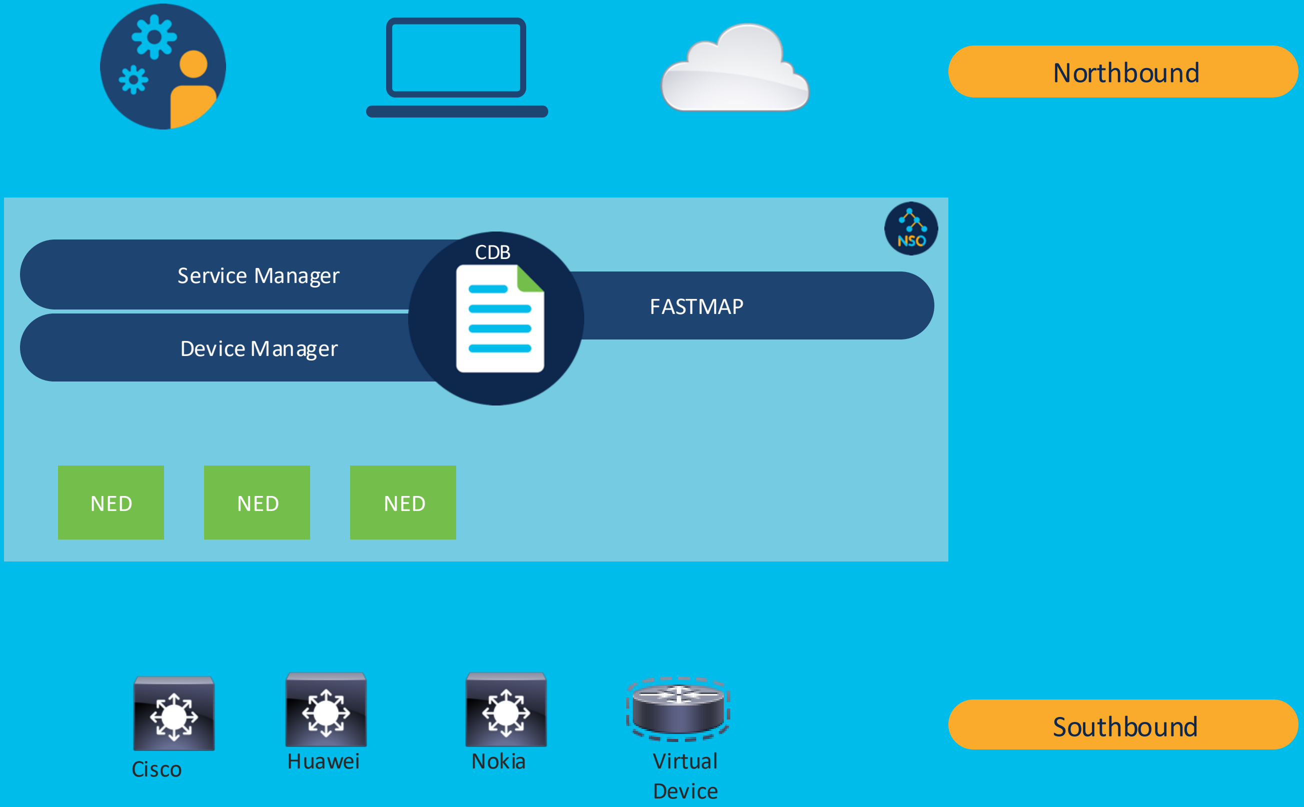
Network Services Orchestration (NSO)

How was it before….

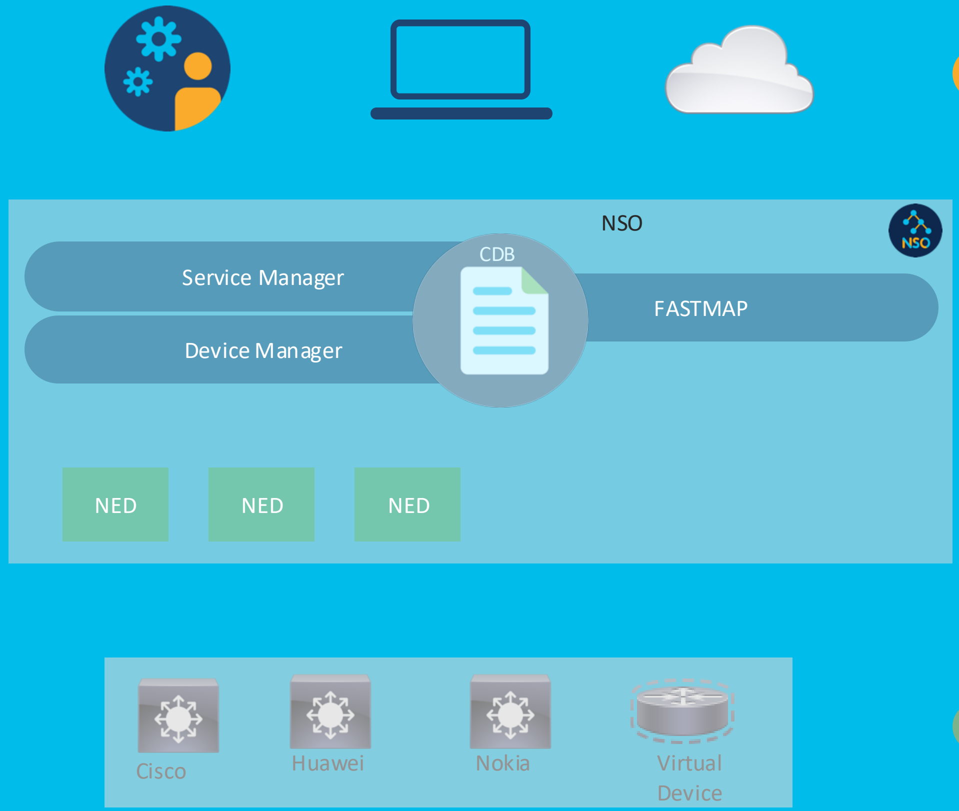


NSO Architecture

NSO High Level Architecture



NSO High Level Architecture

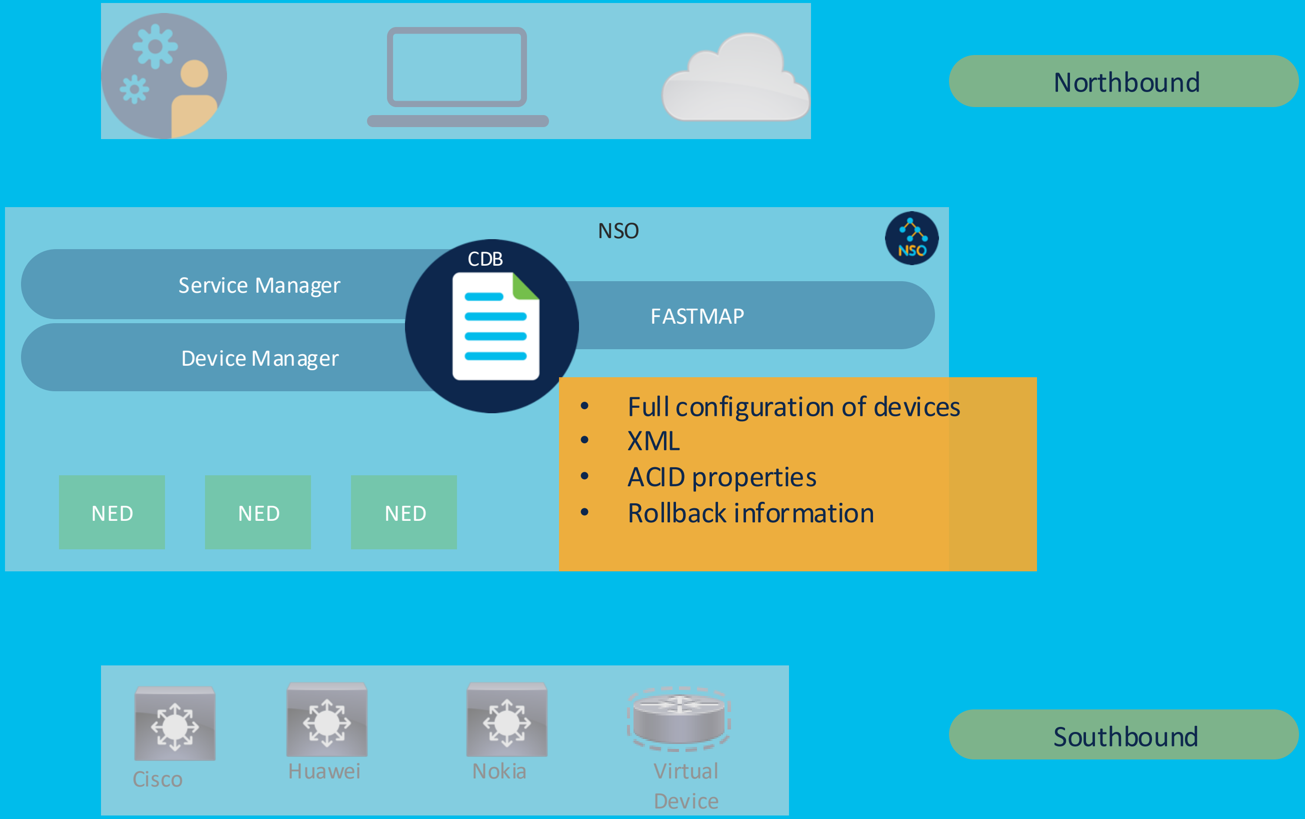


Northbound

- CLI
- REST
- RESTCONF
- ANSIBLE/JAVA
- GUI
- SNMP

Southbound

NSO High Level Architecture



Northbound

NSO

Service Manager

Device Manager

CDB

FASTMAP

- Full configuration of devices
- XML
- ACID properties
- Rollback information

NED

NED

NED

Cisco

Huawei

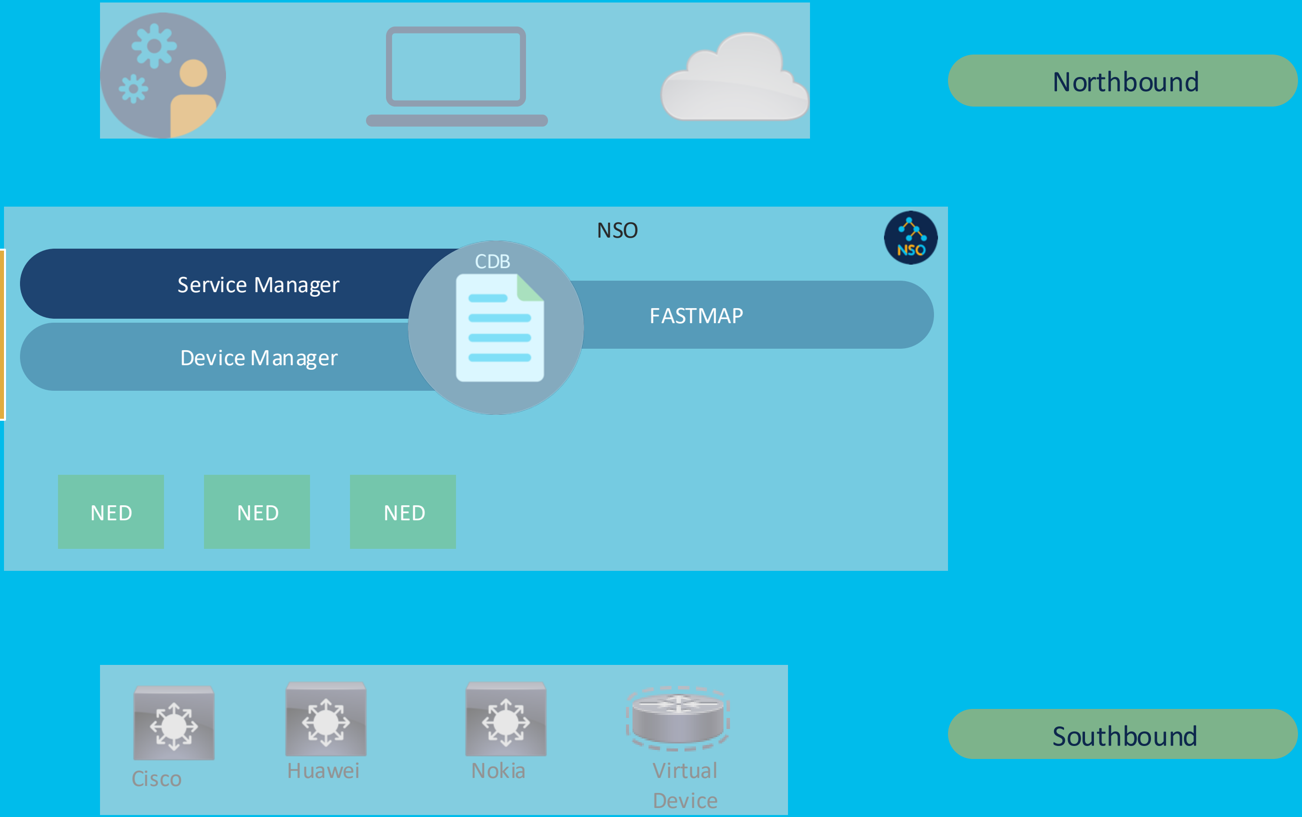
Nokia

Virtual Device

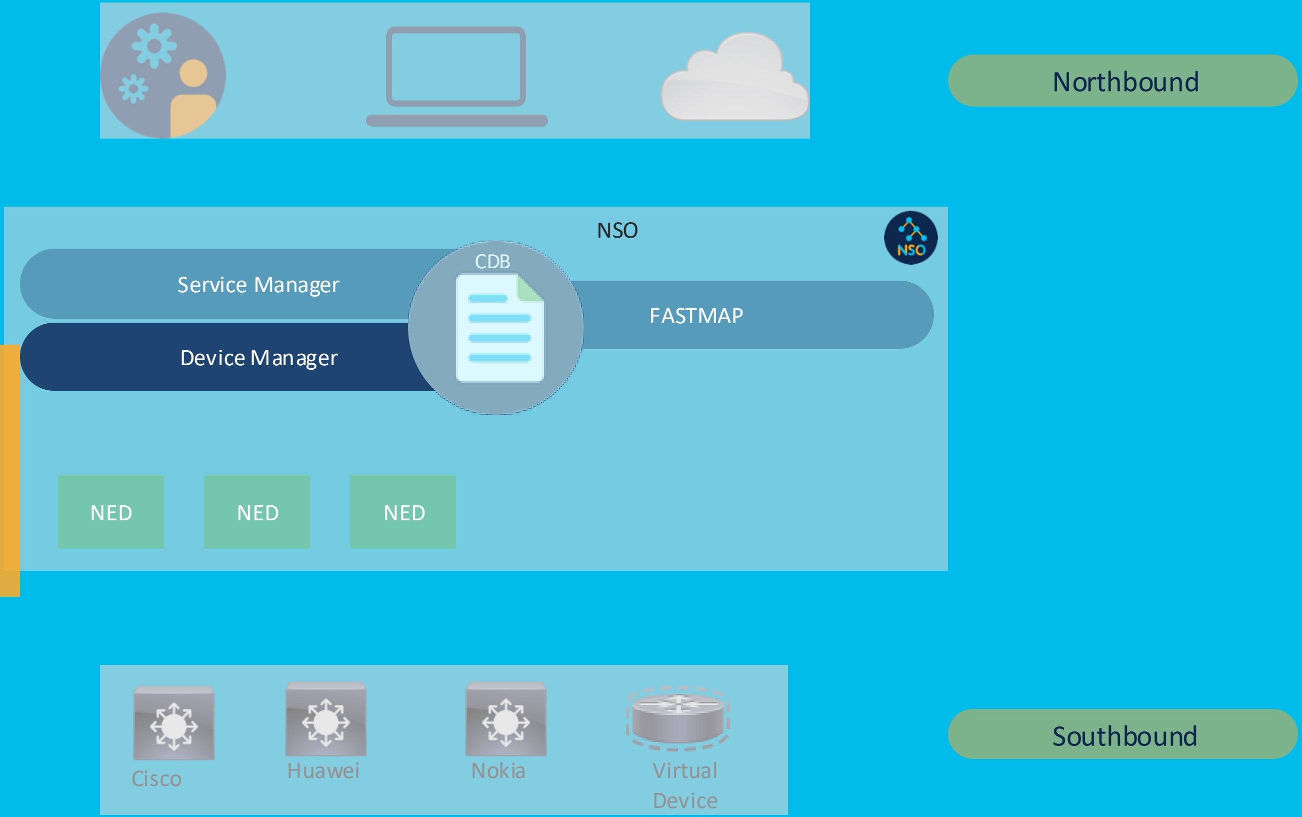
Southbound

NSO High Level Architecture

- Service configuration
- Service YANG model
- Service sychronization with device configuration



NSO High Level Architecture



Northbound

NSO

Service Manager

Device Manager

CDB

FASTMAP

NED

NED

NED

Cisco

Huawei

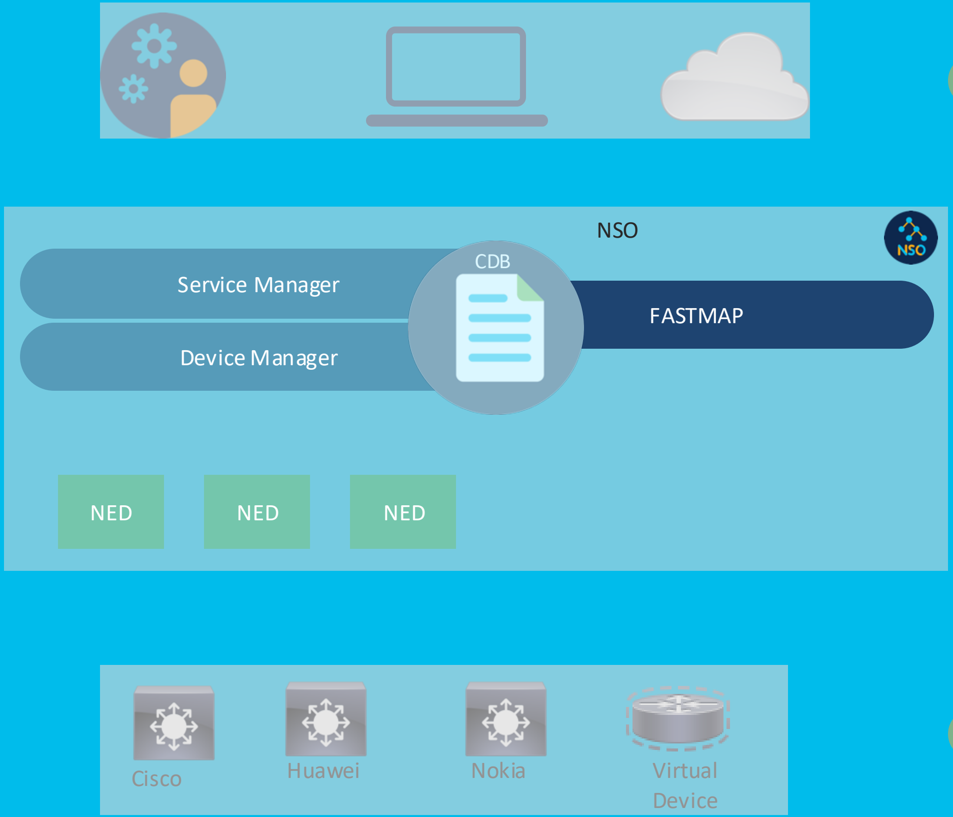
Nokia

Virtual Device

Southbound

- List of all managed devices
- Device YANG model coming from NED
- Device full configuration
- Sync-from, sync-to

NSO High Level Architecture

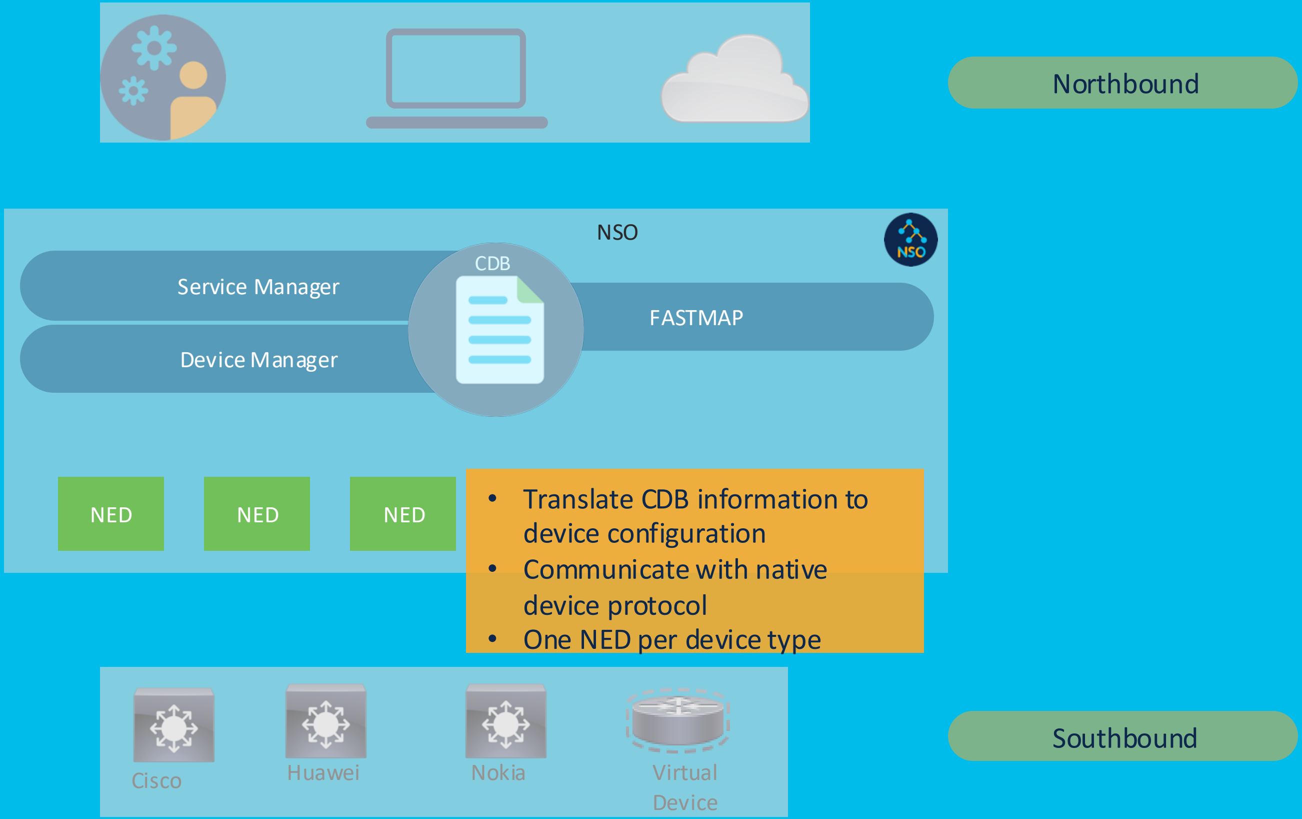


Northbound

- Calculates minimum difference needed for configuration to happen
- Control and optimize the southbound configuration
- Create, modify, delete

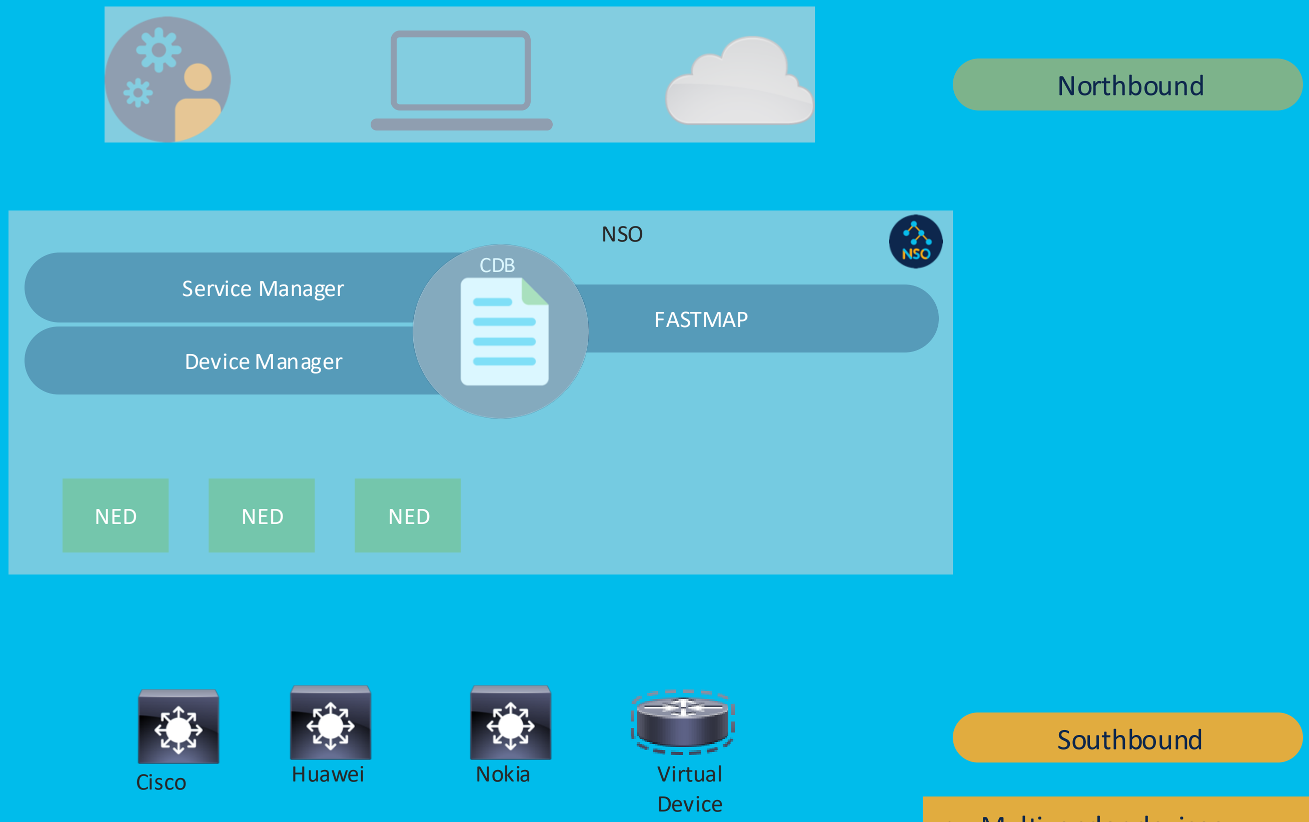
Southbound

NSO High Level Architecture



- Translate CDB information to device configuration
- Communicate with native device protocol
- One NED per device type

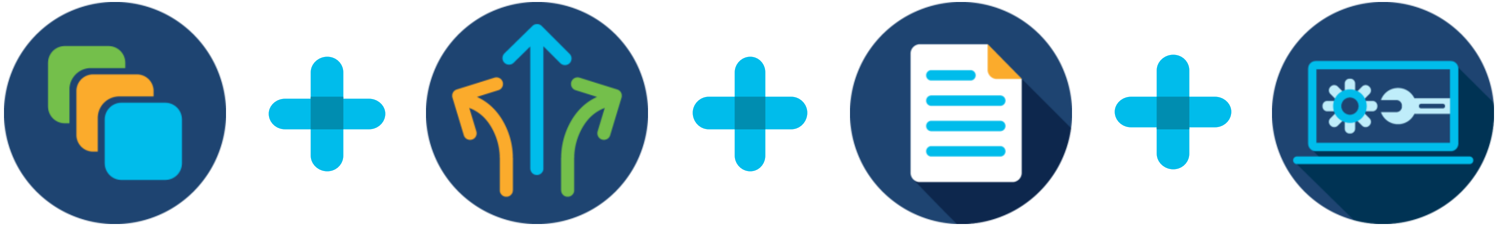
NSO High Level Architecture



- Multivendor devices
- Virtual devices
- VNF

What’s an NSO service?

- A declarative method to abstract and automate a task you want to do repeatedly
- Customer facing and resource facing services
- Internally it maintains mapping between inputs (user intent) and outputs (infrastructure configurations)



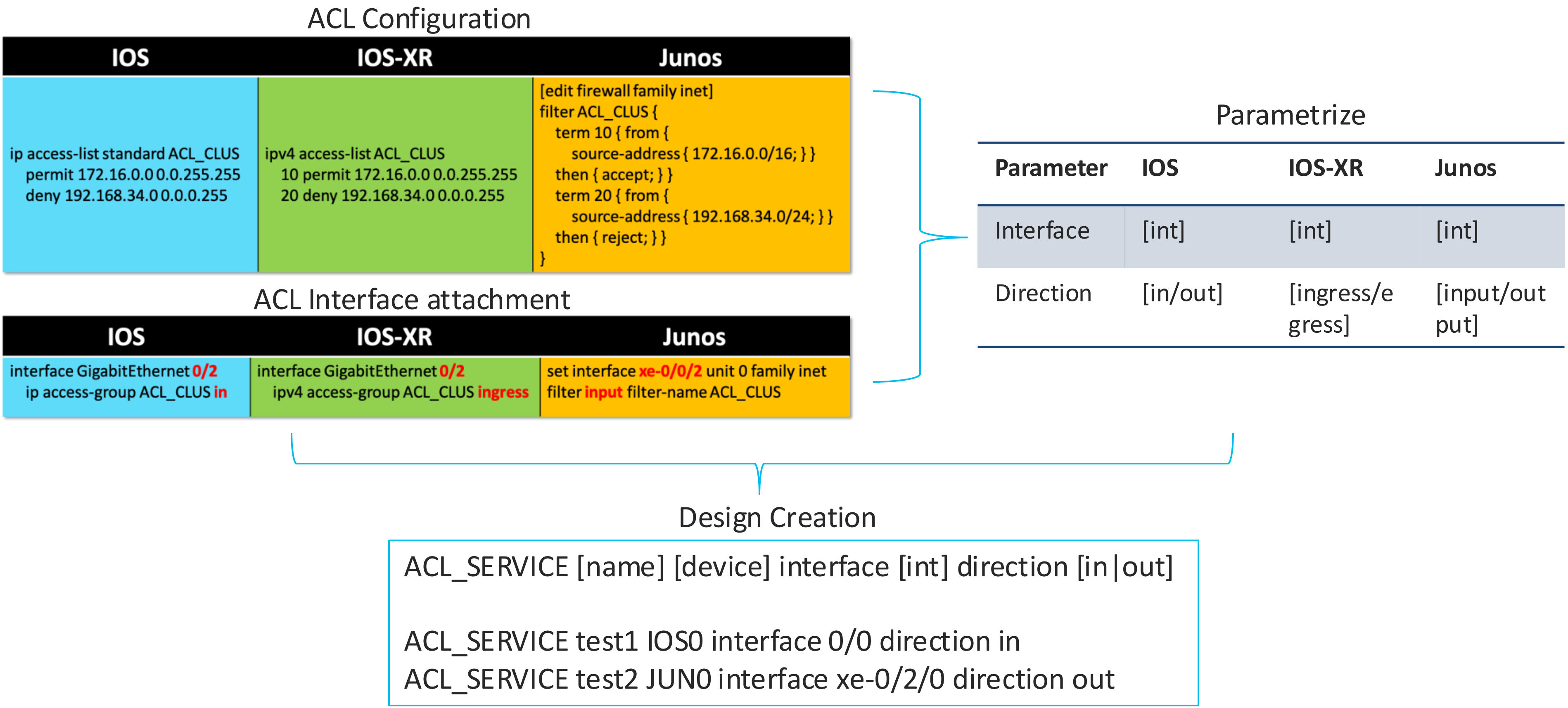
YANG Model
Defines the service in high level terms

Logic
Python/Java collects info, does verification

Template
XML written
How the service is rendered on devices

Package meta data
Version requirements, components, dependencies

Design Creation



ACL Configuration

IOS	IOS-XR	Junos

ip access-list standard ACL_CLUS
permit 172.16.0.0 0.0.255.255
deny 192.168.34.0 0.0.0.255
	ipv4 access-list ACL_CLUS
	10 permit 172.16.0.0 0.0.255.255
	20 deny 192.168.34.0 0.0.0.255
		[edit firewall family inet]
		filter ACL_CLUS {
		term 10 { from {
		source-address { 172.16.0.0/16; } }
		then { accept; } }
		term 20 { from {
		source-address { 192.168.34.0/24; } }
		then { reject; } }
		}

ACL Interface attachment

IOS	IOS-XR	Junos
interface GigabitEthernet 0/2 ip access-group ACL_CLUS in	interface GigabitEthernet 0/2 ipv4 access-group ACL_CLUS ingress	set interface xe-0/0/2 unit 0 family inet filter input filter-name ACL_CLUS

Parametrize

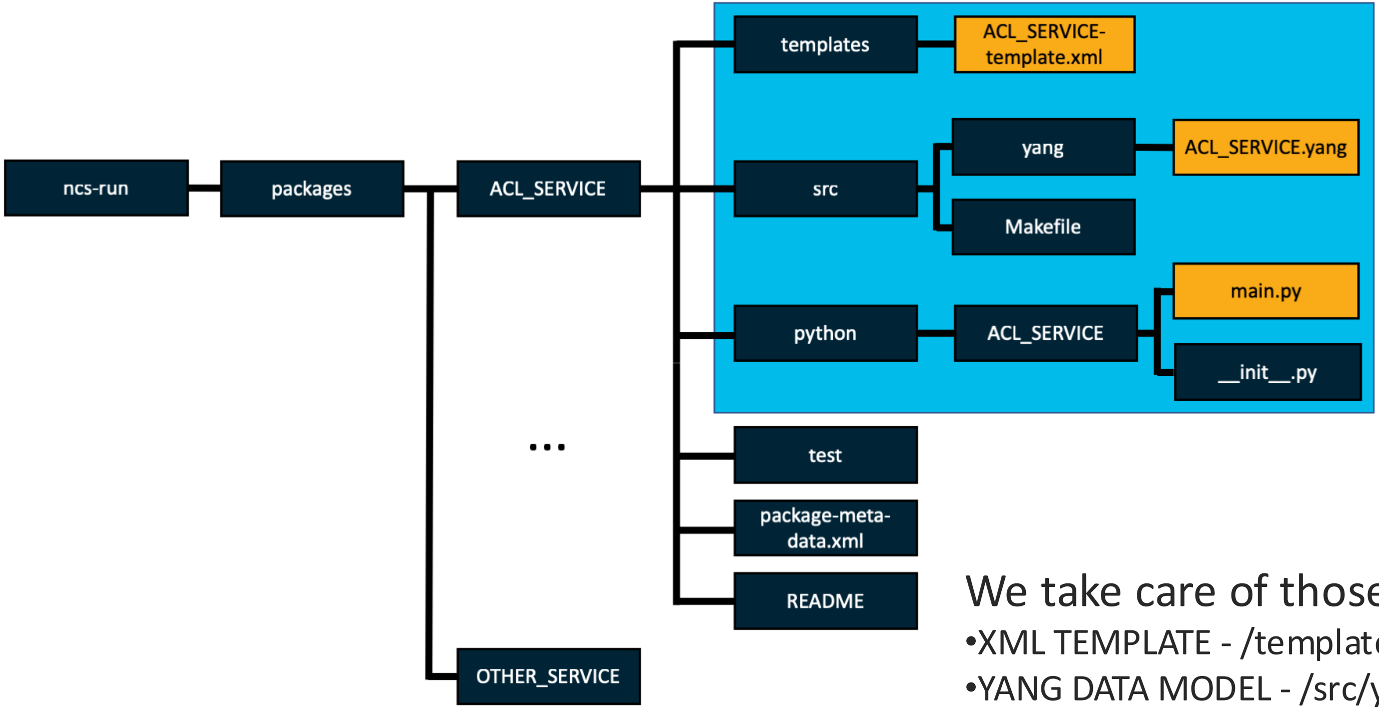
Parameter	IOS	IOS-XR	Junos
Interface	[int]	[int]	[int]
Direction	[in/out]	[ingress/egress]	[input/output]

Design Creation

```
ACL_SERVICE [name] [device] interface [int] direction [in|out]

ACL_SERVICE test1 IOS0 interface 0/0 direction in
ACL_SERVICE test2 JUN0 interface xe-0/2/0 direction out
```

Service Creation



We take care of those 3 parts :

- XML TEMPLATE - /template/ACL_SERVICE-template.xml
- YANG DATA MODEL - /src/yang/ACL_SERVICE.yang
- PYTHON LOGIC (optional) - /python/ACL_SERVICE/main.py

Successfully delivered NSO usecases

Just a summary of what can we do with Cisco NSO

Data Center

- Multi-vendor DC Fabric automation
- PaaS Network Automation
- Zero Touch Provisioning
- NSO + ACI

Security

- Multi-vendor Rule management
- Anti-Spoofing automation
- DMZ: Secure Agile Exchange

Campus

- Multi-technology Layers1/2/3
- Remote worker automation
- Wireless
- SDN-WAN automation

Transport

- Layers 2/3 VPN automation
- Mobile Core automation
- IP + Optical automation
- Transport SDN

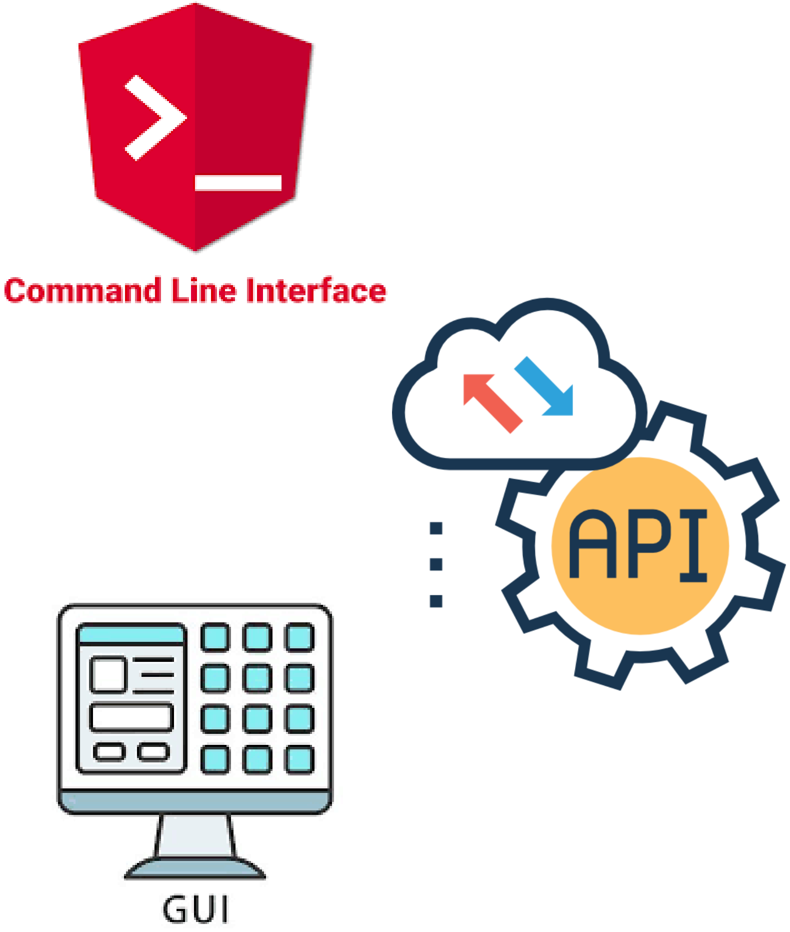
Access

- FTTH: OLT, DSLAM automation
- Subscriber automation, including Radius
- BNG, CMTS automation: IP pools

NFV

- Virtual functions instantiation
- Service chaining, across DC, Transport and network functions
- Management of Containers

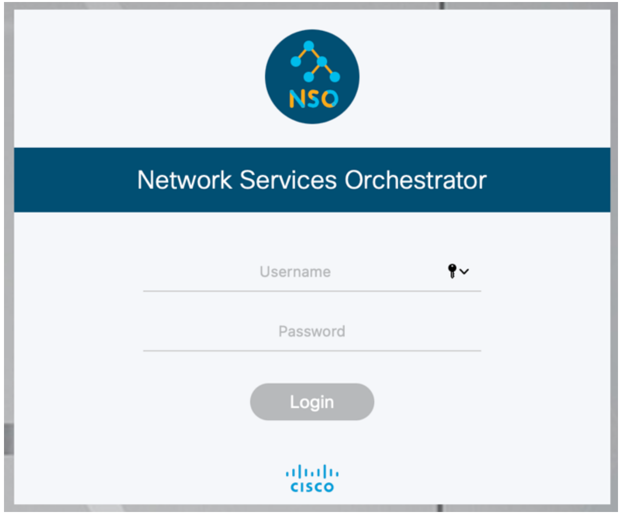
Interacting with NSO.



NSO GUI

- Disabled by default, can be enabled from ncs.conf for system install
- Plain HTTP version runs by default on port 8080, HTTPS on 8888
- Certs used by the HTTPS version are also defined in ncs.conf.
- Allows you to configure NSO, devices and services but also to monitor the current status.
- RBAC allows you to disable some functions to some users.

NSO GUI Login and Home page



NSO

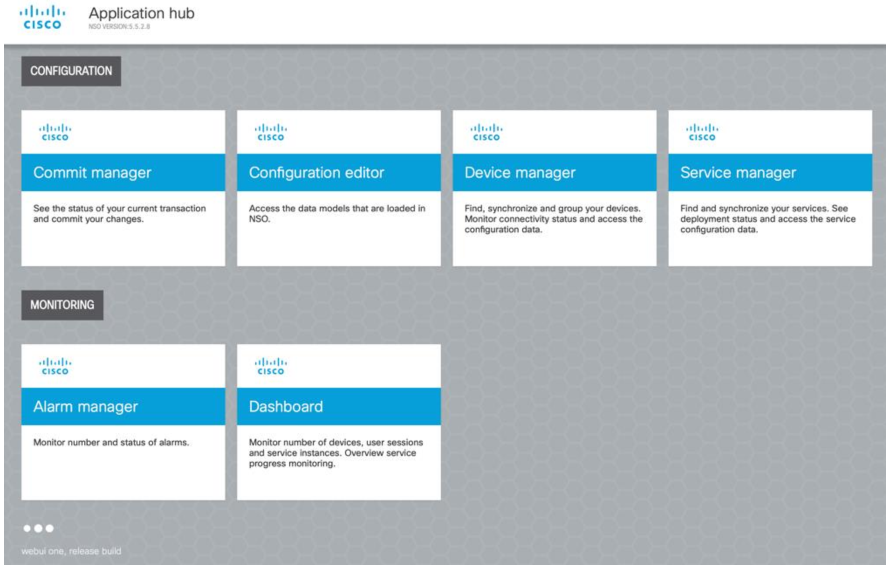
Network Services Orchestrator

Username

Password

Login

cisco



cisco Application hub

NSO VERSION:5.5.2.8

CONFIGURATION

Commit manager

See the status of your current transaction and commit your changes.

Configuration editor

Access the data models that are loaded in NSO.

Device manager

Find, synchronize and group your devices. Monitor connectivity status and access the configuration data.

Service manager

Find and synchronize your services. See deployment status and access the service configuration data.

MONITORING

Alarm manager

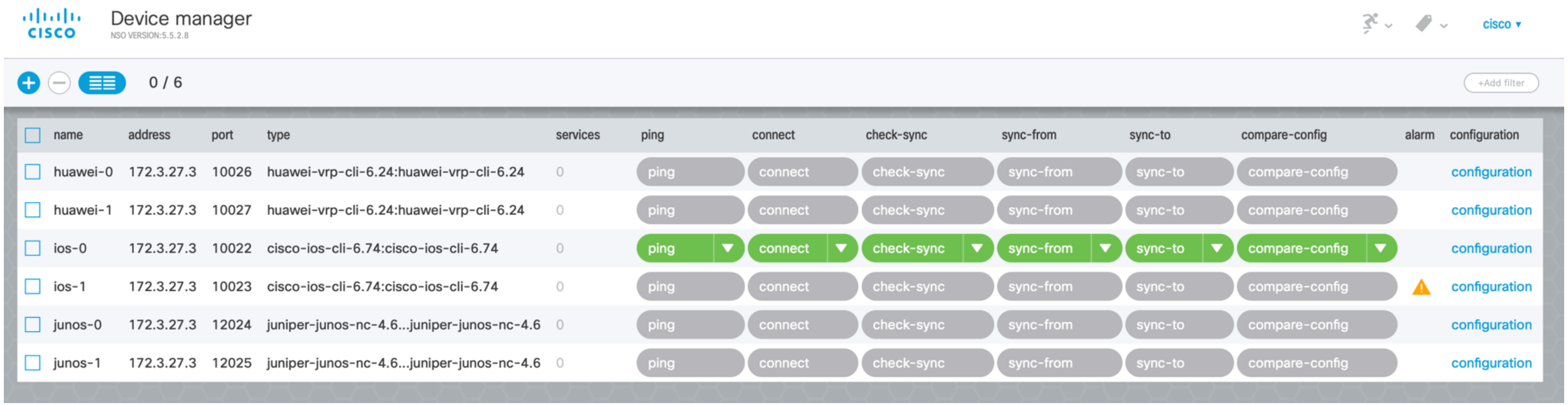
Monitor number and status of alarms.

Dashboard

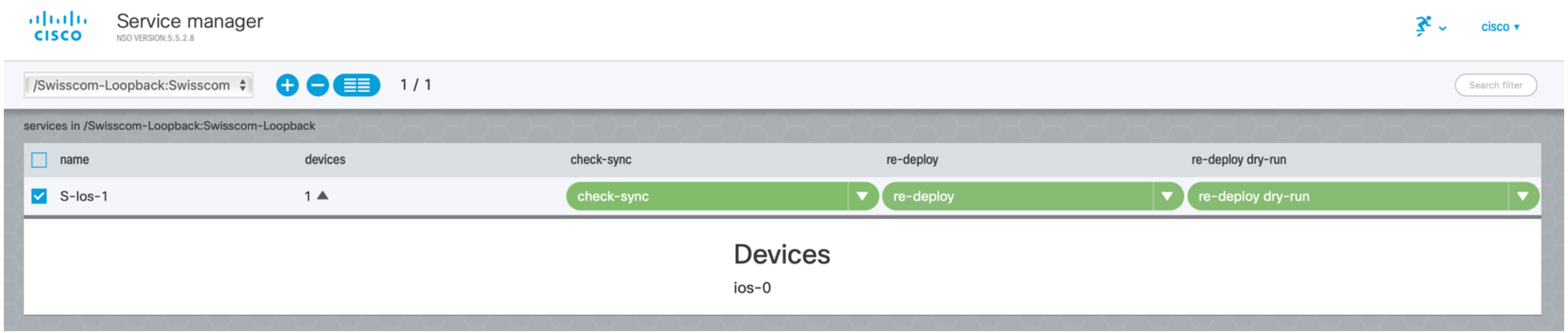
Monitor number of devices, user sessions and service instances. Overview service progress monitoring.

webui one, release build

NSO GUI Device and Service manager



□	name	address	port	type	services	ping	connect	check-sync	sync-from	sync-to	compare-config	alarm	configuration
□	huawei-0	172.3.27.3	10026	huawei-vrp-cli-6.24:huawei-vrp-cli-6.24	0	ping	connect	check-sync	sync-from	sync-to	compare-config		configuration
□	huawei-1	172.3.27.3	10027	huawei-vrp-cli-6.24:huawei-vrp-cli-6.24	0	ping	connect	check-sync	sync-from	sync-to	compare-config		configuration
□	ios-0	172.3.27.3	10022	cisco-ios-cli-6.74:cisco-ios-cli-6.74	0	ping ▼	connect ▼	check-sync ▼	sync-from ▼	sync-to ▼	compare-config ▼		configuration
□	ios-1	172.3.27.3	10023	cisco-ios-cli-6.74:cisco-ios-cli-6.74	0	ping	connect	check-sync	sync-from	sync-to	compare-config	⚠	configuration
□	junos-0	172.3.27.3	12024	juniper-junos-nc-4.6...juniper-junos-nc-4.6	0	ping	connect	check-sync	sync-from	sync-to	compare-config		configuration
□	junos-1	172.3.27.3	12025	juniper-junos-nc-4.6...juniper-junos-nc-4.6	0	ping	connect	check-sync	sync-from	sync-to	compare-config		configuration



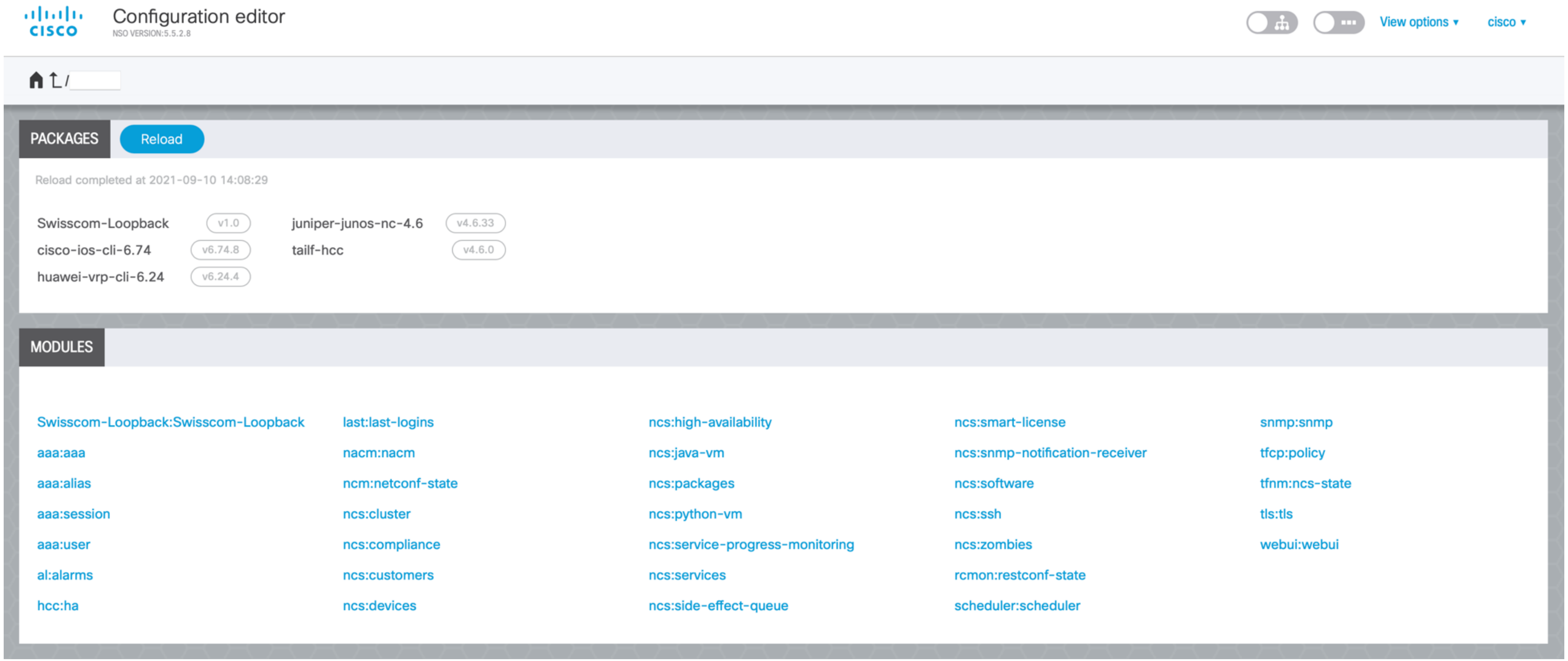
services in /Swisscom-Loopback:Swisscom-Loopback

□	name	devices	check-sync	re-deploy	re-deploy dry-run
☑	S-Ios-1	1 ▲	check-sync ▼	re-deploy ▼	re-deploy dry-run ▼

Devices

ios-0

NSO GUI Configuration editor



PACKAGES

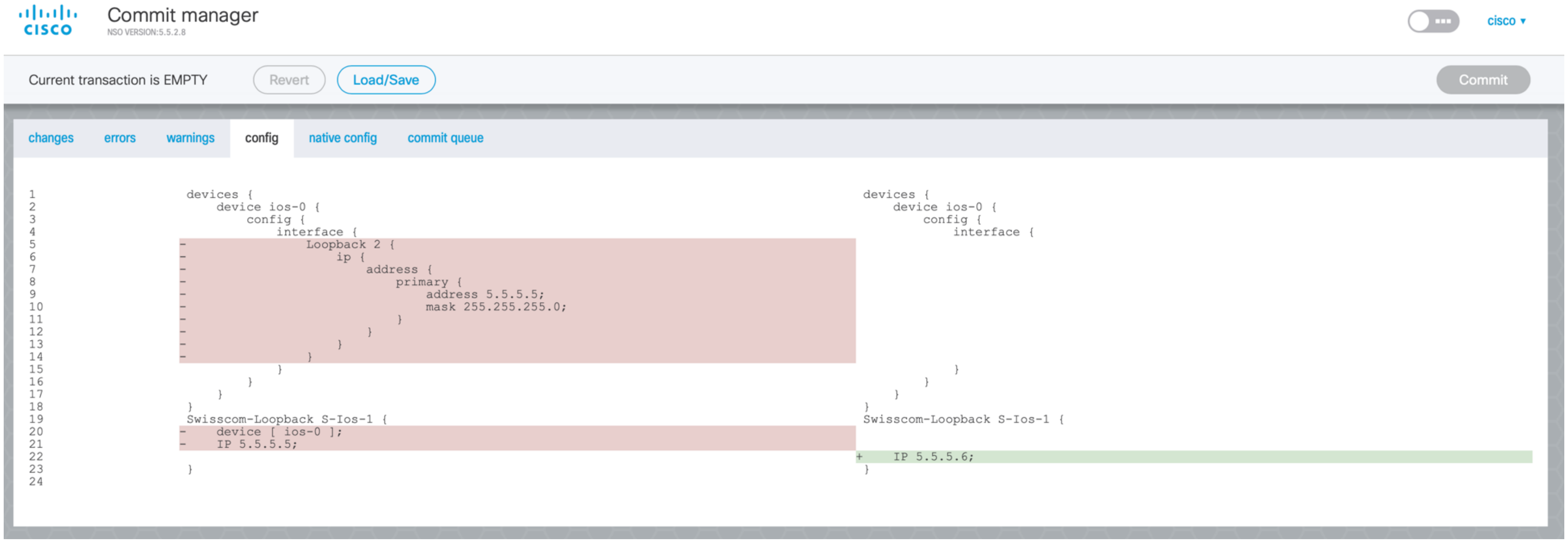
Reload completed at 2021-09-10 14:08:29

Swisscom-Loopback	v1.0	juniper-junos-nc-4.6	v4.6.33
cisco-ios-cli-6.74	v6.74.8	tailf-hcc	v4.6.0
huawei-vrp-cli-6.24	v6.24.4		

MODULES

Swisscom-Loopback:Swisscom-Loopback
aaa:aaa
aaa:alias
aaa:session
aaa:user
al:alarms
hcc:ha
last:last-logins
nacm:nacm
ncm:netconf-state
ncs:cluster
ncs:compliance
ncs:customers
ncs:devices
ncs:high-availability
ncs:java-vm
ncs:packages
ncs:python-vm
ncs:service-progress-monitoring
ncs:services
ncs:side-effect-queue
ncs:smart-license
ncs:snmp-notification-receiver
ncs:software
ncs:ssh
ncs:zombies
rcmon:restconf-state
scheduler:scheduler
snmp:snmp
tfcp:policy
tfnm:ncs-state
tls:tls
webui:webui

NSO GUI Commit manager



Current transaction is EMPTY

```
devices {
    device ios-0 {
        config {
            interface {
-               Loopback 2 {
-                   ip {
-                       address {
-                           primary {
-                               address 5.5.5.5;
-                               mask 255.255.255.0;
-                           }
-                       }
-                   }
-               }
            }
        }
    }
}
Swisscom-Loopback S-Ios-1 {
-   device [ ios-0 ];
-   IP 5.5.5.5;

}
```

```
devices {
    device ios-0 {
        config {
            interface {










            }
        }
    }
}
Swisscom-Loopback S-Ios-1 {


+   IP 5.5.5.6;
}
```

Questions?