Cisco live!

January 29 – February 2, 2018 · Barcelona

BRKNMS-2289

NSO for Network Operators

Michael Maddern – Technical Marketing Engineer
Jay Kurji – Solutions Architect

Cisco live!

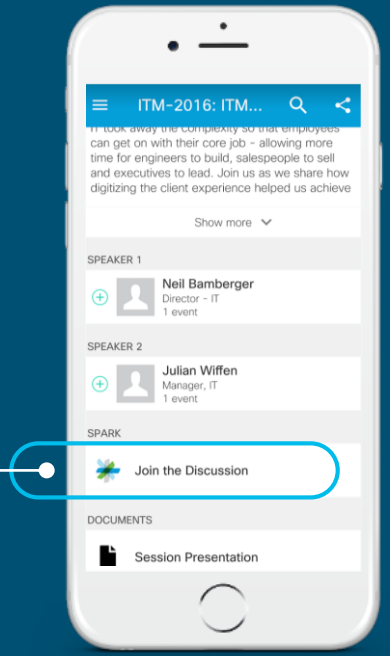
Cisco Spark

Questions?

Use Cisco Spark to communicate with the speaker after the session

How

1. Find this session in the Cisco Live Mobile App
2. Click “Join the Discussion”
3. Install Spark or go directly to the space
4. Enter messages/questions in the space



cs.co/ciscolivebot#BRKNMS-2289

Model driven automation of your **entire network**

Agenda

- Introduction
- NSO Overview
- NSO for Network Engineers
- NSO for Operations Teams
- NSO for Service Developers
- Conclusion

Cisco Network Services Orchestrator

Open via Northbound APIs

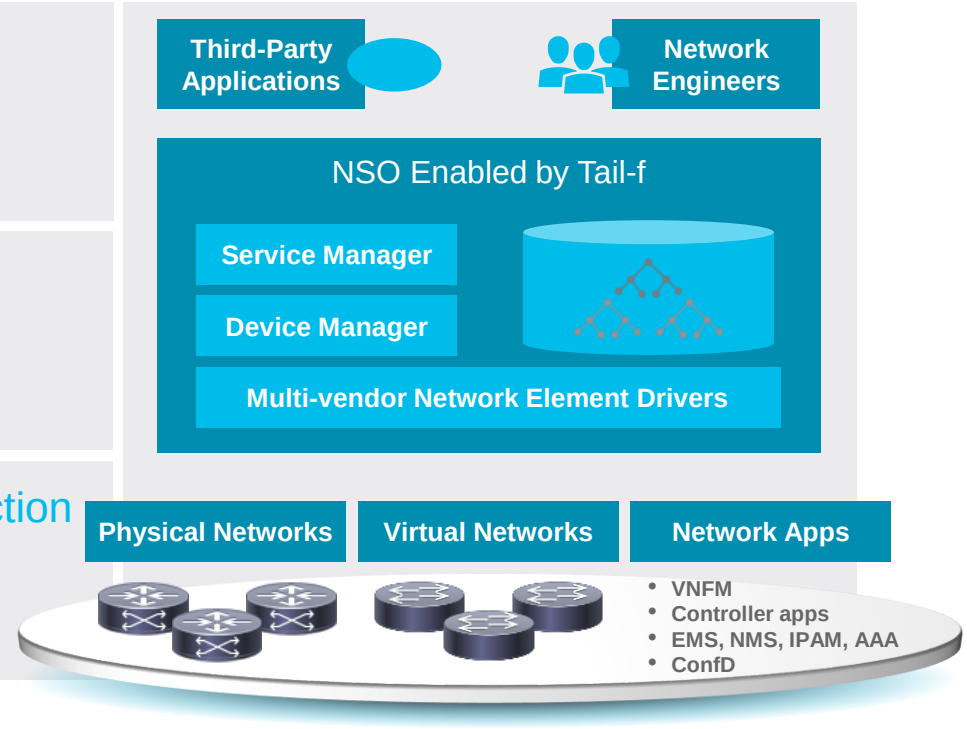
- OSS and BSS
- Third-party applications
- DevOps support

Agile Automation

- Model driven
- Active service/network view
- Any service, any device
- Multi-vendor support

Automated Network Abstraction

- Physical devices
- Virtual devices
- Network applications



A two minute introduction to NETCONF/YANG

NETCONF

Network management protocol specifically designed to support service activation and provisioning.

- Encrypted, efficient transport
- Extensible
- Transactional
- Network-wide

A two minute introduction to NETCONF/YANG

YANG

Text based data modeling language designed for use with NETCONF.

- Operator friendly
- Precise
- Extensible
- Human readable

Key Market Trend Observations

Execution at the speed of software

- Agility, DevOps, NFV, SDN, new services platforms

Changing customer behavior and new expectations

- Everything on demand
- New services with a press of a button

Rapidly changing business models

- Cloud services, virtualization, programmable networks
- New ecosystems and value chains
- OTT Co-opetition

All of this requires successful, flexible automation.
But complexity has destroyed many automation initiatives.

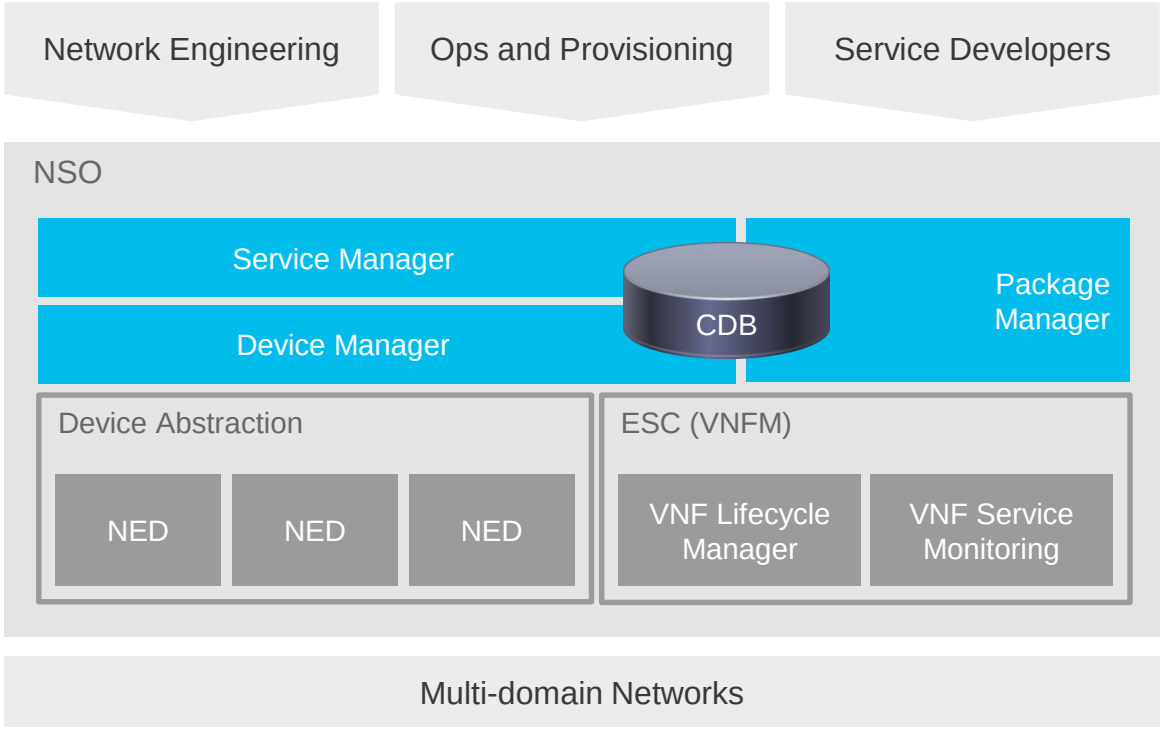
Departmental Pain Points

Network Engineer “Automation”	**Ops and Provisioning Team** “Customer Experience”	**Service Developers** “Time-to-Market”
Day-to-day management of rapidly growing, complex networks **Challenges** • Error-prone manual tasks • Growing backlog • Virtualization is coming	Provisions services and manages service quality in networks **Challenge** • No service insight • Lack of automation • Quality issues in service delivery	Develops new network services on demand **Challenge** • Implementation time • Cost of change • Lack of tooling

Transition Towards Automation

Network Engineer "Automation"	Ops and Provisioning Team "Customer Experience"	Service Developers "Time-to-Market"
Day-to-day management of rapidly growing, complex networks	Provisions services and manages service quality in networks	Develops new network services
Challenge • Device configuration tasks • Growing backlog • Virtualization is coming	**Challenge** • No service insight • Lack of automation • Quality issues in service delivery	**Challenge** • Implementation time • Cost of change • Lack of tooling
Network API Utilize a single interface to all network devices	**Service Abstraction** Leverage one central API for all services	**Transformation** Develop your own services
Stage 1	**Stage 2**	**Stage 3**

System Overview

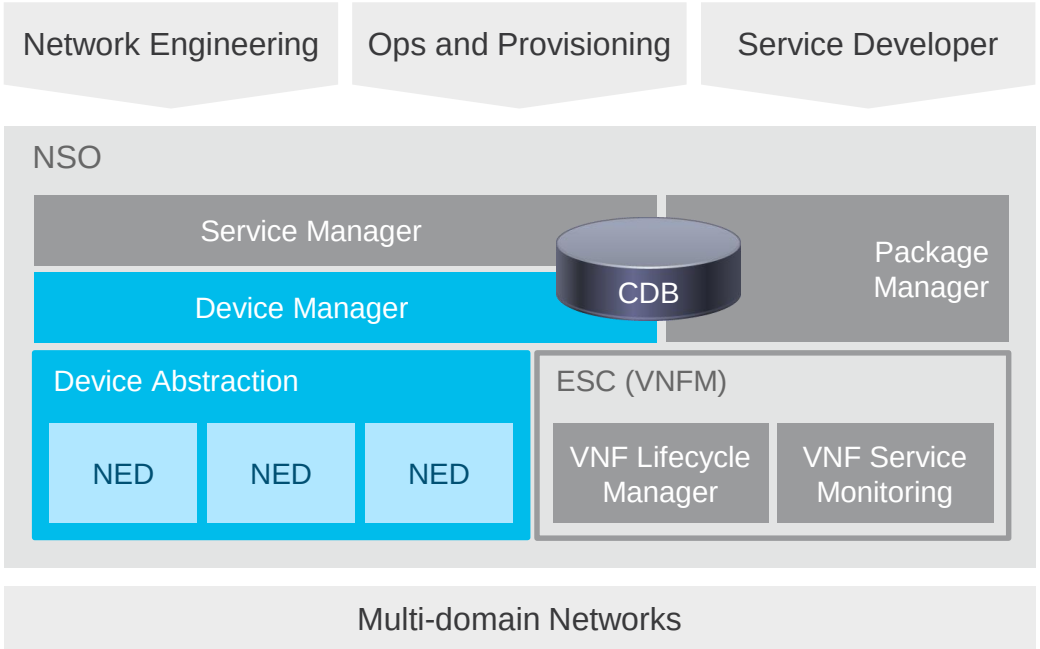


- Model-driven end-to-end service lifecycle and customer experience in focus
- Seamless integration with existing and future OSS/BSS environment
- Loosely-coupled and modular architecture leveraging open APIs and standard protocols
- Orchestration across multi-domain and multi-layer for centralized policy and services across entire network

Stage #1: The Network API

Network Engineer "Automation"	**Ops and Provisioning Team** "Customer Experience"	**Service Developers** "Time-to-Market"
Day-to-day management of rapidly growing, complex networks	Provisions services and manages service quality in networks	Develops new network services
Challenge • Device configuration tasks • Growing backlog • Virtualization is coming	**Challenge** • No service insight • Lack of automation • Quality issues in service delivery	**Challenge** • Implementation time • Cost of change • Lack of tooling
Network API Utilize a single interface to all network devices	**Service Abstraction** Leverage one central API for all services	**Innovation** Develop your own services
Stage 1	**Stage 2**	**Stage 3**

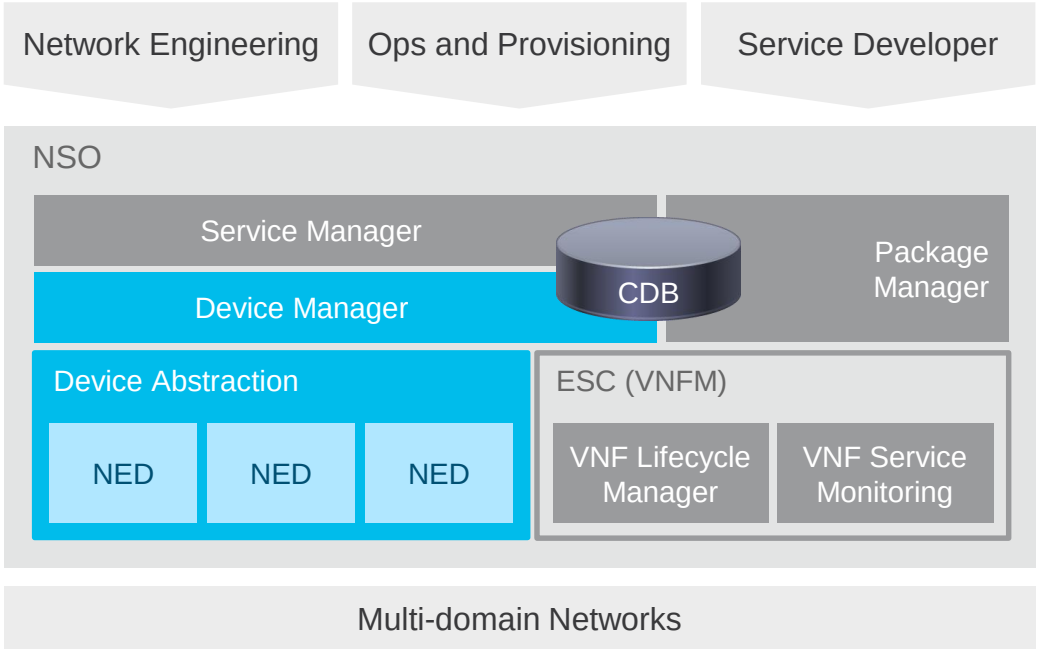
The Network API - Overview



Features in focus:

- Multivendor abstraction through NEDs
- Single data store for all network elements under management
- Multiple interfaces including CLI, REST, Python
- Templates and compliance reporting

Multivendor Abstraction Through NEDs



A NED abstracts

- Underlying protocol and data-models
- Error-handling

The NED computes the ordered sequence of device-specific commands to go:

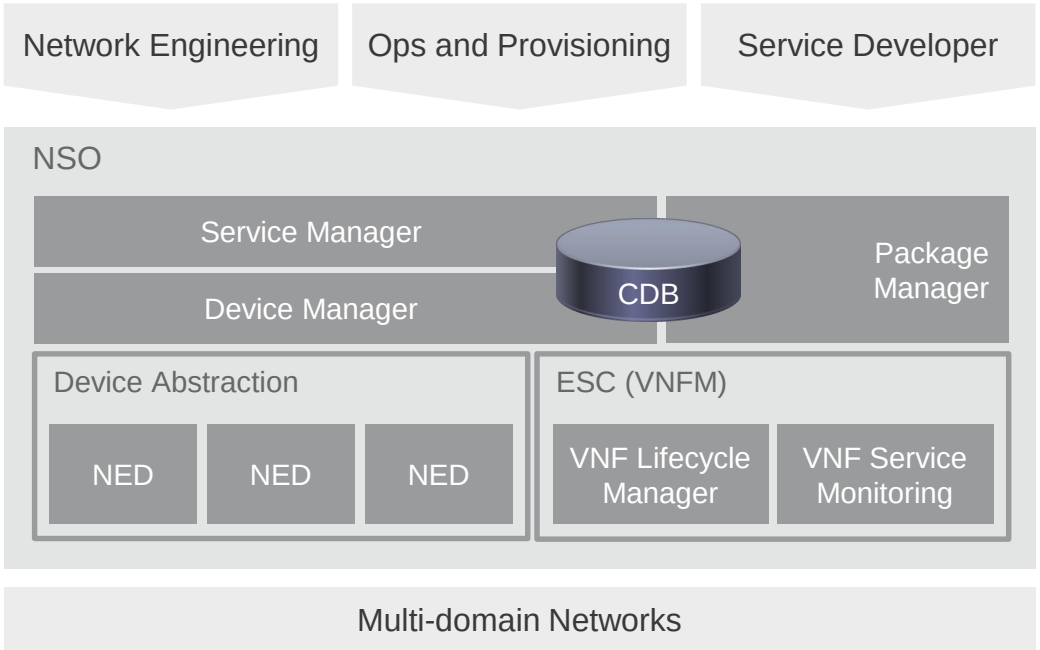
- from current configuration state
- to desired configuration state

Key benefits include: removes the device adapter problem. Removes complex device logic from the service logic

The Industry's Broadest Multivendor Support

Over 100 Supported NEDs—Customization Available

The Configuration Data Store



Built for speed at scale:

- In memory with journaled backend
- YANG is native schema language
- …including when, must, XPath, etc

Highly available

- 1:N hot-standby
- Synchronous and asynchronous slaves
- Slave chaining

Fully integrated

- Managed through NSO interfaces
- Runs in main process memory
- Automatic versioning of YANG modules

Network-wide CLI

- Two flavors of CLI including all main interaction idioms including control-commands, command-line editing
- Strict separation between operational data and configuration data
- Range and group operations for performing configuration changes on sets of devices
- Full AAA (NACM) integration provides policies on both models and instance data
- Leverages the two-phase commit engine in NSO to provide all-or-nothing changes including explicit validation stages

```
admin@ncs(config-endpoint-c2)# commit
Commit complete.
admin@ncs(config-endpoint-c2)#
admin@ncs# show running-config devices device pe2
devices device pe2
 address   127.0.0.1
 port      10030
 authgroup default
 device-type cli ned-id cisco-ios-xr
 state admin-state unlocked
 config
  cisco-ios-xr:hostname PE1
  cisco-ios-xr:vrf volvo
   address-family ipv4 unicast
    import route-target
     65001:1
    exit
    export route-target
     65001:1
    exit
   exit
  exit
  cisco-ios-xr:interface MgmtEth 0/0/CPU0/0
  exit
  cisco-ios-xr:interface TenGigE 0/3/0/0
   shutdown
  exit
  cisco-ios-xr:interface TenGigE 0/3/0/1
   shutdown
admin@ncs(config-endpoint-c2)#
```

Templates and Compliance Reporting

- Engineering teams create device templates from device configuration
- Device templates are then manually applied to groups of devices, reporting diffs
- This process can then be packaged into a compliance report to produce reports (plain text, XML, HTML)

Demo 1

- Device Management
  - Multi-vendor
  - Dry-run operations
  - Rollback
- Interfaces
  - CLI
  - Web UI
- Device Templates
  - SNMP Communities
  - Access Lists
- Compliance Reports

APIs and Language Bindings

- Remember: the northbound APIs are all clients to the same YANG-based datastore
- Many customers start (and build trust) using the CLI, but gradually introduces e.g. REST for scripting trivial tasks
- Choice of technology tightly related to team background, specific use cases and more

Network Scripting

- Traditional scripting
  - Towards CLI interfaces
    - Unstructured / human-orientated
    - Complex regular expressions
    - Parsing libraries (i.e. Netmiko)
  - Manual credential and connection management
- Scripting with NSO
  - Structured data (YANG)
    - Fast easy-to-use API across all devices
  - Transactional guarantees
  - Device management

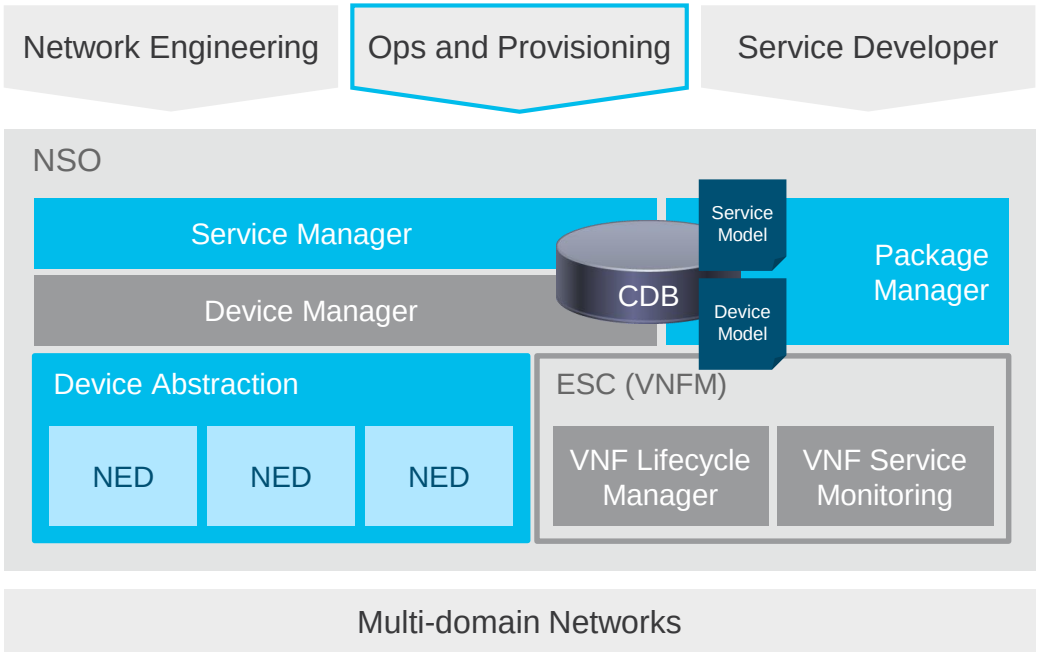
Demo 2

- NSO ad-hoc scripting
  - Python
  - Maagic API

Stage #2: Network Service Abstraction

Network Engineer "Automation"	Ops and Provisioning Team "Customer Experience"	Service Developers "Time-to-Market"
Day-to-day management of rapidly growing, complex networks **Challenges** • Device configuration tasks • Growing backlog • Virtualization is coming	Provisions services and manages service quality in networks **Challenge** • No service insight • Lack of automation • Quality issues in service delivery	Develops new network services **Challenge** • Implementation time • Cost of change • Lack of tooling
Network API Utilize a single interface to all network devices	**Service Abstraction** Leverage one central API for all services	**Innovation** Develop your own services
Stage 1	Stage 2	Stage 3

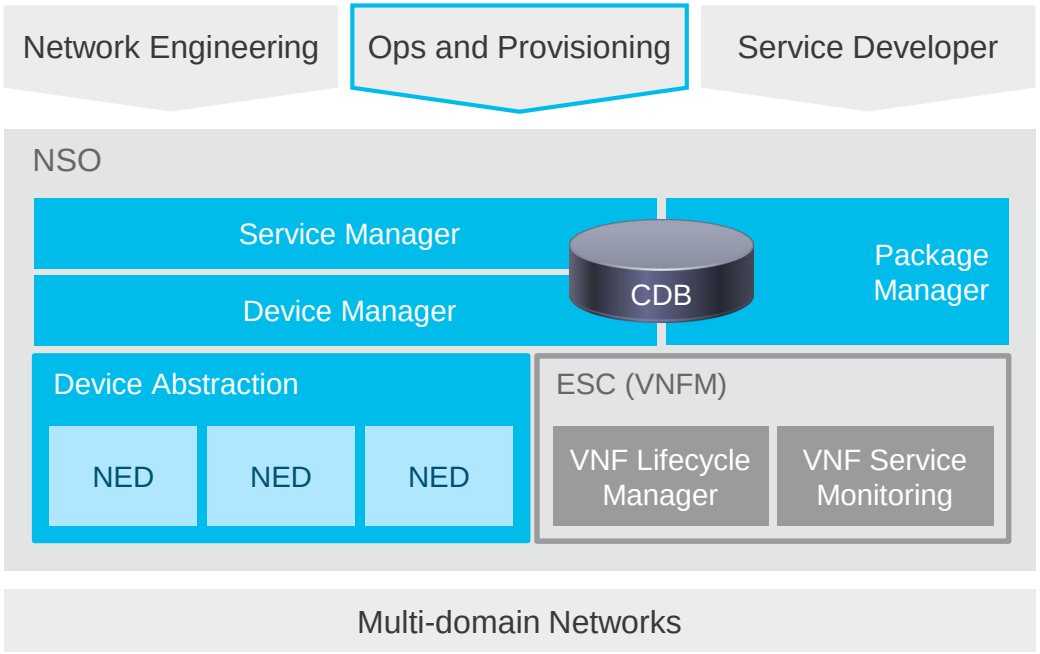
Network Service Abstraction - Overview



Features in focus:

- Full service lifecycle management
- All-or-nothing changes across devices
- Service insight
- Orchestrated assurance

Transactions and Models

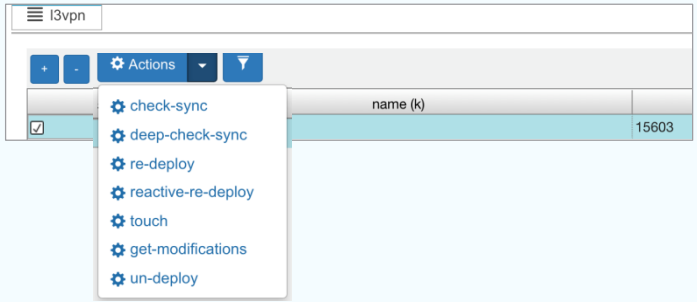


The datastore:

- Implements full ACID properties
- Uses YANG as native schema language
- Provides a two-phase commit protocol towards the network for distributed atomicity

Service Insight

- In order to maintain intent, we need to be able to answer:
  - What is the resulting configuration from this service?
  - Which service(s) does this configuration parameter come from?
  - Are the two in sync?
- To support:
  - Remedial actions
  - Service migration
  - Service discovery
- NSO provides full referential integrity between service and device layers




```
devices {
   device ce5 {
      config {
         ios:ip {
            access-list {
               extended {
+                 ext-named-acl GLOBAL-call-signaling {
+                    ext-access-list-rule "permit tcp any any range 5060 5061";
+                 }
+                 ext-named-acl GLOBAL-ssh {
+                    ext-access-list-rule "permit tcp any any range 22 22";
+                 }
```

Demo 3

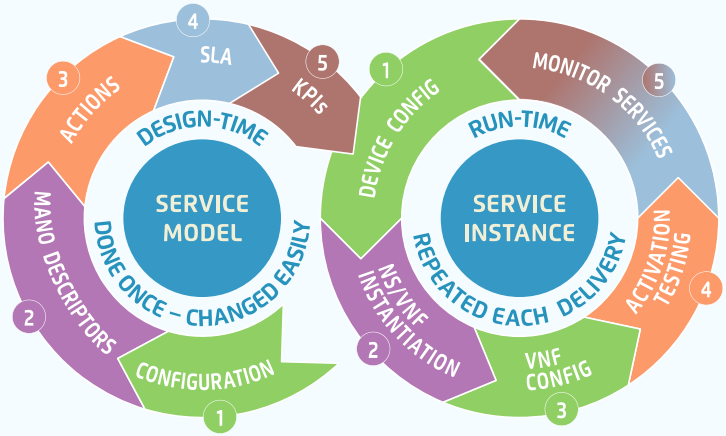
- MPLS VPN Demo
  - Service creation through REST Interface
  - Service creation through custom UI / portal
  - View device changes
  - Out of band changes and re-deploy

Orchestrated Assurance

- Extending the service models with KPI definitions and SLAs
- Working with a programmable collector and correlator
- Allows us to…
  - Automate activation tests and service assurance
  - Provide service-level assurance in hybrid networks

"To assure what is orchestrated, we must orchestrate assurance"

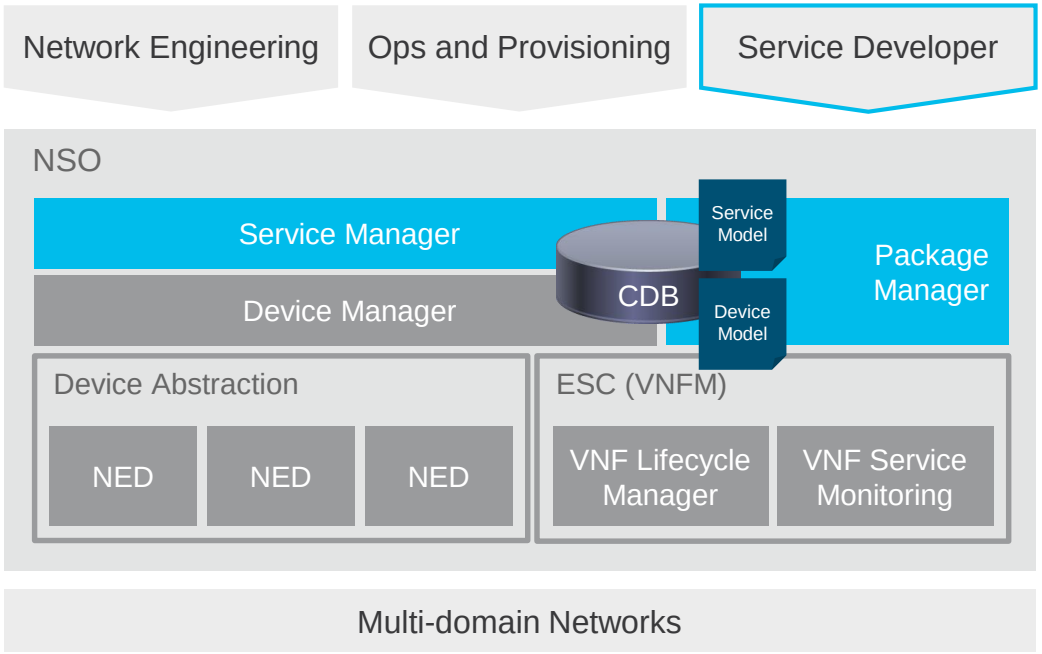
-- Wise Person



Stage #3 Transformation

Network Engineer "Automation"	Ops and Provisioning Team "Customer Experience"	Service Developers "Time-to-Market"
Day-to-day management of rapidly growing, complex networks	Provisions services and manages service quality in networks	Develops new network services
Challenges • Device configuration tasks • Growing backlog • Virtualization is coming	**Challenge** • No service insight • Lack of automation • Quality issues in service delivery	**Challenge** • Implementation time • Cost of change • Lack of tooling
Network API Utilize a single interface to all network devices	**Service Abstraction** Leverage one central API for all services	**Innovation** Develop your own services
Stage 1	Stage 2	Stage 3

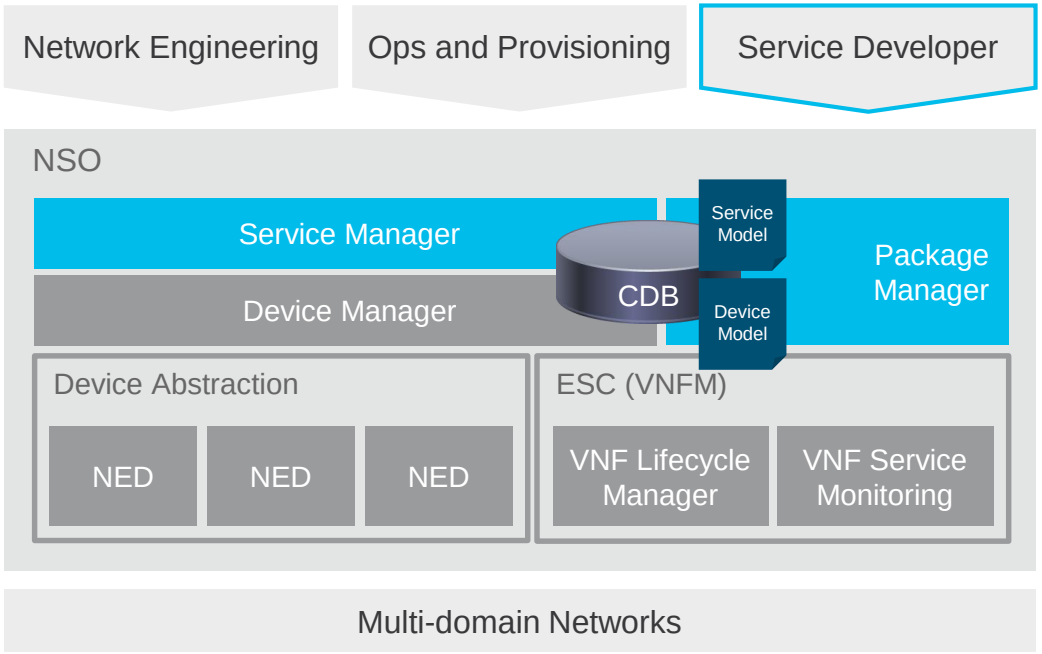
Transformation - Overview



Features in focus

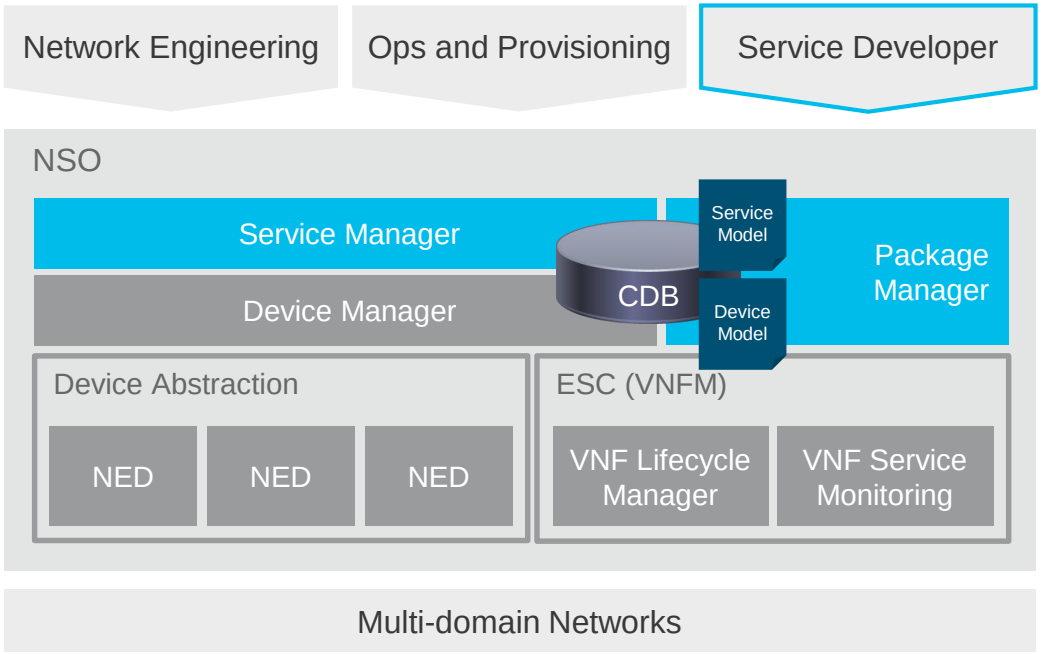
- Fully model-driven framework
- Direct and reactive stateful convergence of services
- Package lifecycle manager
- Suite of design-time development tools

Model-based Architecture



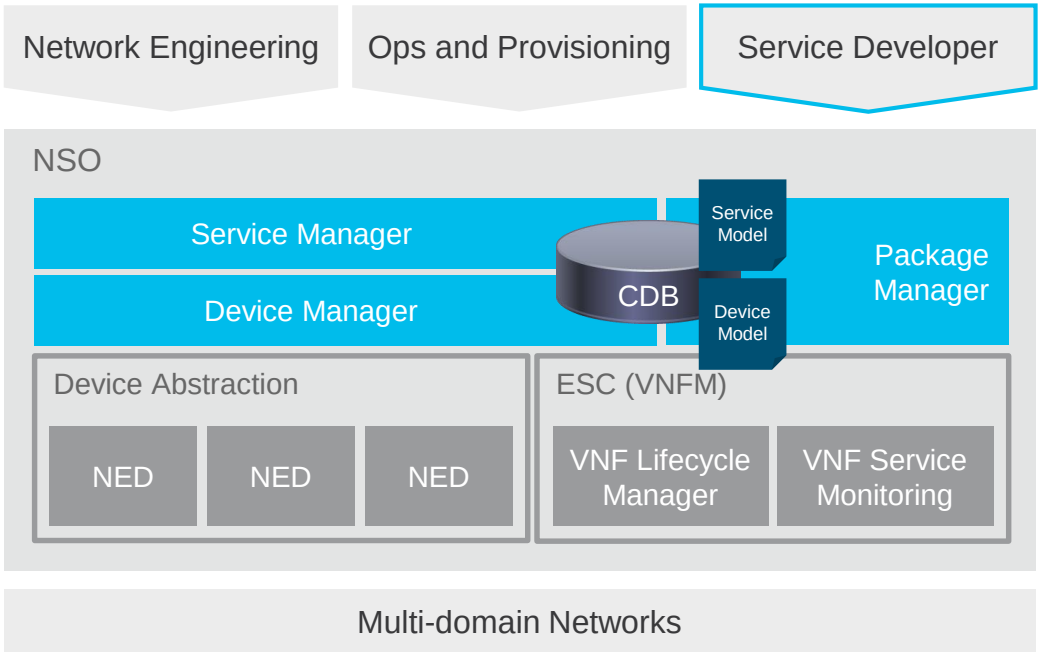
- NSO assumes nothing about:
  - Network services
  - Network devices
- All data sets strictly defined by YANG models
- Tree-to-tree mapping reduces coding for lifecycle to absolute minimum

Model-based Architecture (cont'd)



- Developer owns and versions formal service definition:
  - Product management defines the services
  - Infrastructure team deploys and manages system
  - Ops and Provisioning team consumes the services
- The Service Model is an exact black-box specification. Fast iteration of design-time changes.

Stateful Convergence

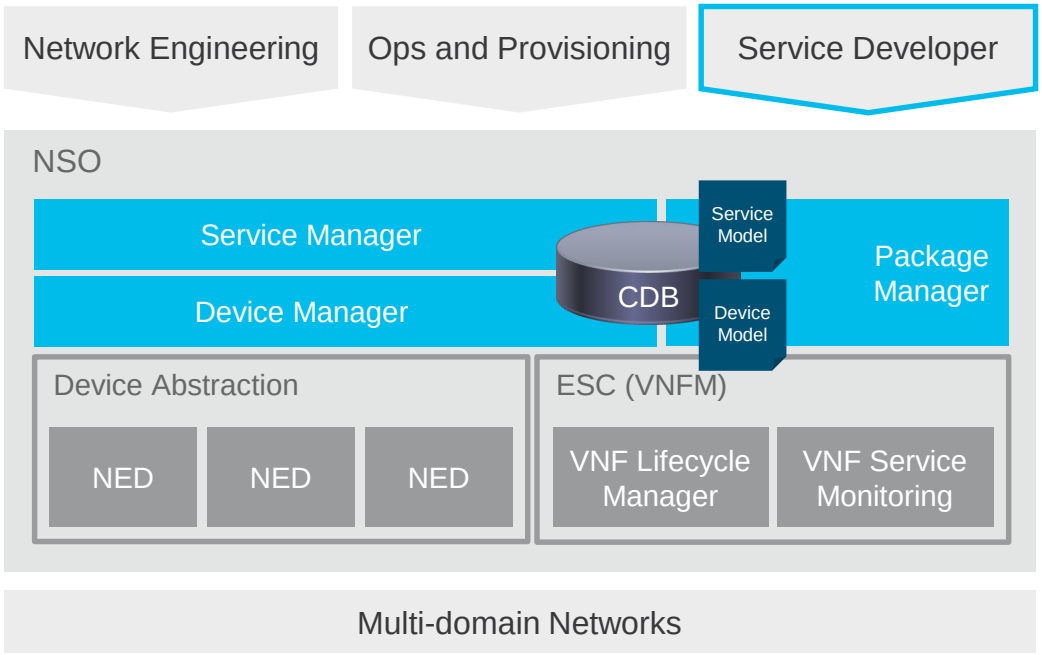


- Only create operation needs to be declared
- Update, delete inferred (!)
- Decomposition logic in:
  - Service templates for straight mapping
  - Java, Python for programmatic mapping (side effects, etc)
  - …or a combination of both

Demo 4

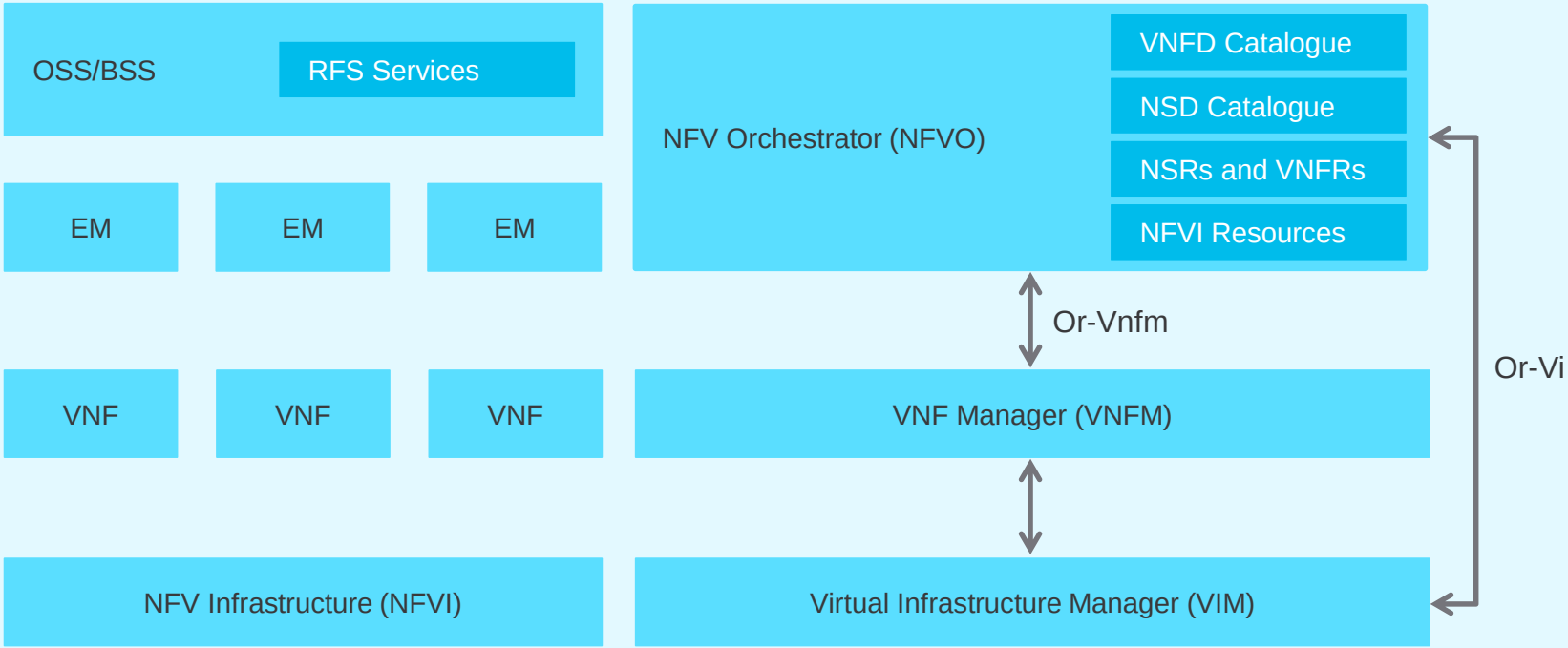
- MPLS VPN Demo
  - Service changes
  - Service migration
  - Service delete

Reactive Fastmap

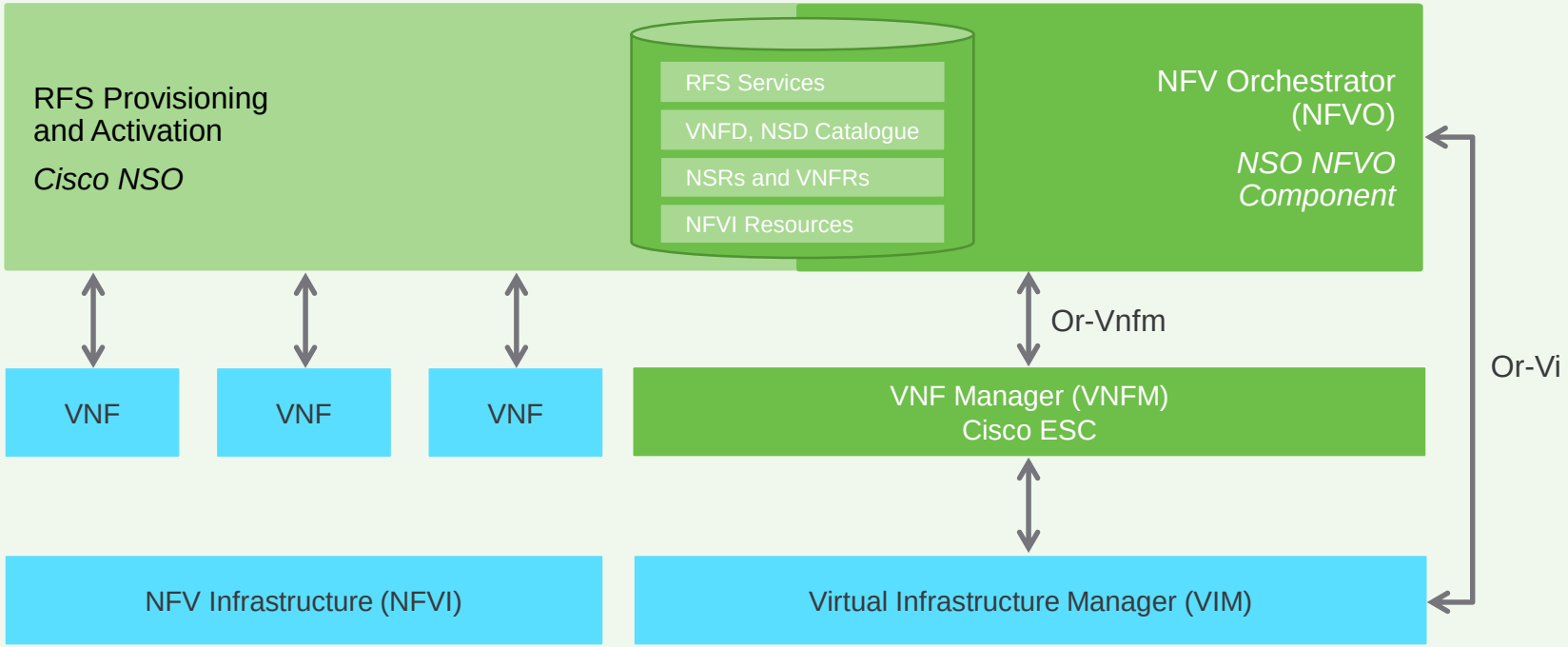


- Events happen in the network that may impact the service instances:
  - VMs started, moved or destroyed
  - Topology changes
- Reactive FastMap calculates the minimum diff to drive towards intent

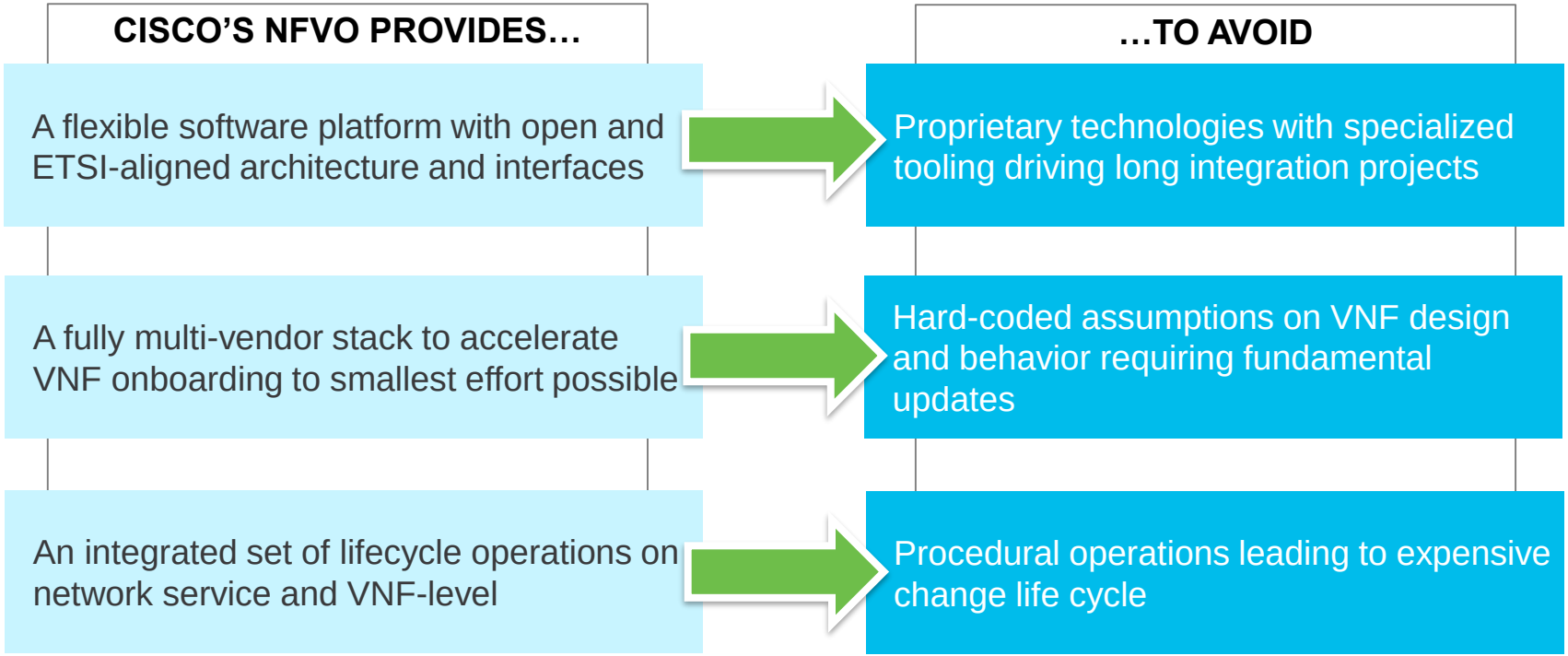
NFVO High Level Architecture



NFVO High Level Architecture Mapping



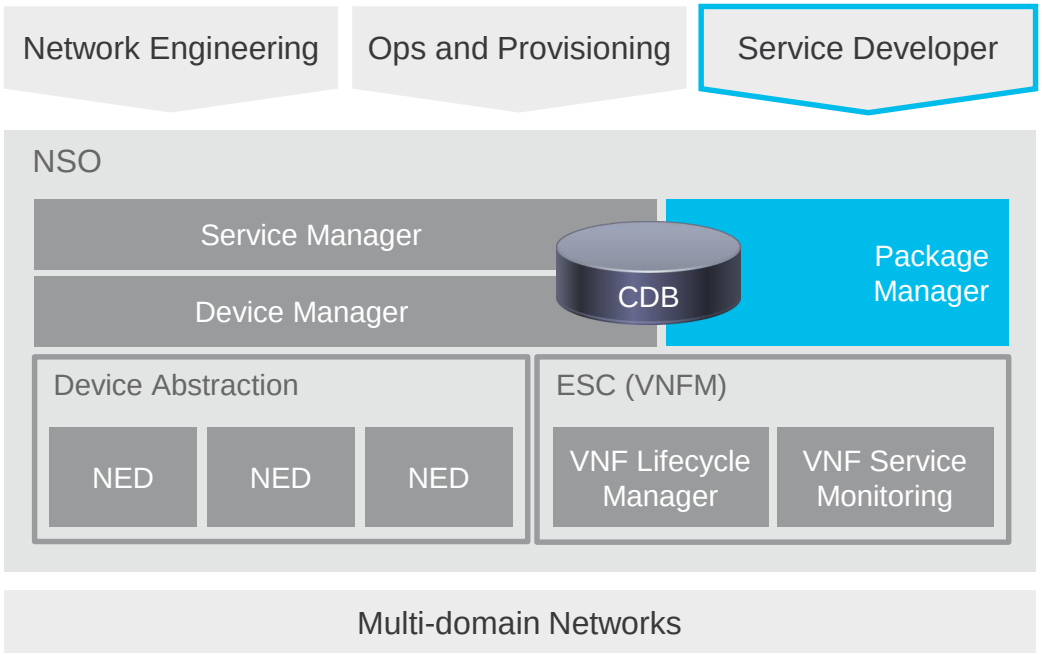
NFV Orchestration Challenges
Lessons Learned



Demo 5

- Virtual MPLS VPN
  - NFVO
  - Elastic Services Controller
  - OpenStack
- Orchestrated Assurance
  - Netrounds Control Centre
  - Virtual probes
  - Activation tests
  - Service SLA monitoring

The Package Manager



Well-defined management of packaged applications, including:

- Install, upgrade, uninstall
- Strict versioning
- Dependencies resolution
- Isolation
- Bundle management
- Distribution across clusters

Developer Tools and SDK Content

- Dev-local multi-vendor network simulator
- Full production-grade installation in dev environments
- YANG tools including validator, compiler
- Project tooling for managing package sets

- Dev-local multi-vendor network simulator
- Build- and runtime validation of package content
- Offline-tools for validating version migration

- Self-contained and versioned package format
- Hitless package installation and version migration
- Local or remote project and package locations

Review

Automating Service Delivery

Network Abstraction

Before:

- Time-consuming, manual provisioning processes
- Days and weeks to implement new services
- Poor visibility across network during service activations

Complexity

Multi-vendor Network Orchestration

Comprehensive lifecycle service automation for hybrid networks

Virtualization Made Easy

After:

- **70% operational efficiency** increase*
- **60% reduced time** to revenue*
- Optimized service and network quality through better visibility

Simplicity

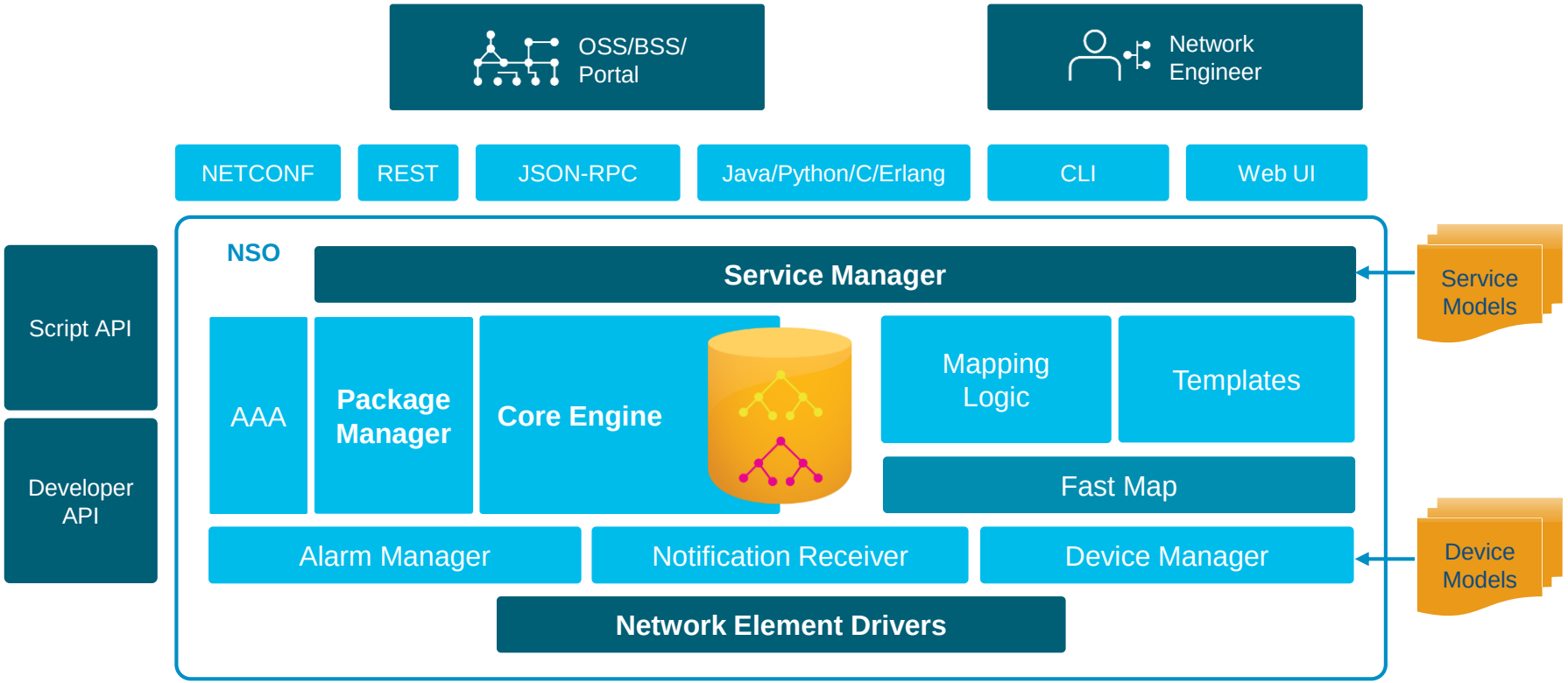
*Cisco BTA

Cisco Network Services Orchestrator (NSO) enabled by Tail-f

What you gain with NSO

- Agility throughout service lifecycle
  - Strict YANG model-driven solution
  - Auto-rendered business logic results in 90% less code
  - Effortlessly re-deployment of updated service and device models
  - DevOps for differentiation
- Full automation
- Robust and proven in tier-1 deployments
- Industry's broadest multivendor support
- Relevant in today's and tomorrow's networks

NSO Architecture Review



NSO DevNet – Key Highlights

The one place to use for sharing, finding and collaborating on NSO public knowledge!

Light start through DevNet content page and Learning-Labs

Constant news and updates to help you keep up to date

Large searchable content pool

Cisco customers, partners and employees all have access

Got a question, ask! We will help ensure a fast response

Easy to share and find public content

Code sharing through public GitHub

Reach it here: www.cisco/go/nsodevnet

Questions

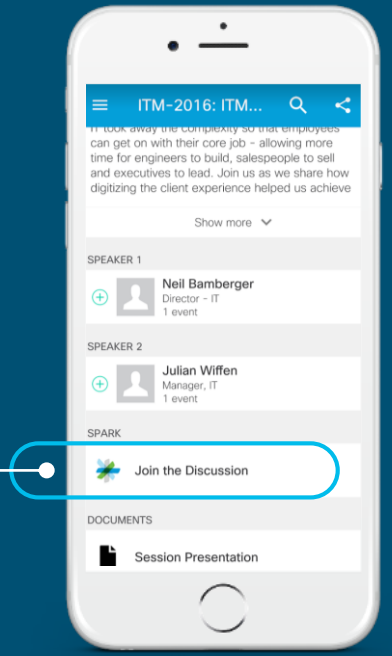
Cisco Spark

Questions?

Use Cisco Spark to communicate with the speaker after the session

How

1. Find this session in the Cisco Live Mobile App
2. Click “Join the Discussion”
3. Install Spark or go directly to the space
4. Enter messages/questions in the space

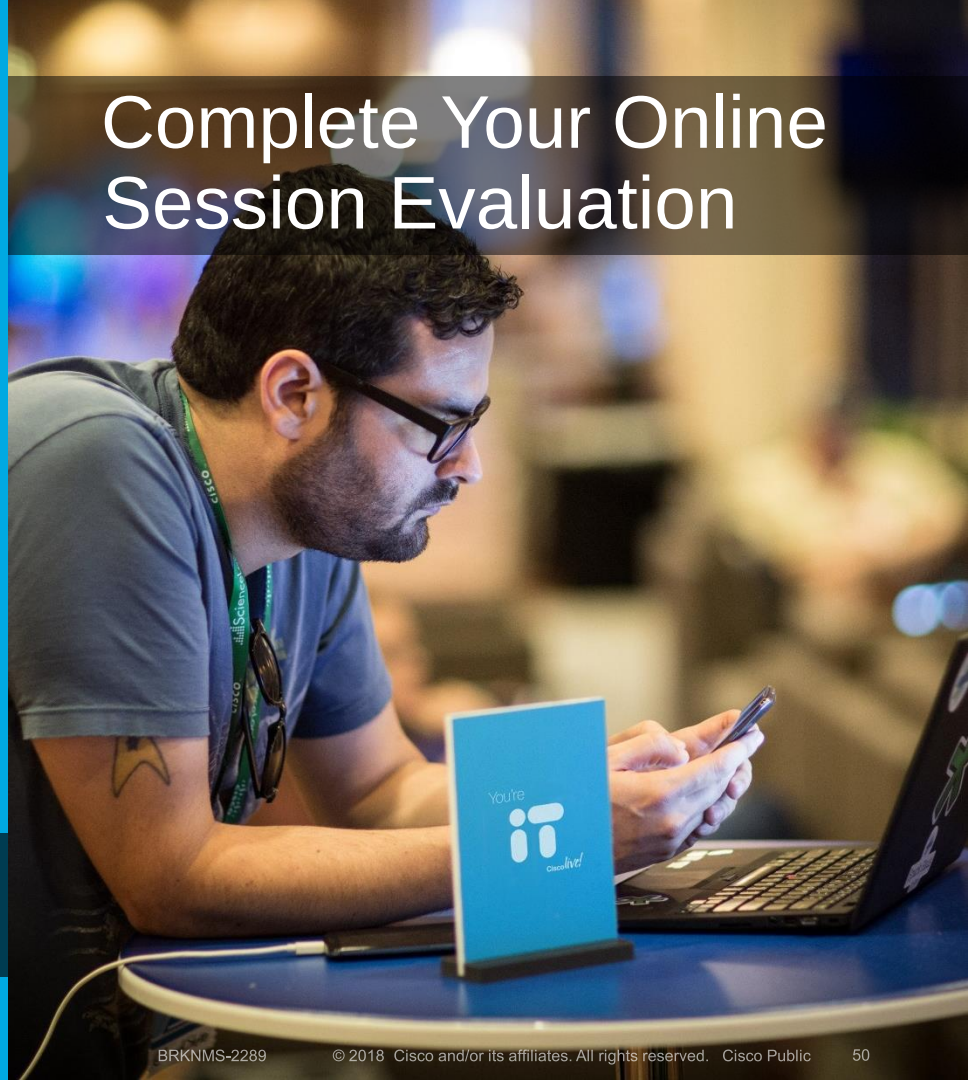


cs.co/ciscolivebot#BRKNMS-2289

Complete Your Online Session Evaluation

- Please complete your Online Session Evaluations after each session
- Complete 4 Session Evaluations & the Overall Conference Evaluation (available from Thursday) to receive your Cisco Live T-shirt
- All surveys can be completed via the Cisco Live Mobile App or the Communication Stations

Don’t forget: Cisco Live sessions will be available for viewing on-demand after the event at www.ciscolive.com/global/on-demand-library/.



Model driven automation of your **entire network**

www.cisco/go/nsodevnet

Continue Your Education

- Demos in the Cisco campus
- Walk-in Self-Paced Labs
- Tech Circle
- Meet the Engineer 1:1 meetings
- Related sessions

Thank you


You're
Cisco live!