Fastmap Templates in Cisco NSO
Rapidly developing a multi-vendor service

Michael Maddern
Use Case

MEF ELAN
Multi-vendor PE configuration (including UNI)
  • Cisco IOS L2VPN
  • Juniper Junos VPLS
  • Alcatel-Lucent VPLS

CE configuration can be added later
NSO template-only service (no code)
Create / update / migrate / repair and delete all supported
ELAN

Multipoint-to-Multipoint EVC

Carrier Ethernet Network
NSO Multi-Vendor ELAN Service
NSO Quick Overview
System Overview

- Model-driven, end-to-end service lifecycle and customer experience focused
- 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 network-wide, centralized policy and services
Configuration Datastore

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 Engineering
- Service Manager
- Device Manager
- Device Abstraction

Ops and Provisioning
- Package Manager
- ESC (VNFM)
- VNF Lifecycle Manager
- VNF Service Monitoring

Service Developers

Multi-domain Networks

© 2018 Cisco and/or its affiliates. All rights reserved.
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
Industry’s Broadest Multivendor Support
Over 100 Supported NEDs – Customization Available
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

Full service lifecycle management

© 2018 Cisco and/or its affiliates. All rights reserved.
The ability to dry-run all operations is key for trust

- **Create**
  - Easy
  - Given a set of service-level inputs, provide a known and valid output to network
  - May require some additional resource collection to fulfill the configuration set

- **Update**
  - Challenging
  - Allow arbitrary changes to the network service
  - May require collecting or handing back resources to fulfill configuration set

- **Delete**
  - Hard
  - Delete any given instance of a service and clean up the resources
  - May require reference counting for shared resources
Stateful Convergence

Decomposition logic:
- Service templates for one-to-one mapping
- Java or Python for programmatic mapping (side effects, etc.)
- ...or a combination of both

Only create operation needs to be declared
Update, delete inferred

Network Services Orchestrator (NSO)
- Network Engineering
- Ops and Provisioning
- Service Developers

- Service Manager
- Device Manager
- Device Abstraction
- ESC (VNFM)
- Package Manager
- VNF Lifecycle Manager
- VNF Service Monitoring

Multi-domain Networks

© 2018 Cisco and/or its affiliates. All rights reserved.
Templates Overview
Using Service Templates

Service Manager
- Service model
  - Commit
  - Template reference
- Service Template
  - Parameter substitution
  - Multi-vendor

Device Manager
- Device models
  - Merge into device models
  - Calculate minimum diff
- Config Diff
  - Generate native configs from device diffs
- Transactional guarantee

NED
- Device native config

© 2018 Cisco and/or its affiliates. All rights reserved.
Creating Service Templates

- **Device**
  - Create service manually (via CLI)
  - Device config

- **NED**
  - Sync device into NSO

- **Device Manager**
  - Device models
  - Extract into XML document

- **Service Manager**
  - Service Template
  - Identify substitution parameters
  - Create service model

- **Create Service Template**

© 2018 Cisco and/or its affiliates. All rights reserved.
Demo
# Demo Devices

<table>
<thead>
<tr>
<th>Device Type</th>
<th>IP Address</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS</td>
<td>172.10.10.x</td>
<td></td>
</tr>
<tr>
<td>ios0</td>
<td>172.10.10.1</td>
<td></td>
</tr>
<tr>
<td>ios1</td>
<td>172.10.10.2</td>
<td></td>
</tr>
<tr>
<td>ios2</td>
<td>172.10.10.3</td>
<td></td>
</tr>
<tr>
<td>Juniper Junos</td>
<td>172.10.20.x</td>
<td></td>
</tr>
<tr>
<td>junos0</td>
<td>172.10.20.1</td>
<td></td>
</tr>
<tr>
<td>junos1</td>
<td>172.10.20.2</td>
<td></td>
</tr>
<tr>
<td>junos2</td>
<td>172.10.20.3</td>
<td></td>
</tr>
<tr>
<td>ALU SR</td>
<td>172.10.30.x</td>
<td></td>
</tr>
<tr>
<td>alu0</td>
<td>172.10.30.1</td>
<td></td>
</tr>
<tr>
<td>alu1</td>
<td>172.10.30.2</td>
<td></td>
</tr>
<tr>
<td>alu2</td>
<td>172.10.30.3</td>
<td></td>
</tr>
<tr>
<td>Existing ELAN Service</td>
<td>172.10.30.x</td>
<td></td>
</tr>
</tbody>
</table>
Demo Summary

- New ELAN service developed from scratch
- Multiple ELAN services instantiated and updated
- Sites added and removed cleanly (all neighbour sites automatically updated)
- Device migrations tested (including across vendor)
- Broken service repaired (for example, when loopback address is changed)
- Web UI automatically generated (and used to provision services)
- NBI (REST) automatically generated (and tested via Postman)
- All device configuration tracked and removed when service deleted
Questions
NSO Digital Ecosystem*

External (Open for all) [www.cisco.com/go/nsodevnet](http://www.cisco.com/go/nsodevnet)

Public material targeting partners and customers

- Selected content
- Training material
- Shared code
- Community and main repository of content and Q&A

*Customer, Partner, and Cisco **Cisco internal only

Free NSO evaluation download!