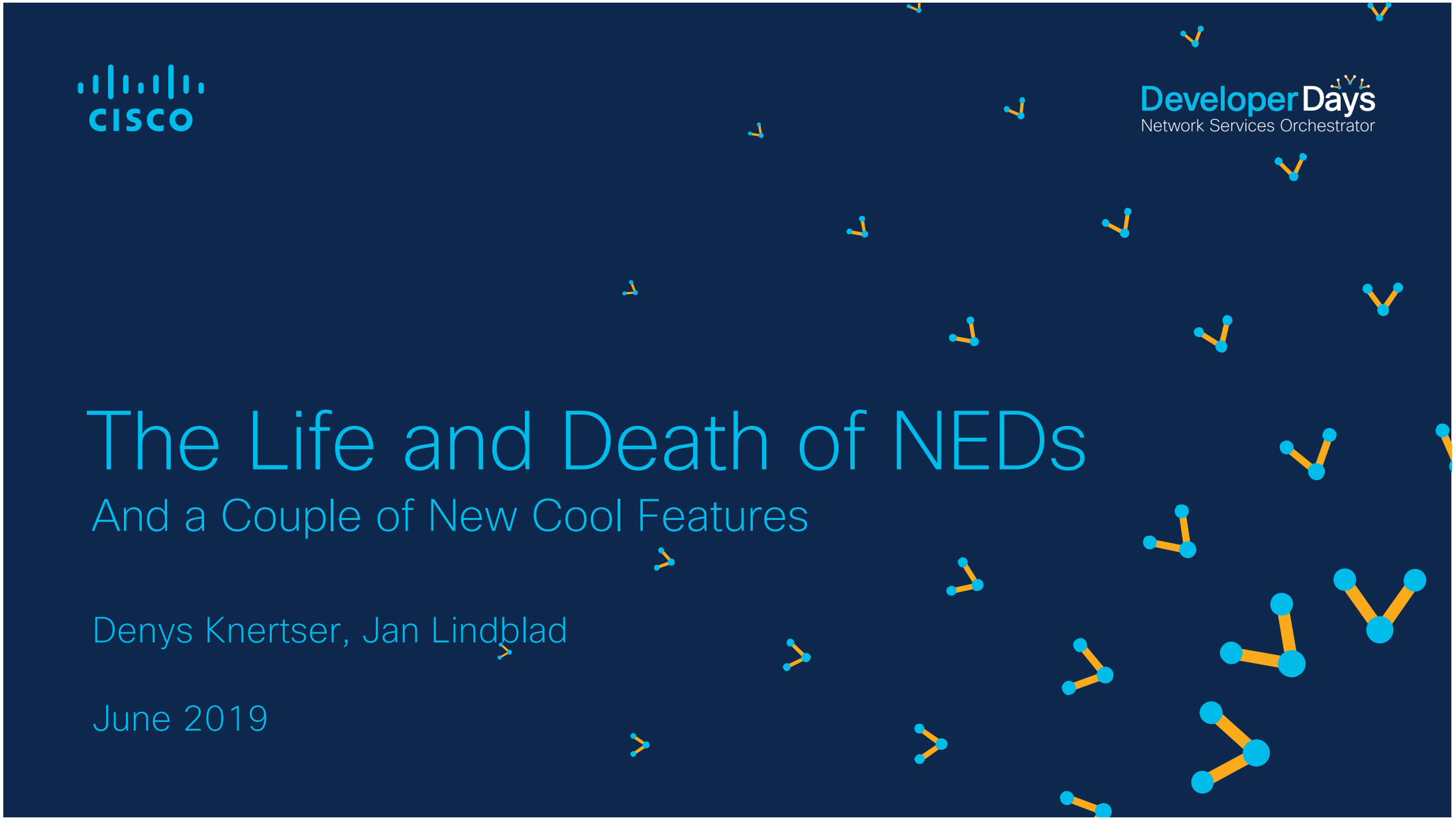


The Life and Death of NEDs

And a Couple of New Cool Features

Denys Knertser, Jan Lindblad

June 2019

The Life and Death of NEDs (in an NSO 5 world)

The NSO 5 NED Life Cycle

- Creating a NED
  - NETCONF NED Builder Demo
- Installing a NED
- Testing a NED
- Upgrading a NED
- Device NED Migration
- Retiring a NED

New Cool Features

- New WebUI views
- Service Progress Monitoring
- Nano Services
- NETCONF Call Home
- LSA NSO YANG View
- LSA Remote Kickers

Creating a New NED

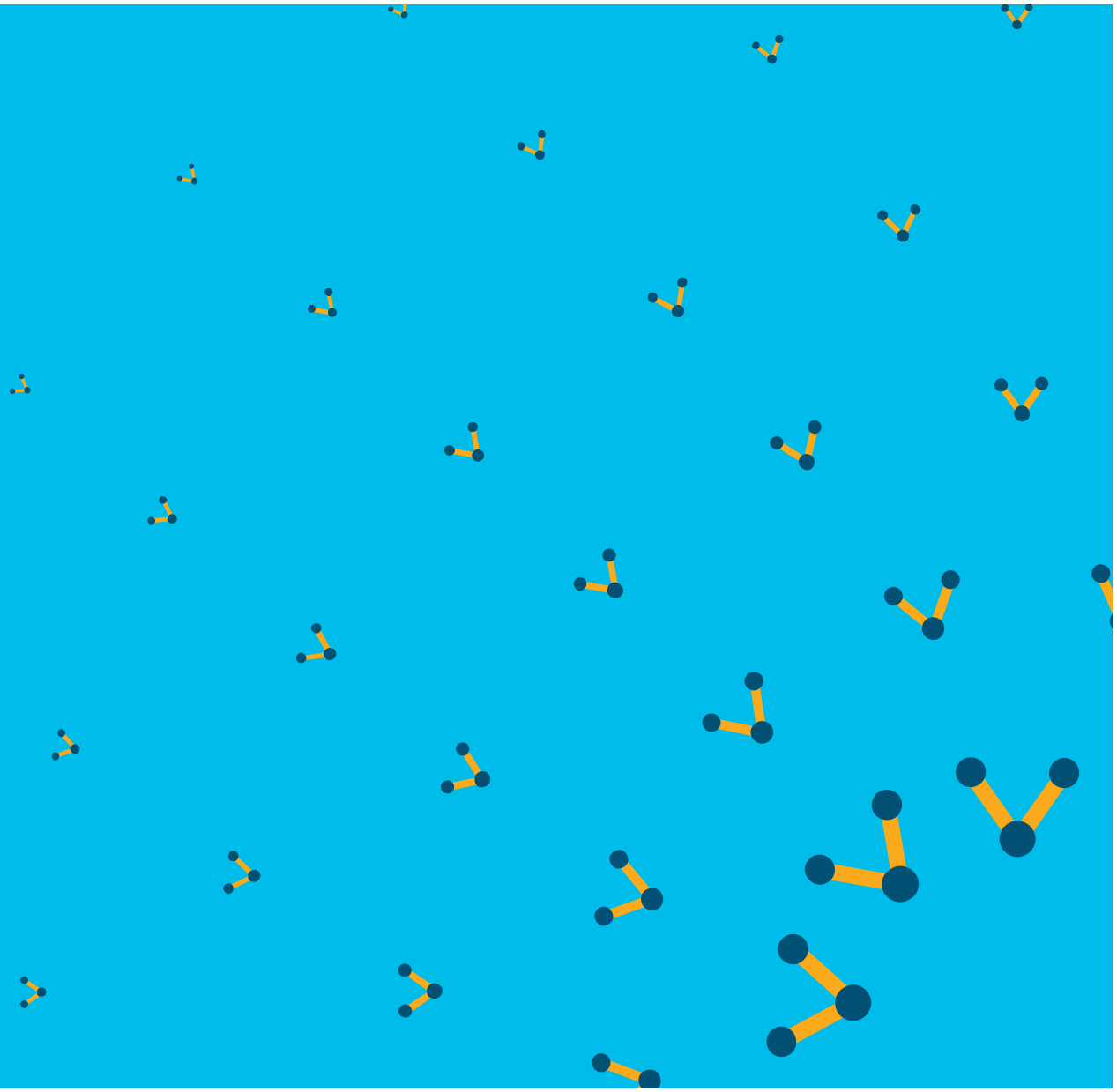
Buy it

- from Cisco
- from Device Vendor?
- from 3rd party, e.g. consultant

Make it yourself

- ncs-make-package
- Pioneer
- NETCONF NED Builder

NETCONF NED Builder Demo



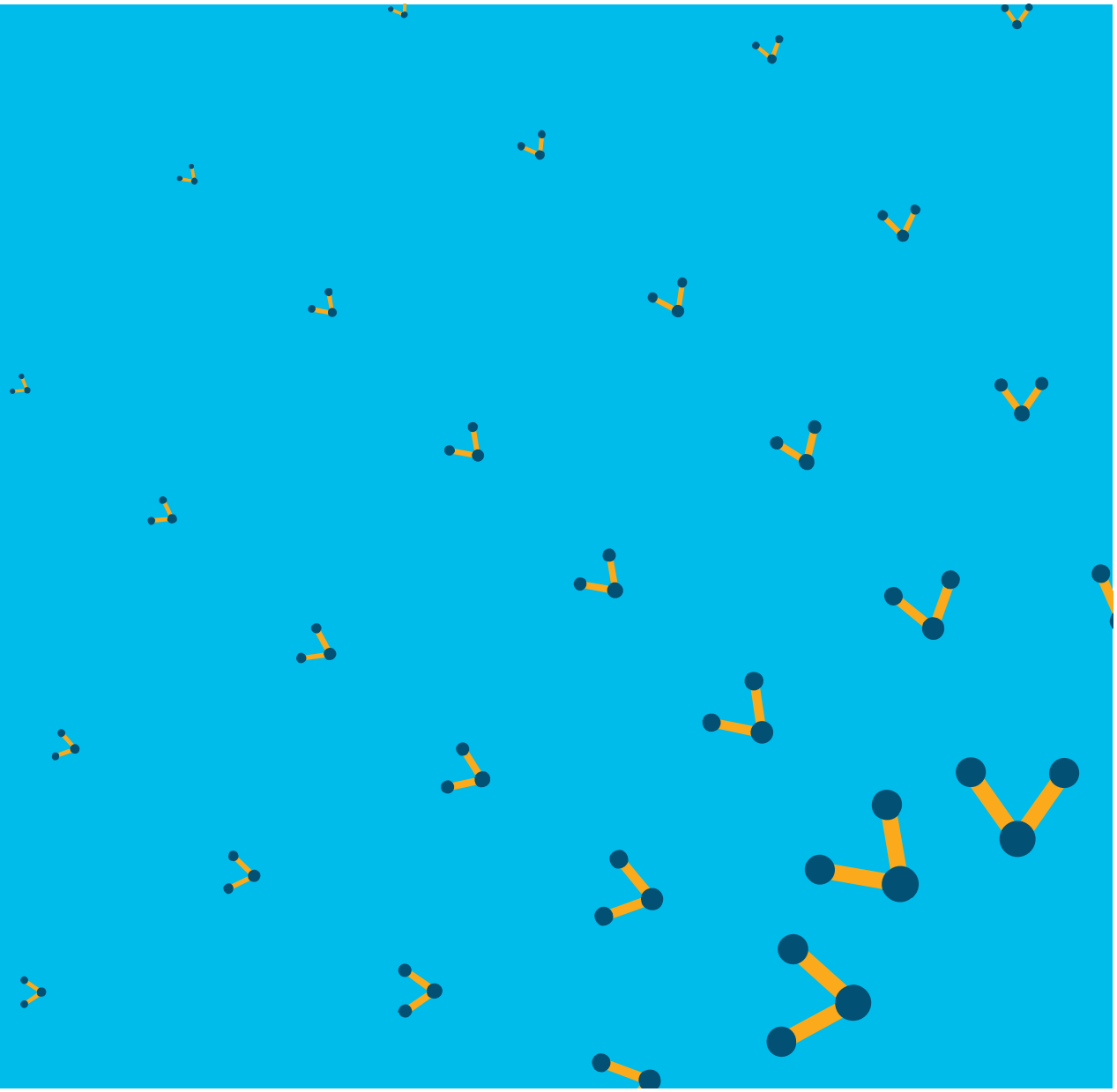
NETCONF NED Builder: Background

- NETCONF devices require a NED
  - no code, only YANG models
- Devices that implement RFC 6022 YANG Module for NETCONF Monitoring provide their own YANG models
- NETCONF NED builder helps build a NED from an RFC 6022-enabled NETCONF device
- Successor to the Pioneer package

NETCONF NED Builder: Workflow

- Configure a reference device
  - Use ned-id netconf
- Configure a netconf-ned-builder project
- Run fetch-module-list action
- Select necessary modules
- Build the NED with build-ned action
- Make a tarball with export-ned action

NED Lifecycle



Creating a New NED

Buy it

- from Cisco
- from Device Vendor?
- from 3rd party, e.g. consultant

Make it yourself

- ncs-make-package
- Pioneer
- NETCONF NED Builder

Installing a NED

NSO 4.x

- Each device has one NED
- NSO uses a single version of any given YANG module
- All services have to agree on the NED version to support for devices they care about

NSO 5.x

- Each device has one NED
- Each NSO NED can use a different version of any given device YANG module
- All services have to agree on the NED version**s** to support for devices they care about

Installing a NED – Many NED Variants

+

- Sometimes devices are upgraded and have new YANG which is incompatible with earlier versions.

  If so, NSO 5.x makes it possible to migrate devices individually to use new NED.

–

- More NED variants means more work for service programmers. Typically more templates that needs development and test.
- More NED variants take up more memory.

Installing a NED – Service Template

NSO 4.x

```
<config>
  <sys xmlns="http://example.com/router">
    <syslog>
      <server foreach="{/server}">
        ...
        <pid/>
        ...
      </server>
    </syslog>
  </sys>
</config>
```

NSO 5.x

```
<config>
  <?if-ned-id router-nc-1.0:router-nc-1.0?>
  <sys xmlns="http://example.com/router">
    <syslog>
      <server foreach="{/server}">
        ...
        <pid/>
        ...
      </server>
    </syslog>
  </sys>
  <?elif-ned-id router-nc-1.1:router-nc-1.1?>
  <sys xmlns="http://example.com/router">
    <syslog>
      <server foreach="{/server}">
        ...
        <option>pid</option>
        ...
      </server>
    </syslog>
  </sys>
```

Testing a NED

Tools

- Traditional test tools, e.g. Lux, Expect, JUnit, pytest
- DrNED Xmnr (Doctor NED Examiner) https://github.com/NSO-developer/drned-xmnr
- Sync-from verbose
- Load-native-config verbose

Publishing Results

- NSO Interop Lab
- EANTC

DrNED Xmnr – Use Case Transactionality Test

- Setup NSO with device & NED
- Config a handful of typical configs for the given use case, e.g. using device CLI
- Use DrNED Xmnr to record the state for each typical config
- When states are recorded, use DrNED Xmnr to test the transitions to and from each state.

DrNED Xmnr reports

- Any failing transitions
- Any unexpected config differences (lost & autoconfig)
- Coverage metrics for the test configs

Sync-from verbose

```
admin@ncs(config)# devices device alu7750-3 sync-from verbose
result true
info
  Number of lines parsed : 255
  Number of lines skipped : 12
  Skipped 4 lines in context '/service/ies/subscriber-interface/group-interface/ipv6/dhcp6' :
   (line 203) : ' option'
   (line 204) : ' interface-id ascii-tuple'
   (line 205) : ' remote-id'
   (line 206) : ' exit'
  Skipped 4 lines in context '/service/ies/subscriber-interface/group-interface/ipv6/dhcp6/relay' :
   (line 208) : ' source-address 2a01:c500:4000::610:1:a1d'
   (line 209) : ' link-address 2a01:c500:4000::610:1:a1d'
   (line 210) : ' server 2a01:c000:14::1'
   (line 211) : ' server 2a01:c000:14:1::1'
  Skipped 4 lines in context '/service/ies/subscriber-interface/group-interface/ipv6/router-advertisements' :
   (line 197) : ' other-stateful-configuration'
   (line 198) : ' prefix-options'
   (line 199) : ' autonomous'
   (line 200) : ' exit'
```

Load-native-config verbose

```
admin@ncs(config-device-alu7750-3)# load-native-config file /home/myuser/test/drned/ALUSR-665.cfg verbose
info
  Number of lines parsed           :  14767
  Number of lines skipped          :     39
  Skipped 8 lines in context '/port/ethernet/access' :
   (line 2916) : '                ingress'
   (line 2918) : '                    queue-group "ing-QG-FTTHcorporate" '
   (line 2920) : '                    exit'
   (line 2922) : '                exit'
   (line 3086) : '                ingress'
   (line 3088) : '                    queue-group "ing-QG-FTTHcorporate" '
   (line 3090) : '                    exit'
   (line 3092) : '                exit'
  Skipped 1 line in context '/python/python-policy' :
   (line 10968) : '            dhcp6 relay-forward direction egress script "vula"'
  Skipped 2 lines in context '/service/vpls/sap' :
   (line 12747) : '                dhcp6-python-policy "VULA"'
   (line 12779) : '                dhcp6-python-policy "VULA"'
  Skipped 9 lines in context '/service/vprn' :
   (line 12833) : '            snmp'
   (line 12835) : '                community "1pxGTwTfXgDg37MGj.vnhk" hash2 rw version v2c'
   (line 12837) : '            exit'
…
```

Upgrading a NED

Changing the contents (YANG) of a NED, while keeping the same ned-id

- This is what you have been doing every time you changed a NED in NSO 4.x
- All devices using this NED will change their YANG at the same time
- NSO handles the YANG upgrade for all the data in the database

To change the NED NSO uses for a particular device (to new ned-id) is not an upgrade. That is called Device NED Migration.

Device NED Migration (new ned-id)

examples.ncs/getting-started/developing-with-ncs/26-ned-migration:

```
admin@ncs% request devices device ex0 migrate new-ned-id router-nc-1.1 verbose
modified-path {
    path /devices/device[name='ex0']/config/r:sys/syslog/server/selector/option/pid
    info sub-tree has been deleted
}
modified-path {
    path /devices/device[name='ex0']/config/r:sys/syslog/server/selector/option
    info node type has changed from non-presence container to leaf-list
}
affected-services-with-changes [ /services/sls:syslog ]
affected-services [ /services/sls:syslog ]
```

- Migration drops service meta data, so re-deploy reconcile needed:

```
admin@ncs% request services syslog re-deploy reconcile { discard-non-service-config }
```

Yanger **diff** Plugin + Model Upgrade Analysis

Not part of any release. Do you need this? Let us know!

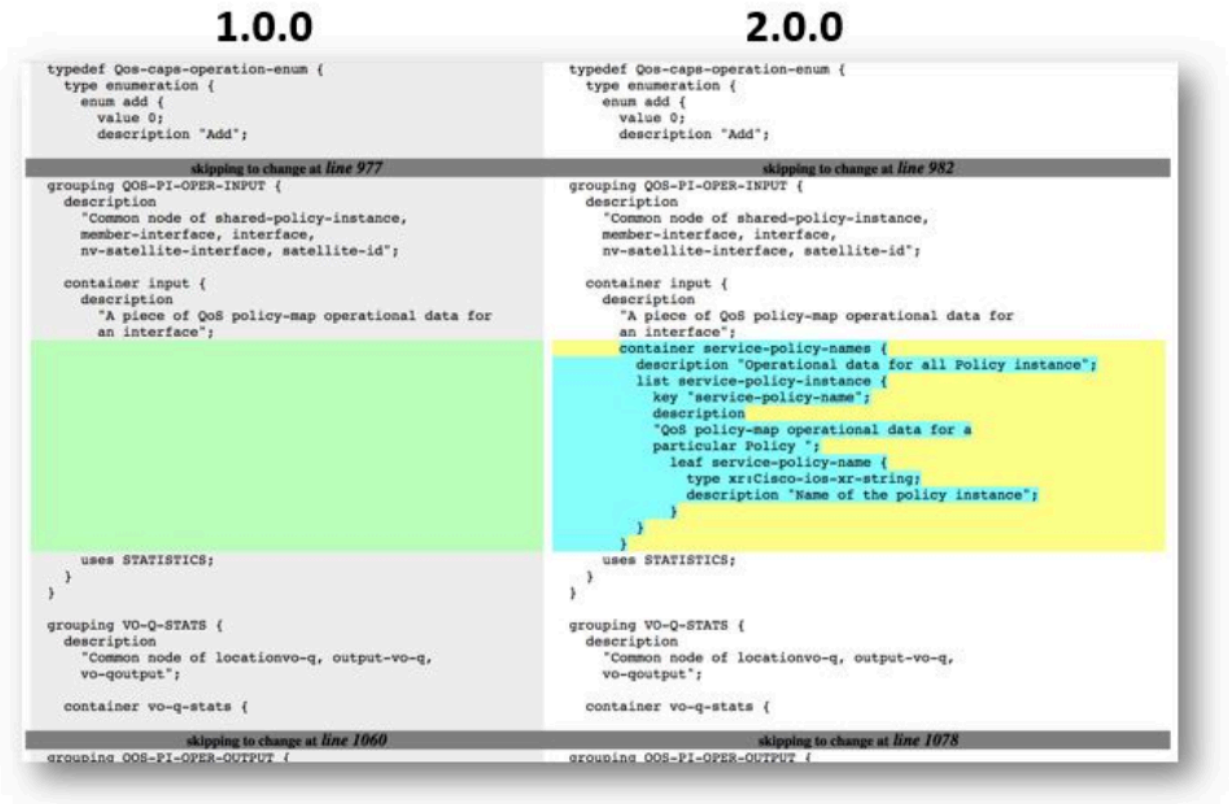
```
yanger -W none -P ../drned/yanger/plugins/confd-6.7/ -p src/yang              \
  -f diff --diff-json --diff-keep-ns --diff-skip-choice                       \
  test/7.7.7/src/yang/tailf-ned-cisco-ios-xr.yang                             \
  test/7.10.1/src/yang/tailf-ned-cisco-ios-xr.yang                            \
  -o iosxr_diff.json


yanger -W none -P ../drned/yanger/plugins/confd-6.7/ -p ./src/yang            \
  -f diff --diff-skip-choice                                                  \
  --diff-left=test/5.6.6/src/yang/tailf-ned-cisco-ios.yang                    \
              test/6.19/src/yang/tailf-ned-cisco-ios.yang                     \
  --diff-include=/router/bgp --diff-exclude=/router/bgp/address-family        \
  --diff-include=/router/ospf --diff-incompatible


./model_upgrade_analysis.py ios_diff.json /tmp/cdb.xml
```

Use this for Upgrade or Migration? Your decision.

Yang Suite @ Yang Catalog

Yang Suite can perform version compatibility analysis of YANG modules

- Easiest way to launch Yang Suite is by surfing to yangcatalog.org, search for your module, then click Module Details and Yang Suite



Retiring a NED

Imagine you have 6 variants of the IOS NED in your NSO system.
You want to retire one variant (probably) no longer in use.

How do you know for sure? Try this:

```
# show running-config devices device device-type netconf ned-id router-nc-1.0 | select
device-type netconf ned-id

devices device ex0
 device-type netconf ned-id router-nc-1.0
```

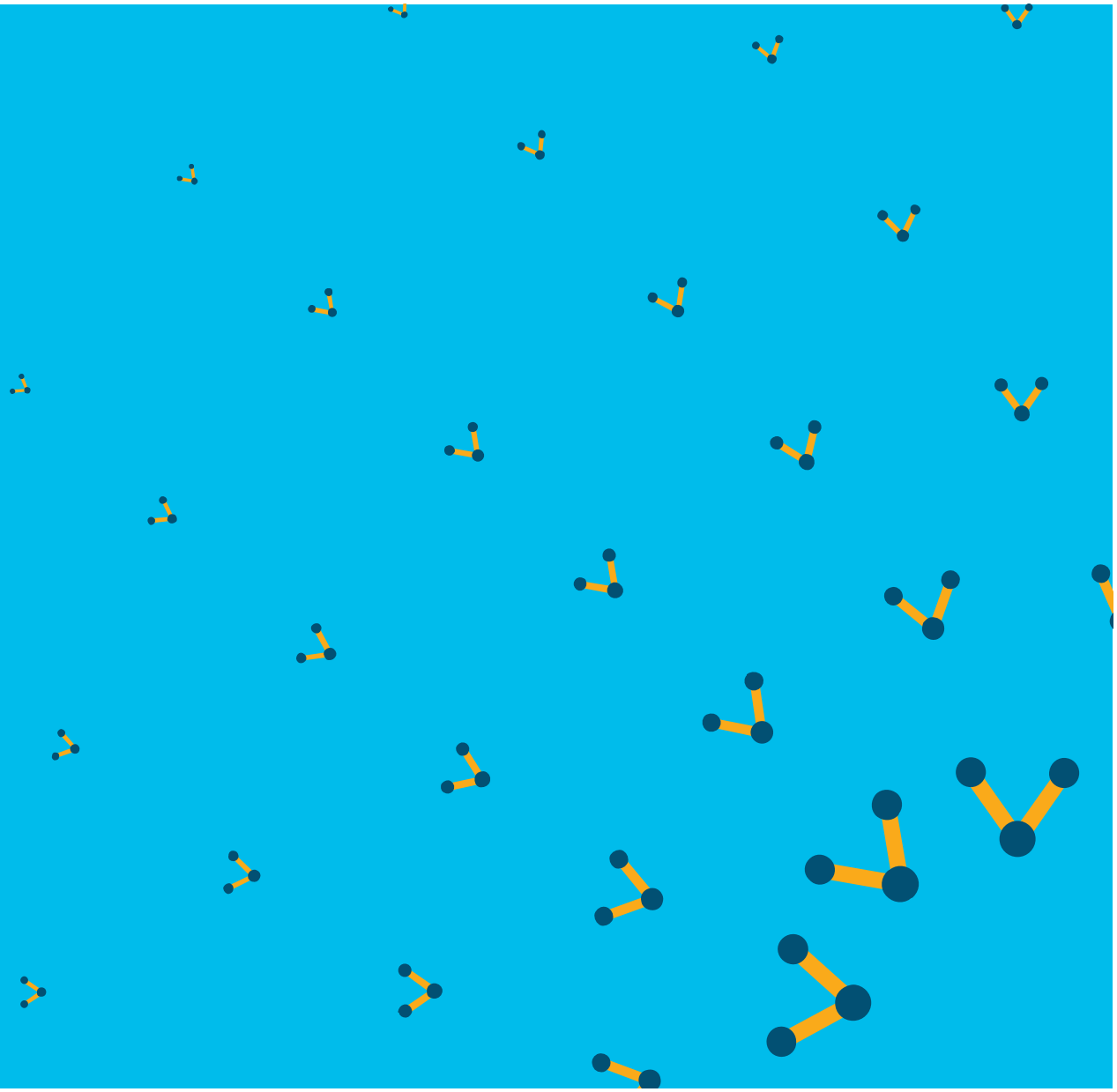
-or-

```
% No entries found.
```

Unfortunately, this is not the complete truth:

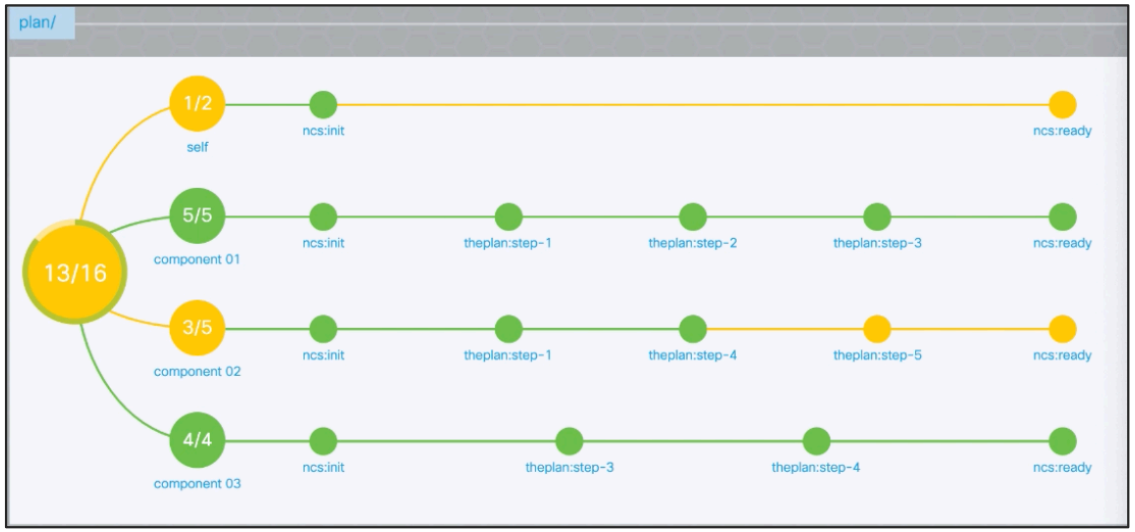
- This works if you have re-deployed all your services
- An action to check if a NED is still referenced is coming in a future release

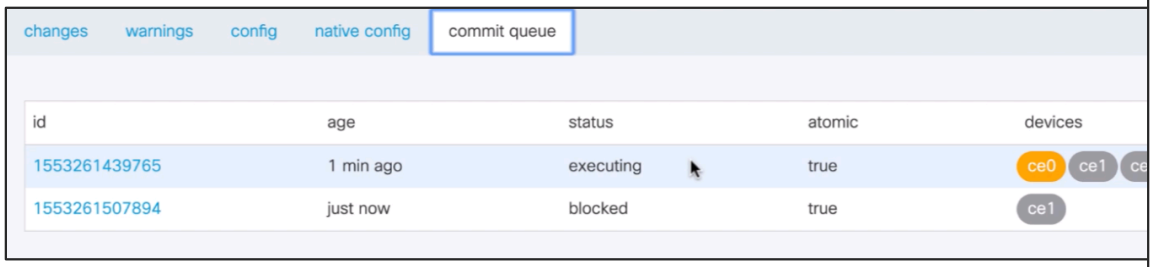
New Feature Highlights

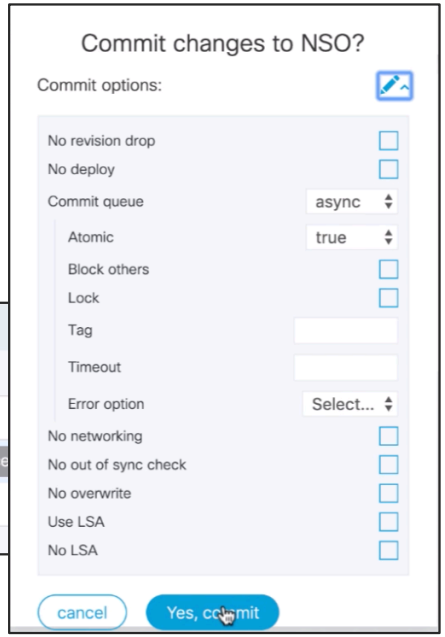


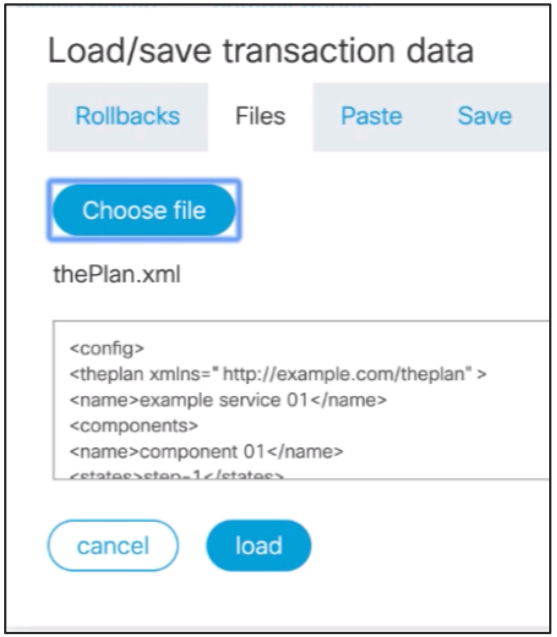
New WebUI Features

- Service Plan Visualization
- Loading/saving files, pasting configs
- Commit queue and detailed commit flag control









Service Progress Monitoring

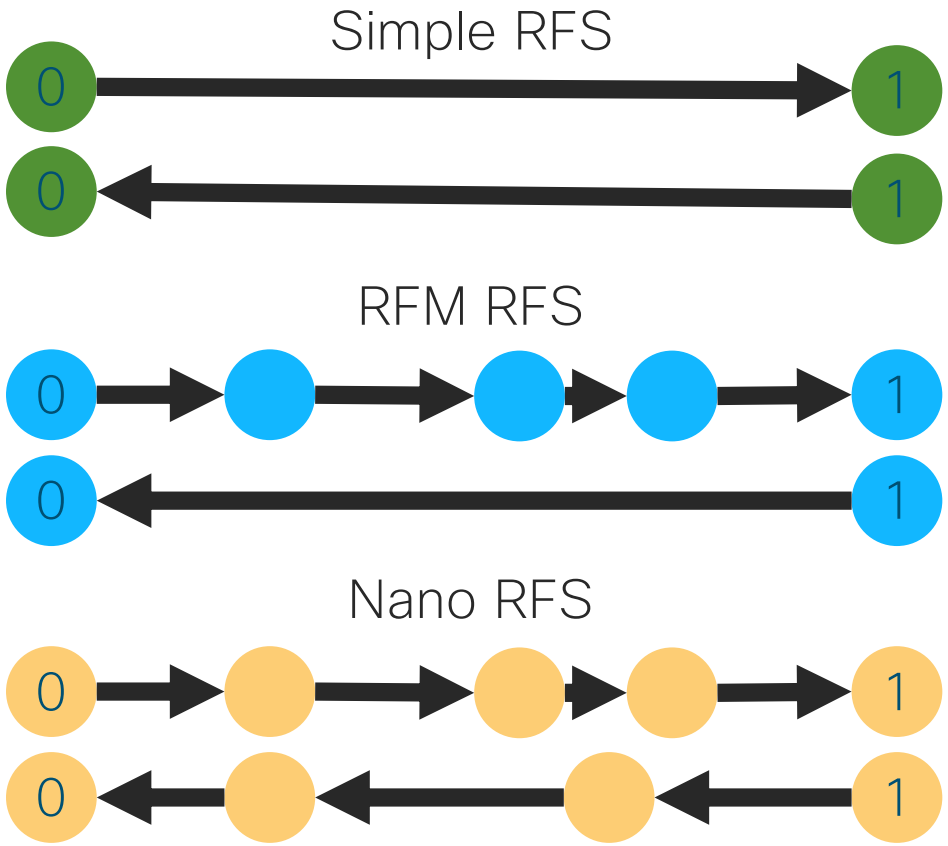
```
uses ncs:service-progress-monitoring-data;
uses ncs:service-progress-monitoring-trigger-action
  refine timeout { tailf:actionpoint myserv-timeout-point; }
```

```
                                                                JEOPARDY                      VIOLATION          SUCCESS
NAME  POLICY         START TIME           JEOPARDY TIME        RESULT    VIOLATION TIME      RESULT     STATUS  TIME
-------------------------------------------------------------------------------------------------------------------
self  service-ready  2018-05-08T13:14:18  2018-05-08T13:24:18  -         2018-05-08T13:34:18  -          running  -

[ok][2018-05-08 13:19:01]
admin@ncs% run show myserv m1 plan
NAME     TYPE    STATE                STATUS       WHEN                 ref
-------------------------------------------------------------------------
self     self    init                 reached      2018-05-08T13:14:18  -
                 ready                not-reached  -                    -
router   router  init                 reached      2018-05-08T13:14:18  -
                 syslog-initialized   reached      2018-05-08T13:14:20  -
                 ntp-initialized      reached      2018-05-08T13:14:20  -
                 dns-initialized      reached      2018-05-08T13:14:27  -
                 ready                reached      2018-05-08T13:14:27  -
router2  router  init                 reached      2018-05-08T13:14:18  -
                 ready                not-reached  -                    -

[ok][2018-05-08 13:19:13]
admin@ncs% run show myserv m1 service-progress-monitoring
                                                                JEOPARDY                      VIOLATION
NAME  POLICY         START TIME           JEOPARDY TIME        RESULT    VIOLATION TIME      RESULT     STATUS      SUCCESS TIME
-------------------------------------------------------------------------------------------------------------------------------
self  service-ready  2018-05-08T13:14:18  2018-05-08T13:24:18  passed    2018-05-08T13:34:18  passed     successful  2018-05-08T13:19:10
```

Nano Services

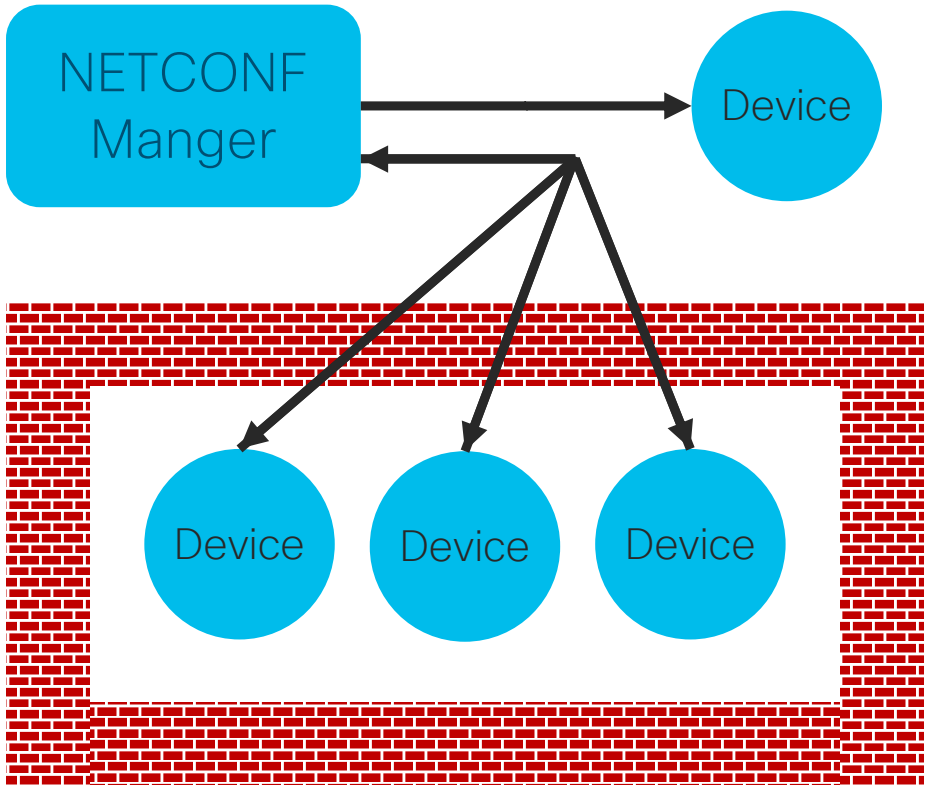
- Nano Services have been experimental for a long time
- Alternative to RFM design pattern
- Useful for services with the most complex plans, esp. with multiple delete stages

RFS = Resource Facing Service
RFM = Reactive FastMap (design pattern)



NETCONF Call Home

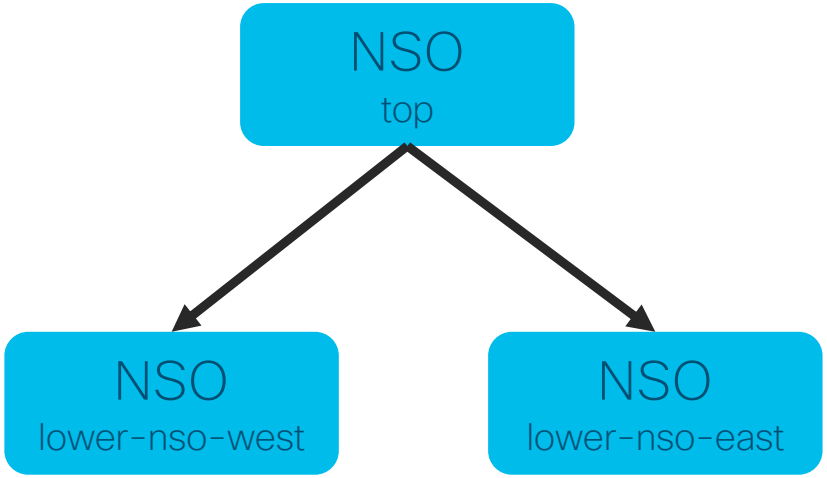
- NSO now supports the new NETCONF Call Home RFC 8071
- Useful when managing devices behind firewalls



LSA: Seeing the Lower NSO YANG Tree

Top LSA Node can now see the lower node's NSO YANG tree

- Access /devices/device
- Configure kickers
- Read Alarms, Notifications

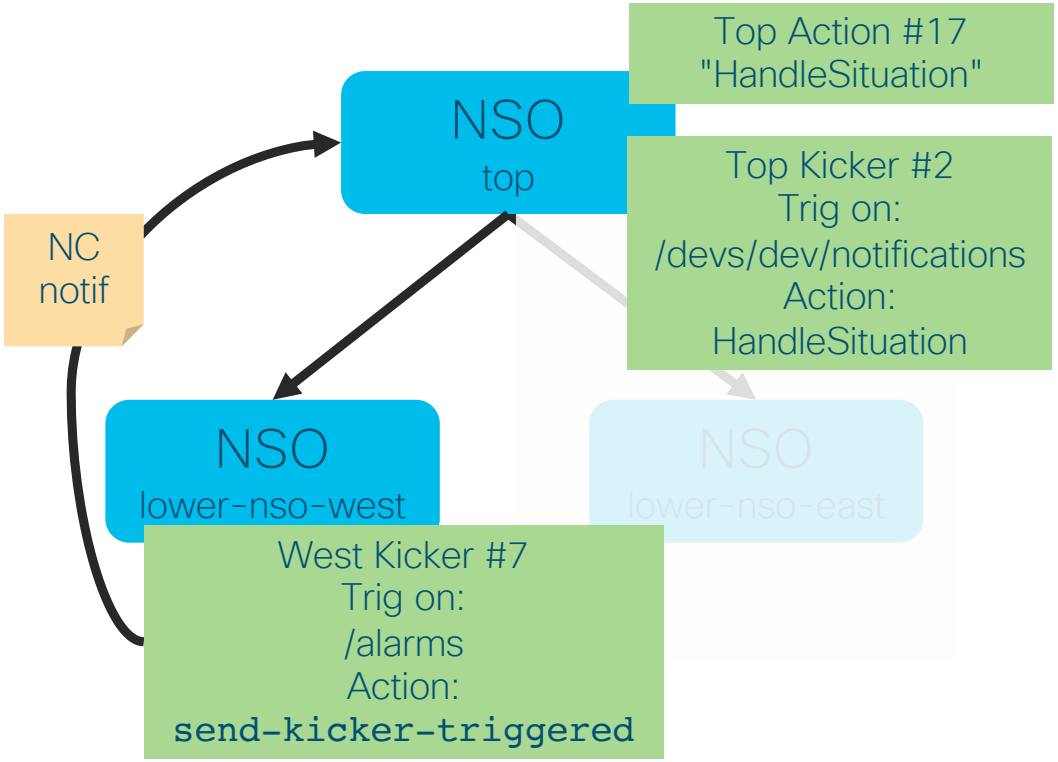



```
devices device lower-nso-west config ncs:devices device device west-ios-47 sync-from
```

LSA: Remote Kickers

Kickers can only react to events happening at the local node.

- Now a kicker can execute a built-in action that sends a NETCONF notification to the upper node.
- Another kicker on the upper node can react to this notification, and invoke an action on the top node.




CISCO