YANG Data Modeling and NETCONF: Cisco and Industry Developments

Benoit Claise, Distinguished Engineer
Carl Moberg, Technology Director
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-2032
Session Abstract

• This session covers the many aspects of using data modeling-driven management as a fundamental approach across network orchestration and automation

• We will help you understand latest industry trends and customer observations around how configuration and data collection and tasks are rapidly changing. The focus is on learning about required technologies including the YANG data modeling language and related protocols like NETCONF and RESTCONF for configuration and data collection

• This last year has seen an unprecedented uptake of the YANG language across various aspects of the networking industry. We are going to provide an overview of the many activities in standards organizations including what is going on in the IETF, IEEE, MEF, and various opensource projects
Session Abstract (cont’d)

• Finally, your orchestration and automation is only as strong as your toolchain so we will guide you some examples out of a growing set of supporting tools. This session includes several small demos, using scripting basics, showcasing the power and usefulness of combining data models and software for network programmability purposes
  • The examples are available on GitHub

• Audience: Network engineers, IT engineers and software developers tasked with automating, operating, and troubleshooting networks and network services. This sessions applies to both enterprise and service provider networking environments. Attendees should be familiar with network management and scripting basics
The Content of This Session Is…

• Not about
  • A detailed training on YANG
  • A detailed training on NETCONF

• About
  • The latest industry trends
  • Provide guidance towards data modeling-driven management
  • Convincing you: YANG is key for you
  • The Cisco strategy
  • Level 2 session: so not only marketing 😊
  • Pointers

• Assuming …
  • That you know about the basics of Network Management
Agenda

- Introduction
- What are NETCONF, RESTCONF, and YANG
- Demo
- Data Modelling, Standard, and Opensource
- Programming the Cisco Networks
- Cisco Direction
- Where to Start?
- Conclusion
Historical Market Leaders in Network Automation
Where Network Automation Needs To Go

- Networking is *well known*, we can build *stable abstractions*
- Modern software practices gives us *reusability*

Integration Services 60%
In-house Development 30%
Software 10%
In-house Development 10%
Software 90%
An Origin Story

How NETCONF, YANG and RESTCONF Got Started

• Show of hands at Internet Engineering Task Force (IETF) pre-meeting in 2000

• Many meetings at events in 2001
  • Operators expressing opinion that the developments in IETF do not really address requirements regarding configuration management.

• June of 2002, the Internet Architecture Board (IAB) held invitational workshop on Network Management (RFC3535) to
  • Identify a list of technologies relevant for network management with their strengths and weaknesses
  • Identify the most important operations needs.
Best Practices Coming Together

NETCONF, RESTCONF and YANG

SNMP Experience

Operations Requirements

CLI Best Practices
The Outcome

The goal of the workshop was to continue the important dialog started between network operators and protocol developers, and to guide the IETFs focus on future work regarding network management.

RFC3535 - Overview of the 2002 IAB Network Management Workshop
Timeline of Specifications

<table>
<thead>
<tr>
<th>Year</th>
<th>Specifications</th>
</tr>
</thead>
<tbody>
<tr>
<td>'10</td>
<td>YANG 1.0, RFC6020, October 2010</td>
</tr>
<tr>
<td></td>
<td>Common YANG Data Types, RFC6991, July 2013</td>
</tr>
<tr>
<td></td>
<td>Interface and IP Modules, RFC7223, RFC7277, May, June 2014</td>
</tr>
<tr>
<td>'11</td>
<td>NETCONF 1.1, RFC6241, June 2011</td>
</tr>
<tr>
<td>'12</td>
<td>NETCONF Access Control, RFC6536, March 2012</td>
</tr>
<tr>
<td>'13</td>
<td>YANG Patch Media Type, RFC8072, February 2017</td>
</tr>
<tr>
<td>'14</td>
<td>RESTCONF Protocol, RFC8040, January 2017</td>
</tr>
<tr>
<td>'15</td>
<td>JSON Encoding, RFC7951, August 2016</td>
</tr>
<tr>
<td>'16</td>
<td>Routing Management, RFC8022, November 2016</td>
</tr>
<tr>
<td>'17</td>
<td>NETCONF Call Home and RESTCONF Call Home, RFC8071, February 2017</td>
</tr>
<tr>
<td></td>
<td>NETCONF over TLS + x.509, RFC7589, October 2016</td>
</tr>
</tbody>
</table>
Beware Confusion on Information vs Data Models

The main purpose of an IM is to model managed objects at a conceptual level, independent of any specific implementations or protocols used to transport the data. [...]

DMs, conversely, are defined at a lower level of abstraction and include many details. They are intended for implementors and include protocol-specific constructs.

-- RFC3444 On the Difference between Information Models and Data Models
Layering Model

**Layer**
- Content
- Operations
- Messages
- Secure Transport

**NETCONF**
- Configuration data
- Notification data
- `<get>`
- `<get-config>`
- `<rpc>`
- `<notification>`
- SSH

**RESTCONF**
- Configuration data
- Notification data
- GET, POST, PATCH
- HTTP Payload
- HTTPS
- W3C Server-Sent Events
Agenda

- Introduction
- What are NETCONF, RESTCONF and YANG
  - Demo
- Data Modelling, Standard, and Opensource
- Programming the Cisco Networks
- Cisco Direction
- Where to Start?
- Conclusion
The NETCONF Protocol
The NETCONF Protocol

• IETF network management protocol
• Distinction between configuration and state data
• Multiple configuration data stores (candidate, running, startup)
• Configuration change validations and transactions
• Selective data retrieval with filtering
• Streaming and playback of event notifications

Why you should care:
NETCONF provides fundamental programming features for comfortable and robust automation of network services
NETCONF Base Operations

Data Manipulation
<get>
<get-config>
<edit-config>
<copy-config>
<delete-config>

Locking
<lock>
<unlock>

Session Management
<close-session>
<kill-session>
NETCONF Capabilities Exchange (Legacy)

Capabilities advertised by server and client at start of session

```xml
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <capabilities>
    <capability>urn:ietf:params:netconf:base:1.1</capability>
    <capability>urn:ietf:params:netconf:capability:startup:1.0</capability>
    <capability>urn:ietf:params:xml:ns:yang:ietf-interfaces</capability>
  </capabilities>
  <session-id>4</session-id>
</hello>
```
NETCONF Protocol Capabilities

```xml
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <capabilities>
    <capability>urn:ietf:params:netconf:base:1.1</capability>
    <capability>urn:ietf:params:netconf:capability:startup:1.0</capability>
    <capability>urn:ietf:params:xml:ns:yang:ietf-interfaces</capability>
  </capabilities>
  <session-id>4</session-id>
</hello>
```
NETCONF Datastore Capabilities

Additional operations and content supported on a device

Candidate

Running

Startup

:candidate

:startup

<edit-config>

<get-config>

<get>

<copy>

<commit>

Working copy to manipulate with no impact on current configuration

Complete and active configuration and operations data

Configuration loaded at startup
Network Management Datastore Architecture (NMDA)

- At times there are differences between
  - the value configured by the user or an application (configuration) and…
  - the value that the device is actually using (operational state)

- These two values may be different for a number of reasons, e.g., system internal interactions with hardware, or simply the time it takes to propagate a configuration change
  - intended configuration: configuration that is intended to be used by the device
  - applied configuration: configuration that is actively in use by a device

- Introduces the read-only intended configuration datastore (<intended>)

- Described in draft-ietf-netmod-revised-datastores
NETCONF Transactions

Transaction Definition: the *ACID test*

- Atomicity: all-or-nothing, great for error handling
- Consistency: all-at-once, great for simplicity
- Independence: no-crosstalk, great for many concurrent clients
- Durability: done-is-done, great for reliability

Introduction of transactions + SQL caused a DB industry boom in the 80’s. Applications got reliable. Could run against many different DBMS’

*NETCONF makes the network a distributed database*
NETCONF YANG Module Capabilities (Hello message)

```xml
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <capabilities>
    <capability>urn:ietf:params:netconf:base:1.1</capability>
    <capability>urn:ietf:params:netconf:capability:startup:1.0</capability>
    <capability>urn:ietf:params:xml:ns:yang:ietf-interfaces</capability>
  </capabilities>
  <session-id>4</session-id>
</hello>
```
NETCONF YANG Module Capabilities (YANG library)

GET request on well-known path /modules-state

```xml
<modules>
  <module>
    <name>ietf-interfaces</name>
    <revision>2014-05-08</revision>
    <schema>http://localhost:8008/restconf/tailf/modules/ […]</schema>
    <feature>pre-provisioning</feature>
    <feature>if-mib</feature>
    <feature>arbitrary-names</feature>
    <conformance-type>implement</conformance-type>
  </module>
</modules>
```
<rpc message-id="102" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
        <identifier>ietf-interfaces</identifier>
    </get-schema>
</rpc>

[...]

<rpc-reply message-id="102" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <data xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-monitoring">
        module ietf-interfaces { //default format (yang) returned
            //ietf-interfaces yang module
            //contents here ...
        }
    </data>
</rpc-reply>
The RESTCONF Protocol
The RESTCONF Protocol

- HTTP API for NETCONF Datastores
- IETF network management protocol (RFC 8040)
- Not intended to replace NETCONF, but rather provide an additional simplified interface
- Defines HTTP-based Create, Retrieve, Update, Delete (CRUD) operations
- Configuration data and state data exposed as resources
- Operations defined with YANG RPC invoked with the POST method
- Simplified transaction model

**Why you should care:**

RESTCONF provides a lighter-weight interface to NETCONF data stores leveraging the well known combination of HTTP and JSON/XML
RESTCONF versus NETCONF: Summary

• RESTCONF: no notion of transaction
• RESTCONF: no notion of lock
• RESTCONF: no notion of candidate config and commit
• RESTCONF: so no notion of two phase commit
• RESTCONF: no <copy-config>
• RESTCONF: some more granularity for query => "config", "nonconfig", "all".
• RESTCONF: XML or JSON (while NETCONF is XML only)

NETCONF might be better for router and switches
RESTCONF might be better for controller north-bound interface
RESTCONF Base Method Mapping

HTTP methods implement the equivalent of NETCONF operations, enabling basic CRUD operations

<table>
<thead>
<tr>
<th>RESTCONF</th>
<th>NETCONF</th>
</tr>
</thead>
<tbody>
<tr>
<td>OPTIONS</td>
<td>none</td>
</tr>
<tr>
<td>HEAD</td>
<td>&lt;get-config&gt;, &lt;get&gt;</td>
</tr>
<tr>
<td>GET</td>
<td>&lt;get-config&gt;, &lt;get&gt;</td>
</tr>
<tr>
<td>POST</td>
<td>&lt;edit-config&gt; (nc:operation=&quot;create&quot;)</td>
</tr>
<tr>
<td>POST</td>
<td>invoke an RPC operation</td>
</tr>
<tr>
<td>PUT</td>
<td>&lt;copy-config&gt; (PUT on datastore)</td>
</tr>
<tr>
<td>PUT</td>
<td>&lt;edit-config&gt; (nc:operation=&quot;create/replace&quot;)</td>
</tr>
<tr>
<td>PATCH</td>
<td>&lt;edit-config&gt; (nc:operation depends on PATCH content)</td>
</tr>
<tr>
<td>DELETE</td>
<td>&lt;edit-config&gt; (nc:operation=&quot;delete&quot;)</td>
</tr>
</tbody>
</table>
RESTCONF Datastores

GET
POST
PUT
PATCH
DELETE

{+restconf}/data

Complete and active configuration and operational data
RESTCONF YANG Module Information

GET /restconf/data/ietf-yang-library:modules-state HTTP/1.1
Content-Type: application/yang-data+json

[...]
{
  "ietf-yang-library:modules-state" : {
    "module-set-id" : "5479120c17a619545ea6aff7aa19838b036ebbd7",
    "module" : [
      {
        "name" : "foo",
        "revision" : "2014-05-08",
      }
    ]
  }
}
Getting YANG Modules with RESTCONF

GET https://example.com/modules/ietf-interfaces/2014-05-08 HTTP/1.1
Host: example.com
Accept: application/yang

HTTP/1.1 200 OK
Date: Thu, 11 Feb 2016 11:10:31 GMT
Server: example-server
Content-Type: application/yang

module example-jukebox {
  // contents of YANG module deleted for this example...
}

Client

Server
The YANG Data Modeling Language
The YANG Language

• A Data Modeling Language for Networking
• Human readable and easy to learn
• Hierarchical configuration data models
• Reusable types and groupings (structured types)
• Extensibility through augmentation
• Formal constraints for configuration validation
• Data modularity through modules and sub-modules
• Well defined versioning rules

**Why you should care:**
YANG is a full, formal contract language with rich syntax and semantics to build applications on.
Anatomy of a YANG module

- **Header**
- **Imports & Includes**
- **Type definitions**
- **Configuration & Operational data declarations**
- **Action (RPC) & Notification declarations**
The YANG Header

```yang
module ietf-interfaces {
  namespace "urn:ietf:params:xml:ns:yang:ietf-interfaces";
  prefix if;
  import ietf-yang-types {
    prefix yang;
  }
  organization
    "IETF NETMOD (NETCONF Data Modeling Language) Working Group";
  contact
    "WG Web:  <http://tools.ietf.org/wg/netmod>/
    WG List:  <mailto:netmod@ietf.org>
    ...
    description
      "This module contains a collection of YANG definitions for
       managing network interfaces.
    ...
    revision 2014-05-08 {
      description
        "Initial revision."
      reference
        "RFC 7223: A YANG Data Model for Interface Management";
    }
}
```
Defining a Container

Container statement:
• Defines an interior data node in the schema tree
• One argument – identifier
• No value, but has a list of child nodes in the data tree

```plaintext
container interfaces {
    description
        "Interface configuration parameters."
    ...
}
```
Defining a List

List statement:
- Defines an interior data node in the schema tree.
- Single argument - identifier,
- Represents a collection of entries — each entry consists of one or more nodes

```
container interfaces {
    ...
    list interface {
        key "name";
        description
            "The list of configured interfaces on the device."
    }
}
container interfaces-state {
    config false;
    list interface {
        key "name";
        description
            "Data nodes for the operational state of interfaces."
    }
}
```

`config false` — Data under `interfaces-state` is read-only

Config (RW) and State (RO) clearly separated in this model
Defining Leaves

A leaf is defined by an identifier and has a type

```
list interface {
    key "name";
    description "...";

    leaf name {
        type string;
        description "The name of the interface"
    }
}
```

- Leaf `name` serves as list key
- The type is string
YANG Data Types

YANG has a set of built-in types, similar to those of many programming languages

- binary
- bits
- boolean
- decimal64
- empty
- enumeration
- identityref
- instance-identifier
- int8, int16, int32, int64
- leafref
- string
- uint8, uint16, uint32, uint64
- union

Use `pattern`, `range`, and `length` statements to restrict values

```yang
type string {
    length "0..4";
    pattern "[0-9a-fA-F]*";
}
```
Defining new types

New types can be defined using the `typedef` statement

```c
typedef percent {
  type uint8 {
    range "0 .. 100";
  }
  description "Percentage";
}
```

```c
leaf completed {
  type percent;
}
```

**RFC 6991: Common YANG Data Types**
- ietf-inet-types (ipv4- and ipv6-addresses, domain-name, etc)
- ietf-yang-types (counters, gauges, date-and-time, etc)
Conditional Leaves - Features

The `feature` statement is used to mark parts of the model as conditional. The `if-feature` statement makes the parent statement conditional.

This leaf is a part of our model only if the `if-mib` feature is supported in the server.
Agenda

- Introduction
- What are NETCONF, RESTCONF, and YANG
  - Demo
- Data Modelling, Standard, and Opensource
- Programming the Cisco Networks
- Cisco Direction
- Where to Start?
- Conclusion
Demo Setup

• Staring at YANG
  • Sublime text - a sophisticated text editor for code, markup and prose (sublimetext.com)
    • With sublime-yang-syntax and sublime-yang-snippets

• Interacting with servers (NETCONF and RESTCONF)
  • ncclient - Python library for NETCONF clients (ncclient.org)
  • Postman – A Complete API Development Environment (www.getpostman.com)
  • Cisco NSO

• The Servers
  • IOS XE 16.6.2
  • ConfD (6.3) – For NETCONF and RESTCONF

• The demo is available on GitHub.
Agenda

• Introduction
• What are NETCONF, RESTCONF, and YANG
• Demo
• **Data Modelling, Standard, and Opensource**
• Programming the Cisco Networks
• Cisco Direction
• Where to Start?
• Conclusion
Internet Engineering Task Force

- Open process for Internet Standards
  - Produce the RFCs

- The Standard Development Organization that specifies
  - NETCONF, RESTCONF, and YANG,
  - but also SNMP and MIBs

- Foresaw a « tsunami of YANG models »

- The IESG redistributed workload in order to allow for resources to be focused on YANG model coordination (Dec 2014)
  - “Primary oversight responsibility and coordination of this work across areas (AD document ownership) becomes the responsibility of Benoit Claise”
MIB Modules versus YANG Data Models

- Writable MIB Module IESG Statement (March 2014):
  “IETF working groups are therefore encouraged to use the NETCONF/YANG standards for configuration, especially in new charters”


- RFC 6643: Translation of Structure of Management Information Version 2 (SMIv2) MIB Modules to YANG Modules.

- YANG data models for configuration and monitoring of new features

- Will SNMP disappear?
  - No: SNMP and MIB models do a good job for monitoring
  - SNMP MIBs are configuration and state information, but represented in a way that is unsuitable for configuration
IETF Achievements

- RFC 6087: Guidelines for Authors and Reviewers of YANG Data Model
- RFC 6241: Base NETCONF Protocol 1.1 (update from RFC 4741)
- RFC 6244: An Architecture for Network Management Using NETCONF and YANG
- RFC 7950: YANG 1.1 specifications (update from RFC 6020) YANG 1.0
- RFC 7951: JSON Encoding of Data Modeled with YANG
- RFC 8040: RESTCONF protocol

In progress:
- Revised Datastore: a better way to design YANG modules
- Subscribing to YANG datastore push updates (telemetry)
IETF: YANG Models Growth

http://claise.be/IETFYANGPageCompilation.png
Cisco is committed to YANG model standardization and development.
Coordination is Required, now!
Coordination is Really Required, Now!

- Previous picture is about the IETF YANG models
  - New dimensions: different SDOs/Opensource projects
  - New dimension: versioning

- These YANG models must work together to create services
- Good problem to have: All YANG models arrive at the same time
  - As opposed to MIB modules in the past

- Standard Development Organizations (SDOs) can’t work in isolation: industry wide coordination is required

- Openconfig:
  - Pro: a few editors, for all YANG modules
  - Con: YANG modules change on regular basis
OPENCONFIG

• Operators-led YANG models
  • Google, AT&T, British Telecom, Microsoft, Facebook, Comcast, Verizon, Level3, Cox Communications, Yahoo!, Apple, Jive Communications, Deutsche Telekom / TeraStream, Bell Canada

• Focus: 123 network elements YANG models
  • Routing (BGP, ISIS, RIB, network-instance), routing policy, interfaces
  • Layer2 (vlan, spanning tree), ACL, optical transport, MPLS, etc.

• YANG models not aligned with the IETF
• Location: https://github.com/openconfig/public

www.openconfig.net
OPENCONFIG

- Streaming Telemetry specifications and configuration
- gRPC Network Management Interface (gNMI)
  - Protocol: gRPC
  - Encoding: protobuf
- Network management paradigm:
  - config without transaction,
  - then telemetry to check when applied
- Cisco is implementing openconfig data models
YANG Tsunami in the Industry

https://cisco.jiveon.com/groups/model-based-management/blog/2016/10/06/yang-modeling-efforts-in-the-industry
Controller

Orchestrator

Client

Service Delivery
YANG Modules

Network
YANG modules

Network Element
YANG modules

BRKNMS-2032 © 2018 Cisco and/or its affiliates. All rights reserved. Cisco Public 58
SDOs Alignement and Trajectory

Network Service YANG data models
- VPWS
- L2VPN
- VPLS
- L2VPN
- L3VPN

Network Element YANG data models
- MPLS
- BGP
- IPv4 & IPv6
- Ethernet
Data Model Location and Type (Network Element)

Network Element

- Standard YANG Model
- Proprietary Extension to Standard YANG Model
- Proprietary YANG Model (also called « native » models)
Numbers

- IETF YANG modules
  - Total from RFC: 50
  - Total in drafts: 237

- Openconfig
  - Total: 123

- Number of YANG data models in my VM
  - Total: 11510
  - Duplicates removed: 2591
  - Operational removed: 2423
  - Vendors removed: 1140

This becomes an industry problem!
How to Organize the Industry?

• With a YANG catalog, which contains all the modules

• The related metadata regarding maturity level, model type, implementation (which ones are important?), etc

• Based on the openconfig catalog as a starting point

• The inventory of all YANG modules, cross SDOs, cross vendors
  • SDOs on board: IETF, BBF, IEEE, ONF… some under discussion
  • Some vendors on board: Cisco, Huawei… some under discussion (Juniper)
  • Openconfig

• Started as IETF hackathons
A repository of YANG tools and the metadata around YANG models with the purpose of driving collaboration between authors and adoption with consumers.
Agenda

• Introduction
• What are NETCONF, RESTCONF, and YANG
• Demo
• Data Modelling, Standard, and Opensource
• **Programming the Cisco Networks**
• Cisco Direction
• Where to Start?
• Conclusion
Why Data Model-Driven Management?

- APIs derived from the data models:
  - Data models = definitions and constraints
  - The protocol: NETCONF or RESTCONF (or GRPC)
  - The encoding: JSON or XML
  - The programming language: Python, Ruby, Java, C, Erlang, ...
    Advice: learn python

- Industry focusing on YANG as the data modeling language for services and devices
Data Model Language (schema language)

Data Modeling (schema)

Encoding (serialization)

Protocol

Application

Prog. Language

YANG

YANG Data Model

XML

JSON

ProtoBuf

Thrift

NETCONF

RESTCONF

GRPC (HTTP/2)

YANG Development Kit

Python

C++

C++

Any language
NETCONF versus RESTCONF

- **RESTCONF:**
  - The HTTP POST, PUT, PATCH, and DELETE methods are used to edit data resources represented by YANG data models
  - RESTCONF is rest-like
  - Encoding: XML or JSON

- **NETCONF**
  - XML only
Data Model Driven Management: Example

Acting on resources

```
Module my-interfaces {
    namespace "com.my-interfaces";

    container interfaces {
        list interface {
            key name;
            leaf name { type string; }
            leaf admin-status { type enum; }
        }

        rpc flap-interface {
            input {
                leaf name { type string; }
            }
            output {
                leaf result { type boolean; }
            }
        }
    }
}
```

**GET**: Gets a resource

- GET /restconf/data/my-interfaces:interfaces
- GET /restconf/data/my-interfaces:interfaces/interface/<some name>

**POST**: Creates a resource or invoke operation

- POST /restconf/operations/my-interfaces:flap-interface + JSON/XML Form Data (including name)
  - Response will have JSON/XML result

**PUT**: Replaces a resource

- PUT /restconf/data/my-interfaces:interfaces/interface/<some name> + JSON/XML Form Data (name, admin-status)

**DELETE**: Removes a resource

- DELETE /restconf/data/my-interfaces:interfaces/interface/<some name>
Data Models Driven Set of APIs
Data Model-Driven Management

• Scripting: easy to create, hard to maintain/clean-up
  • => Data model-driven set of APIs

Data Models = APIs

• However,

Automation is as good as your data models and your toolchain

• Advice: you must play with/learn the toolchains
Cisco: Evolved Manageability & Data Models

- Deliver NETCONF and RESTCONF interfaces defined by YANG models
- Consistent implementation across IOS-XR, IOS-XE and NX-OS
- Common guidelines for model syntax and structure
- Shared repository and tool chains to support consistent model implementation
Programming the Cisco Networks

1. Understand the data model-driven management concept
2. Learn the NETCONF/RESTCONF/YANG basics
3. Where are the YANG data models?
4. Finding the right YANG data model
5. Experiment from a GUI and Code Generation
6. Testing with a virtual Image
7. What about upgrading the IOS?
8. What’s missing? PubSub / Telemetry
9. Putting it all together: Network Services Orchestrator (NSO)
3. Where are the Supported YANG Data Models?

YANG models for all platforms:

- **common** - across NX-OS, IOS-XE, and IOS-XR
- **nx** – NX-OS specific models
- **xe** - IOS-XE specific models
- **xr** - IOS-XR specific models

Each subdirectory has OS/platform-specific info in a README file

https://github.com/YangModels/yang/tree/master/vendor/cisco
3. Where are the Supported YANG Data Models?
Publication by OS

OS-specific YANG models;

- organized by OS-version
- sub-directories for contain models for each OS-version
- README files contain notes relative to models in the directory
- links to relevant platform specific documentation

https://github.com/YangModels/yang/tree/master/vendor/cisco/xr
3. Where are the Supported YANG Data Models? Publication by OS-release

OS-version specific YANG models;

- All models support in OS-version
- Compliance statements
- Deviations
- Known issues

https://github.com/YangModels/yang/tree/master/vendor/cisco/xr/611
3. Where are the Other YANG Data Models?

- IETF RFC:
  - https://github.com/YangModels/yang/tree/master/standard/ietf

- IETF drafts:

- Much statistic information on www.claise.be (daily cron job)

- And don’t forget the discovery capability

- Now all loaded in www.yangcatalog.org
  - See next slide
4. Finding the Right YANG Data Model

- [http://www.yangcatalog.org](http://www.yangcatalog.org)
  - Busy including all the YANG modules from the industry
  - Busy working on a Cisco specific instance
  - Contains a YANG DB search

- Allow one to find out what YANG modules and features are supported by a given platform, OS, license, etc.

- Demo: YANG search + YANG metadata + YANG tree
# YANG DB Search

Enter your search term(s) below:

| vrf |

## Search Options

- Case-Sensitive
- Regular Expression
- Include MIBs

## Schema Types

- All
- **Typedef**
- **Identity**
- **Container**
- **Leaf**
- **Grouping**
- **Extension**
- **List**
- **Notification**
- **Feature**
- **RPC**
- **Leaf-List**

[Search] [Reset]
<table>
<thead>
<tr>
<th>Name</th>
<th>Revision</th>
<th>Schema Type</th>
<th>Path</th>
<th>Module</th>
<th>Origin</th>
<th>Organization</th>
<th>Maturity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>address-family-vrf-grouping</td>
<td>2017-02-07</td>
<td>grouping</td>
<td>/ios-rip.address-family-vrf-grouping</td>
<td>Cisco-IOS-XE-rip</td>
<td>Vendor-Specific</td>
<td>cisco</td>
<td></td>
<td></td>
</tr>
<tr>
<td>af-ipv4-uc-and-vrf.cmds</td>
<td>2010-11-29</td>
<td>grouping</td>
<td>/bgp:af-ipv4-uc-and-vrf.cmds</td>
<td>brocade-bgp (Impact Analysis)</td>
<td>Vendor-Specific</td>
<td>brocade.com</td>
<td></td>
<td>Clear all non-default and default OSPF VRFs</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
4. Finding the Right YANG Data Model

Metadata are Important => Notion of Health Metric

- Organization: contact, maturity level
- Module: name, prefix, version, type, category, dependencies, document uri, submodules
- Implementation: status, platform, software release, opensource, contact
- And new one all the time. Ex: tree structure, generated from MIB, expired

Automation is as good as your data models, their metadata, and your toolchain
```json
{
  "input": {
    "organization": "openconfig",
    "implementations": {
      "implementation": {
        "vendor": "cisco",
        "software-version": "6.1.3",
        "platform": "ASR9K"
      }
    }
  }
}
```
Impact Analysis

• Demo


  https://www.yangcatalog.org/yang-search/impact_analysis.php?modules[]=ietf-interfaces&recurse=0&rfcs=1&show_subm=1&show_dir=both
Impact Analysis
Impact Analysis

YANG Impact Graph for Module(s): ietf-routing

Graph Options

Click on legend elements below to toggle highlighting on the graph.

Highlight All

Element Colors

- IETF
- CISCO

Rim Colors

- Maturity: ADOPTED
- Maturity: INITIAL
- Maturity: COMPILATION
- Maturity: RATIFIED

Modules: ietf-routing

Orgs: 

Recursion Levels: 0

Include Ratified Standards? 

Include Sub-modules? 

Show Graph Direction: Both

Generate

Export
Tracking Dependencies and Dependents
5. Experiment from a GUI and Code Generation

Demo:
- YANG Suite
- YDK
5. Generation of Model-Driven APIs Using YANG Development Kit (YDK)
5. YDK: Example

```python
# import providers, services and models
from ydk.services import CRUDService
from ydk.providers import NetconfServiceProvider
from ydk.models.bgp import bgp as oc_bgp

def config_bgp(bgp):
    bgp.global_.config.as_ = 65001  # assign AS number
    # configure IPv4 unicast address family
    afi_safi = bgp.global_.afi_safis.AfiSafi()
    afi_safi.afi_safi_name = "ipv4-unicast"
    afi_safi.config.afi_safi_name = "ipv4-unicast"
    afi_safi.config.enabled = True
    bgp.global_.afi_safis.afi_safi.append(afi_safi)

    bgp = oc_bgp.Bgp()  # create config model object
    config_bgp(bgp)  # add object configuration
```

https://developer.cisco.com/site/ydk/
6. Testing with a Virtual Cisco OS

• Testing with Vagrant ([http://gitlab.cisco.com/rschmied/dp-workbench](http://gitlab.cisco.com/rschmied/dp-workbench))
  • 1-2 CSR1000V virtual routers running IOS XE 16.3.2
  • The platform for the virtual machines is Vagrant using the VirtualBox provider.

• Documenting with Jupyter

• [https://learninglabs.cisco.com/labs](https://learninglabs.cisco.com/labs) with NETCONF and YANG tags

• Feedback: Integration into yangcatalog.org?
7. What About Upgrading the IOS?

Semantic Versioning: Tracking Module Changes

- YANG modules backward compatibility
  - What if I upgrade my router? Will my automation/automated service break?
- Note: the native models might not be backward compatible?
- Demo: check-semantic-version API

- Future:
  - working on a NSO package. Useful?
7. What About Upgrading the IOS?
Semantic Versioning: Tracking Module Changes

- Derived-semantic-version is determined using:
  1. Order all modules of the same name by revision from oldest to newest.
  2. If module A, revision N+1 has failed compilation, bump its derived semantic MAJOR version.
  3. Else, run "pyang --check-update-from" on module A, revision N and revision N+1 to see if backward-incompatible changes exist.
  4. If backward-incompatible changes exist, bump module A, revision N+1's derived MAJOR semantic version.
  5. If no backward-incompatible changes exist, compare the pyang trees of module A, revision N and revision N+1.
  6. If there are structural differences (e.g., new nodes), bump module A, revision N+1’s derived MINOR semantic version.
  7. If no structural differences exist, bump module A, revision N+1’s derived PATCH semantic version.
7. What About Upgrading the IOS?
Semantic Versioning

```
POST https://yangcatalog.org:8443/check-semantic-version

```

```
{
  "input": {
    "old": {
      "implementations": {
        "implementation": {
          "vendor": "cisco",
          "software-version": "6.1.1",
          "platform": "ASR9K"
        }
      }
    },
    "raw": {
      "implementations": {
        "implementation": {
          "vendor": "cisco",
          "software-version": "6.1.3",
          "platform": "ASR9K"
        }
      }
    }
  }
}
```

```
{
  "output": [
    {
      "derived-semantic-version-results": "Both modules failed compilation",
      "name": "Cisco-IOS-XR-bundlemgr-cfg",
      "new-compiled-semantic-version": "2.0.0",
      "old-compiled-semantic-version": "1.0.0",
      "organization": "cisco",
      "module-name": " IOS-13.1K"
    }
  ]
}
```
### 7. What About Upgrading the IOS?

#### Semantic Version Diffs

<table>
<thead>
<tr>
<th>Module: ietf-interfaces-old-tree.txt</th>
<th>Module: ietf-interfaces-tree.txt</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Diagram of module structure" /></td>
<td><img src="image" alt="Diagram of module structure" /></td>
</tr>
</tbody>
</table>

The Catalog can provide links to diff the module’s tree and full structure.
Eating our own Dog Food: yangcatalog.org is API driven

- Yangcatalog is Confd based, with the YANG module in draft-clacla-netmod-model-catalog-02
  - All APIs are automatically generated

- Demo: POSTMAN collection
  - Ex: https://yangcatalog.org:8443/search/name/Cisco-IOS-XR-ipv4-bgp-cfg
  - Ex: all openconfig YANG modules on OS/platform

- We want operators to integrate the APIs in their toolchain
  - https://www.getpostman.com/collections/9c941bd4c3243cc8137f

- Even if this is proof of concept, we have the API ready for a tool on cisco.com.
  - Feedback: GUI or RESTCONF call?
Eating our own Dog Food: YANG Module Structure

Module Sub-tree

Vendor Sub-tree
8. What’s Missing? PubSub / Telemetry
Solving a SNMP Polling, Data Modeling, and OPEX Issues

- Apps
  - App1
  - App2
  - App3

- APIs
  - Model-Driven APIs
    - YANG Development Kit (YDK)

- Network
  - Protocol
  - Encoding
  - Transport

- Models
  - XR Data Models
    - (native, open)

- Data
  - IOS-XR

Configuration
Streaming Telemetry
8. YANG for Notification: Pub Sub Concepts

- Any YANG subtree on device
- Statically configured or dynamically signaled
- Standards based mechanism

Streaming of updates
- Customized to recipient
- Periodic or on-change
- Encoding options (XML, JSON)
- Transport options (NETCONF, HTTP/2, GRPC)
- OpenDaylight code
9. Putting it all together: Network Services Orchestrator (NSO)

- Logically centralized network services
- Data models in YANG for data structures for:
  - Service instances
  - Network configuration and state
- Mapping service operations to network configuration changes
- Transactional integrity
- Multiprotocol and multivendor support

Applications

- REST, NETCONF, Java, Python, Erlang, CLI, Web UI

Engineers

- Service Manager
- Device Manager
- Network Equipment Drivers (NEDs)
- Service Models
- Device Models
- NETCONF, REST, SNMP, CLI, etc

Physical Networks

Virtual Networks

Network Apps

VNF Manager

- Overlay Mgmt
- Controller Apps
- EMS and NMS

Engineers

- Logically centralized network services
- Data models in YANG for data structures for:
  - Service instances
  - Network configuration and state
- Mapping service operations to network configuration changes
- Transactional integrity
- Multiprotocol and multivendor support
Agenda

• Introduction
• What are NETCONF, RESTCONF, and YANG
• Demo
• Data Modelling, Standard, and Opensource
• Programming the Cisco Networks
• Cisco Direction
• Where to Start?
• Conclusion
# Programmatic Interfaces: NETCONF Support

<table>
<thead>
<tr>
<th></th>
<th>Shipping</th>
<th>Committed for CY18</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>IOS-XR</strong></td>
<td>as of IOS-XR 5.3.0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• ASR 9000</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• NCS 1000</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• NCS 5000</td>
<td></td>
</tr>
<tr>
<td><strong>IOS-XE Nova</strong></td>
<td>as of IOS-XE 3.9.1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Catalyst 4500</td>
<td></td>
</tr>
<tr>
<td><strong>IOS-XE 16.x.1</strong></td>
<td>as of IOS-XE 16.3.1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Catalyst 3650</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Catalyst 3850</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• ASR 1000</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• ISR 4400</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• CSR 1000v</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• ISRv</td>
<td></td>
</tr>
<tr>
<td><strong>NX-OS</strong></td>
<td>as of “7.0(3)I5(1)” Release</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Nexus 3000</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Nexus 9000</td>
<td></td>
</tr>
</tbody>
</table>
# Programmatic Interfaces: RESTCONF Support

<table>
<thead>
<tr>
<th></th>
<th>Shipping</th>
<th>Committed for CY18</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>IOS-XR</strong></td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td><strong>IOS-XE Nova</strong></td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td><strong>IOS-XE 16.x.1</strong></td>
<td>IOS-XE 16.6.1</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>• Catalyst 3650, 3850, 9300, 9400, 9500</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• ASR 1000</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• ISR 4200, 4300, 4400</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• CSR 1000v</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• ISRv</td>
<td></td>
</tr>
<tr>
<td><strong>NX-OS</strong></td>
<td>as of “7.0(3)I5(1)” Release</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>• Nexus 3000</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Nexus 9000</td>
<td></td>
</tr>
</tbody>
</table>
# Programmatic Interfaces: gRPC Support

<table>
<thead>
<tr>
<th>Platform</th>
<th>Shipping</th>
<th>Committed for CY18</th>
</tr>
</thead>
<tbody>
<tr>
<td>IOS-XR</td>
<td>IOS XR 6.1.1- ASR 9000 (64 bit ) NCS 5500/5000 NCS 6000 NCS 1000 NCS 4000</td>
<td>-</td>
</tr>
<tr>
<td>IOS-XE Nova</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>IOS-XE 16.x.1</td>
<td>-</td>
<td>Yes</td>
</tr>
</tbody>
</table>
| NX-OS          | as of “7.0(3)I5(1)” Release  
• Nexus 3000  
• Nexus 9000 | -                  |
IOS-XR: YANG Models

• 823 YANG modules in 6.3.1

• Mix of native, openconfig and IETF YANG modules

• Currently we support Yang Model for pretty much everything on XR barring few exceptions, such as Multicast.
IOS-XE: Native Configuration Model Modularization

Pre IOS-XE 16.5.1

Single Monolithic Data Model

ned.yang

IOS-XE 16.5.1+

Individual Feature Specific Data Models

feature 1.yang
feature 2.yang
feature 3.yang

feature 'n'.yang
## NX-OS: YANG Models

### NX-OS 7.0(3)I7(1)

<table>
<thead>
<tr>
<th>Type of model</th>
<th>Features</th>
</tr>
</thead>
</table>
| **NX-API REST and Native YANG models** | • Interface  
• Port Channel  
• VPC  
• VLAN  
• BFD  
• STP  
• SVI  
• DHCP  
• ARP  
• CDP  
• RBAC  
• AAA  
• TACACS  
• Tunnels  
• Sub-interfaces  
• Breakout  
• UDLD  
• Bootvar  
• Hostname  
• ICMP  
• ACL  
• QoS  
• BGP  
• L3VM  
• Segment Routing  
• VXLAN EVPN  
• ARP/AM, ND  
• IPv4/ICMPv4  
• IPv6/ICMPv6  
• VRRPv2  
• VRRPv3  
| **Open Config Models** | • BGP,  
• Interfaces  
• Local-routing,  
• Vlan  
• Syslog  
• NTP  
• Route Policy  
• HSRP  
• SNMP  
• L2RIB  
• MPLS static  
• Static Route  
|
XR Telemetry Evolution

- **6.0.0 / 6.0.1**
  - Policy driven Telemetry
  - 30 Sec
  - TCP and UDP

- **6.1.1**
  - Model driven telemetry (dynamic)
  - 10 Sec
  - gRPC

- **6.3.1**
  - Event Driven telemetry
  - RIB, Interface, Syslog
Agenda

• Introduction
• What are NETCONF, RESTCONF, and YANG
• Demo
• Data Modelling, Standard, and Opensource
• Programming the Cisco Networks
• Cisco Direction
• Where to Start?
• Conclusion
General Education

• Link to IETF 94 Recording: NETCONF, YANG, pyang
  • Link to slides at IETF 94: NETCONF Slides, YANG Slides,

• RFC 6244, An Architecture for Network Management Using NETCONF and YANG
  • http://www.yang-central.org
  • http://www.netconfcentral.org/
  • http://www.yangcatalog.org for the tooling

• World of Solutions this week
Latest Blogs

- YANG Data Modules Statistics
- YANG Catalog Latest Developments (IETF 100 Hackathon)
- NETCONF versus RESTCONF: Capability Comparisons for Data Model-driven Management
- YANG Catalog Latest Development (IETF 99 Hackathon)
- YANG Opensource Tools for Data Modeling-driven Management
Every Network Engineer should Learn Scripting => Tooling

• CCIE is not enough for operation and management 😊
• Virtualization is fast 😊
• Devnet will help
  • http://devnet.cisco.com
• No excuses not to learn Python: it's simple and free to learn/use

http://www.claise.be/2014/05/why-i-learned-python/
Tooling – Exploring and using NETCONF/YANG

- Editor plug-ins
  - emacs (yang-mode.el)
  - vim (yang.vim)
  - sublime text (sublime-yang-syntax)

- pyang
  - an extensible YANG validator written in Python. (Video training: pyang)
  - can be used standalone as a validator of YANG modules, or to generate YIN, YANG, DSDL and XSD from YANG and YIN.
    - https://github.com/mbj4668/pyang
    - http://www.yangvalidator.com/

- libsmi
  - A library allowing the generation of YANG models from SMI/SMIv2 compliant MIBs

http://www.yang-central.org
Tooling – Exploring and using NETCONF/YANG

- ncclient
  - a Python library that facilitates client-side scripting and application development around the NETCONF protocol (only supports NETCONF 1.0)

- Postman
  - a Chrome plugin for RESTCONF, allowing for customized sets of REST snippets to be easily built, maintain and shared. Useful for NETCONF via RESTCONF, for example Open Daylight

- OpenDaylight
  - enables auto-generation of RESTconf APIs from YANG models, with NETCONF client support
  - APIdocs feature provides a way to explore both local and mounted YANG models
Where to Learn and Share – DevNet

DevNet is Cisco's developer program. DevNet provides you with the training, tools, APIs and support you need to learn, innovate and build ... run your Cisco network even more effectively

http://devnet.cisco.com
Tooling – YDK

Starting point: https://developer.cisco.com/site/ydk/

GitHub

- YDK Python API – YDK-Py (https://git.io/vaWsg)
- YDK-Py sample apps (https://git.io/vaw1U)
- YDK Generator – YDK-gen (https://git.io/vaw1M)
- YANG Explorer (https://git.io/vg7Jm)

DevNet

- YDK at DevNet (https://goo.gl/Wqwp3C)
- Cisco IOS XR 6.0 at DevNet (https://goo.gl/uaxrpN)
Tooling – YDK

YDK Sandboxes
- Ubuntu YDK-PY Vagrant box (https://git.io/vaw1U)
- YDK on dCloud.cisco.com

YDK Support
- Cisco support community (https://communities.cisco.com/community/community/developer/ydk)

XR Programmability
- Model-Driven Programmability Overview (https://cisco.box.com/v/xr-mdp)
- dCloud Programmability Lab 1 - NETCONF/YANG (https://goo.gl/gMYXiT)
- dCloud Programmability Lab 2 - RESTCONF / NETCONF/gRPC (https://goo.gl/GYV9H8)
DevNet – Sandboxes, Learning Labs

- **Coding 101 - REST Basics**
  - Learn the basics of how to use REST APIs. Use POSTMAN to test making REST API calls using the APIC-EM APIs.
  - Duration: 20 min

- **Coding 102: Calling REST APIs from Python**
  - Learn the basics of how to call and consume a REST API in Python.
  - Duration: 35 min

- **Coding 201 - Parsing XML**
  - Learn the basics of how to use parse XML results using Python.
  - Duration: 20 min

- **Coding 202 - Parsing JSON**
  - Learn the basics of how to use parse JSON results using Python.
  - Duration: 15 min

https://learninglabs.cisco.com/labs/tags/Coding
DevNet – Sandboxes, Learning Labs

- **NETCONF 101: Introduction to NETCONF.**
  Learning the basics of the NETCONF protocol and how it differs from other programmatic interfaces such as RESTCONF.

- **NETCONF 102: Using Python to generate NETCONF API calls.**
  Learn how to call and consume a NETCONF API in Python.

- **NETCONF 103: Porting Your Scripts to NETCONF**
  Learn the basics of how NETCONF simplifies and improves network scripting.

- **YANG 101: Introduction to YANG and Data Modeling.**
  Learning the basics of the YANG data modeling language.

Visit [https://learninglabs.cisco.com/labs](https://learninglabs.cisco.com/labs) with NETCONF and YANG tags.
Telemetry Tools on Github:

- https://github.com/cisco/bigmuddy-network-telemetry-stacks
- https://github.com/cisco/bigmuddy-network-telemetry-collector
- https://github.com/cisco/xr-telemetry-m2m-web

Demos and Lab

- https://dcloud-cms.cisco.com/?p=22317 (dCloud telemetry lab)
- https://youtu.be/F_S9-ctNFe0 (demo on Fretta)
External Resources

- https://developer.cisco.com/site/ios-xr/ (devnet landing page for telemetry)

Panda

- http://panda.cisco.com/

YANG

- https://github.com/YangModels/yang/tree/master/vendor/cisco (Cisco YANG models)
- https://github.com/CiscoDevNet/openconfig-getting-started (lots of sample code)
# Standard References: NETCONF

<table>
<thead>
<tr>
<th>V 1.0</th>
<th>V 1.1</th>
<th>Extension</th>
</tr>
</thead>
</table>
| RFC 3535  
Background and Requirements    | RFC 6241  
1.1 Base NETCONF Protocol         | RFC 5277  
Event Notifications               |
| RFC 4741  
1.0 Base NETCONF Protocol        | RFC 6242  
NETCONF over SSH                  | RFC 5717  
Partial Locking                    |
| RFC 4742  
NETCONF over SSH                |                                    | RFC 6243  
With defaults                      |
|                                    |                                    | RFC 6244  
NETCONF + YANG Architectural Overview |
|                                    |                                    | RFC 6536  
NETCONF Access Control Model       |
Extra YANG Resources

• YANG Doctors: http://www.ietf.org/iesg/directorate/yang-doctors.html

• YANG Model Coordination Group: http://www.ietf.org/iesg/directorate/yang-model-coordination-group.html

• The Routing Area YANG Coordination Forum: http://trac.tools.ietf.org/area/rtg/trac/wiki/RtgYangCoord

• All the stats compiled on http://www.claise.be/2015/10/ietf-yang-modules-statistiques/

• Blogs at http://www.claise.be

• http://www.yangvalidator.org/

• http://trac.tools.ietf.org/area/ops/trac/wiki/YANGModelingEffortsInTheIndustry
Standard References: YANG Data Models

- RFC 6020: YANG – A Data Modeling Language for the Network Configuration Protocol
- RFC 6087: Guidelines for Authors and Reviewers of YANG Data Model Documents
- RFC 6110: Mapping YANG to Document Schema Definition Languages and Validating NETCONF Content
- RFC 6643: Translation of SMIv2 MIB Modules to YANG Modules
- RFC 6991: Common YANG Data Types
- RFC 7223: A YANG Data Model for Interface Management
- RFC 7224: IANA Interface Type YANG Module
- RFC 7227: A YANG Data Model for IP Management
- RFC 7317: A YANG Data Model for System Management
- RFC 7407: A YANG Data Model for SNMP Configuration
Network Services Orchestrator (NSO)

Test your code in NSO Sandbox

The NSO Sandbox is the Cisco cloud service provided to help customers, partners, and developers quickly integrate their solutions with a number of Cisco technologies. The NSO Sandbox provides a developer with an environment to design, develop and test using the NSO RESTful APIs.

FEATURED LAB

Cisco Network Services Orchestrator 4.1.1 Sandbox

This sandbox allows you to install and explore NSO, but the sandbox installation also includes examples.

Try It Out

developer.cisco.com/site/nso/
CiscoLive Follow Up

• Interested in a follow up this week?
  • We can plan “Meet the Engineer” for a more technical discussion
  • « Whisper Suite » for product manager and roadmap questions

• Spark room

• Contact:
  • Benoit Claise bclaise@cisco.com
  • Carl Moberg: camoberg@cisco.com
  • Robert Grasby (Manager Business Development): rgrasby@cisco.com
Agenda

• Introduction
• What are NETCONF, RESTCONF, and YANG
• Demo
• Data Modelling, Standard, and Opensource
• Programming the Cisco Networks
• Cisco Direction
• Where to Start?
• Conclusion
Summary and Key Messages

• Automation and programmability are required these days
• Data Modeling-driven set of APIs is key for automation
• YANG is the data modeling language for configuration and monitoring
• Many YANG data model developments
  • Standard development organizations (but primarily at the IETF),
  • Consortium
  • Native Models
• Cisco is committed: NETCONF (XML), RESTCONF (XML/JSON), gRPC
• Play with your toolchain
• For follow-ups, you know how to contact us.
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-2032
• 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/.
Continue Your Education

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

• Introduction
• What are NETCONF, RESTCONF, and YANG
• Demo
• Data Modelling, Standard, and Opensource
• Programming the Cisco Networks
• Cisco Direction
• Where to Start?
• Conclusion
Thank you
You’re it
Backup