Representing operational state in YANG

draft-openconfig-netmod-opstate

OpenConfig network operators group
Rob Shakir, BT.
Marcus Hines, Google.
Anees Shaikh, Google.

netmod interim.
June 2015.

Statements that are intended to be axiomatic (requirements).

Op-state must be in YANG data models.

network management != solely provisioning.

The way we represent op-state must be consistent across all models.

model-specific code continues to propagate complexity into NMS/OSS.

models are not used (and not useful) in isolation

Data models should be transport protocol independent.

YANG != NETCONF-specific.

Requirements (cont'd).

Be able to recognise intended configuration vs. applied configuration.

Not all devices are synchronous - or all data on a single device.

Not all management systems are transactional or synchronous.

Configuration should be considered part of state.

Be able to retrieve configuration and opstate separately.

Device: separate config and stats databases.

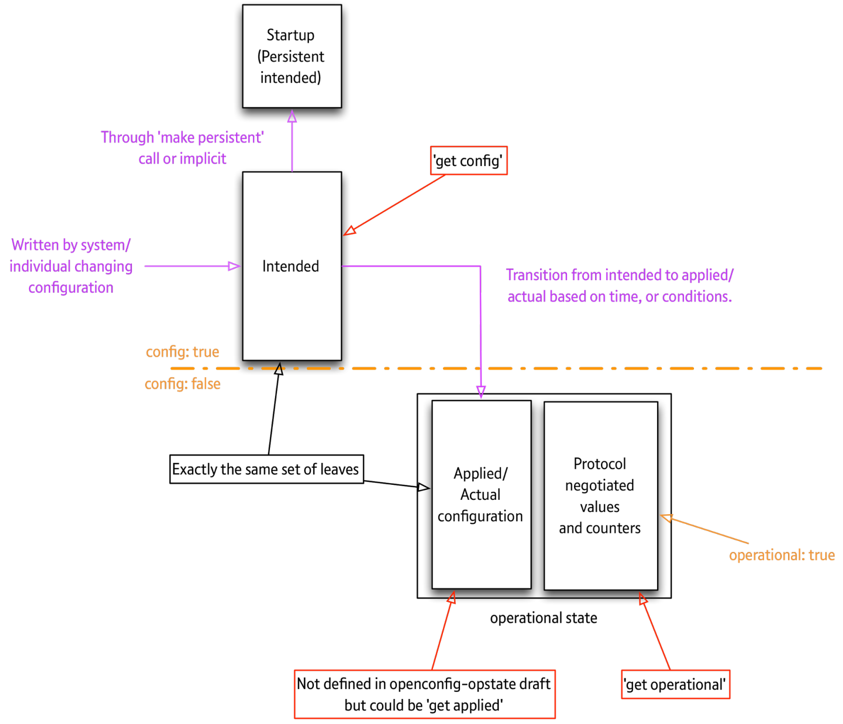
NMS: retrieve specific information that is owned by device (stats).

Ability to relate configuration to operational state must be consistent.

Common and deterministic get_state() & consistency_check() mechanisms which are model independent.

Illustration

based on NETMOD mailing list discussion after 6/18 interim



Startup (Persistent intended)
Through 'make persistent' call or implicit
'get config'
Written by system/ individual changing configuration
Intended
Transition from intended to applied/ actual based on time, or conditions.
config: true
config: false
Exactly the same set of leaves
Applied/ Actual configuration
Protocol negotiated values and counters
operational: true
operational state
Not defined in openconfig-opstate draft but could be 'get applied'
'get operational'

Rejected solutions.

{config,state}:/country[code='gb']/city[name='london']/device…

Requires datastore support in all APIs (IETF or otherwise)

Unclear what data is being accessed -- hidden in the API call

/device/routing-instance[name='base']/mpls/{config,state}/…

Non-deterministic split - why mpls/{..} rather than mpls/rsvp-te/{..}

/bgp/neighbors/neighbor[address='192.0.2.1']/timers/intended-hold-time
/bgp/neighbors/neighbor[address-'192.0.2.1']/timers/applied-hold-time
/bgp/neighbors/neighbor[address='192.0.2.1']/timers/negotiated-hold-time

Mixing of config true & config false leaves, difficult to filter

OpenConfig Solution.

/device[name='rt0']/interfaces/interface[name='eth0']/config/enabled

Intended.

/device[name='rt0']/interfaces/interface[name='eth0']/state/enabled

Applied.

/device[name='rt0']/interfaces/interface[name='eth0']/state/oper-status

Operational state.

get/get-config can filter on path.

Need a new get-operational RPC - filters on new operational true metadata.

consistency-check: foreach leaf in 'config': if ../state/leaf != leaf -> inconsistent.

Open netmod comments/queries.

Decreases readability and/or ease of model writing.

Only at the expense of required functionality - fair trade off.

# of model readers/writers << the number of NMS writers.

Subjective -- for some, common convention makes models easier to read

What do we do with existing RFC'd models?

Not clear that these are widely implemented/used - more important to have a consistent set of models which do become widely implemented.

'Why not use metadata?'

Not clear on the advantages, or how this would be implemented.

'Duplicates data on the wire'

<get-operational>, <get-config>, <get> or regexp path filters solve this issue.

Open netmod comments/issues.

‘Does not allow items that are [not configured | configured, not present | system-configured]?’ (e.g., for interfaces)

Config / state containers can be empty in case of unconfigured or not present

Oper status reflects status of an interface

Config items do not need to be only human/operator configured

‘Not clear what to do when intended and actual config are different’

An operational decision, should not be prescribed by the model

‘An "operational-path" statement solves this’

More complex solution -- potentially required on every data node

It is hard to control / check how people write YANG models

Checking for compliance to the structure is very simple

Summary of proposed changes

YANG modeling (RFC 6087)

- design pattern that:
  - provides consistent locations of modeled config and operational state, independent of model composition
  - includes configuration as part of state

NETCONF (RFC 6241) and other protocols

- RPC to support retrieval of only operational state (get-oper) based on YANG extension (operational: true)

YANG language (RFC 6020)

- more conventional map / list support (similar issues raised in ODL)
- relaxed constraints on config:true under config:false would further simplify the approach

OpenConfig - testing approach against real models.

BGP model - proposed to IDR, WG adoption call.

Uses config/state.

MPLS model - consolidated.

Reaching agreement with MPLS WG DT, converted to config/state.

Re-worked interfaces and local-routing.

To be published - uses config/state.

Widely discussed in OpenConfig (network operator community) with major implementors, and with routing model architecture design team.

No unresolved issues - or major objections raised.