

draft-openconfig-netmod-model- structure.

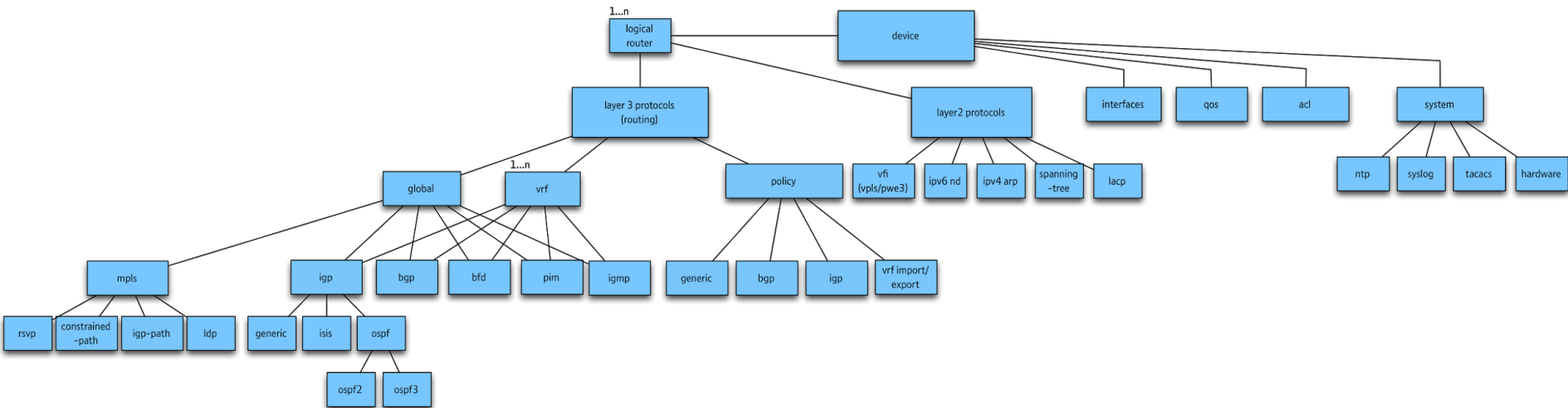
OpenConfig – www.openconfig.net

Anees Shaikh (Google), Rob Shakir (BT), Kevin D'Souza (AT&T), Luyuan Fang (Microsoft)

Motivation (I).

- Numerous IETF YANG models being defined.
 - Many assume that they sit at /<module-name>...
 - Structure for common config elements is not always specified – interface under protocol, or an augmented interface container?
 - No real defined granularity (LSP ping model vs. routing-cfg model...)
 - How does an operator actually know where one might find specific configuration or state in a YANG model tree?
- Without structure – difficult to define higher-layer services.
 - Which YANG modules are required to configure that service?
 - How should we map elements from a service model to elements in a device model?

Motivation (II).

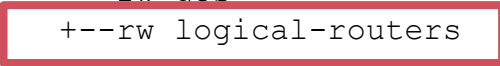


Aim of the draft.

- Put forward a strawman meta-model for how config & state related to 'device' can be structured.
 - We **do not** intend to be exhaustive – extensible.
 - Aim to be **vendor-neutral** – what makes sense from an operator perspective.
- YANG module is illustrative (not intended to be supported by a device)...
 - Start to define structure – thinking about how operator tools might interact with device.
 - Expect multiple instantiating protocols – so have not considered protocol-specific capability exchange yet.

High-level structure

```
+--rw device
  +--rw info
  +--rw hardware
  +--rw system
  | +--rw dns
  | +--rw ntp
  | +--rw dhcp
  | +--rw syslog
  | +--rw ssh
  | +--rw stat-coll
  | +--rw oam
  | +--rw aaa
  | +--rw users
+--rw interfaces
+--rw acl
+--rw qos
+--rw logical-routers
```



```
+--rw logical-router* [router-id]
  +--rw router-id          uint8
  +--rw router-name?      string
  +--rw layer-2-protocols
  | +--rw vsi
  | +--rw ipv6-ndp
  | +--rw arp
  | +--rw rstp
  | +--rw lldp
  | +--rw ptp
  +--rw layer-3-protocols
    +--rw global
    | +--rw bgp
    | +--rw igp
    | +--rw bfd
    | +--rw pim
    | +--rw igmp
    | +--rw static-routes
    | +--rw mpls
    +--rw vrf* [vrf-name]
    | +--rw vrf-name      string
    | +--rw bgp
    | +--rw igp
    | +--rw bfd
    | +--rw pim
    | +--rw igmp
    | +--rw static-routes
    +--rw routing-policy
```

How to structure models?

- Today: many root level containers – only some of which reference each other.
- Two potential approaches:
 - ‘Pull’ – each model defines only ‘grouping’, root module includes groupings.
 - ‘Push’ – each model defines augmentations to base model.
 - A hybrid would be ideal – but raises requirements for YANG compilers.

Model catalogue.

- Services will need to assemble models defined by the IETF and other organizations:
 - BBF, ONF, IEEE etc. already defining models.
 - Elements which are outside of the IETF's current scope – e.g., PHY configuration for G.FAST in BBF...
- Operationally useful to understand – dependencies for the model (inter-SDO); who is responsible; namespace...
 - useful when composing higher-layer service model.
 - a model structure would help us define this

Discussion.

- We believe that a well-defined structure for data models is critical.
 - Routing area already examining this question.
 - For example, VRF-centric vs. protocol centric.
- What are the other considerations that a structure needs to take into account?
 - Are these protocol-specific?
 - e.g., Do we need to take into account how a certain protocol advertises capabilities to support particular parts of models?