Operational Data in NETCONF & YANG

draft-bjorklund-netmod-operational-00

Martin Björklund
⟨*mbj@tail-f.com*⟩

Ladislav Lhotka
⟨*lhotka@nic.cz*⟩

11 March 2013

Problems to Address

❶ Parameters with “dual” character, where the value used operationally may differ from configured value. The client needs to be able to learn the operational value.

Example: IP address leafs in `ietf-ip` contain *configured* addresses. The address that’s in operational use should also be available.

Currently, this can only be accomplished by some form of data duplication (separate config and operational value, or config and RPC).

❷ I2RS talks about “fast path” configuration, i.e. ability to change state without the overhead of intervening data stores.

Such mechanisms should play well together with NETCONF.

❸ Information about system inventory, i.e. installed hardware, software options etc.

For example, this is where a client could find out about names and locations of installed interfaces.

❹ Default list contents: list entries that cannot be deleted but need to coexist with other (configured) entries in the same list.

Example: the main routing table is created by the system, and it cannot be deleted, but additional tables may be configured by the client.

Operational State Datastore

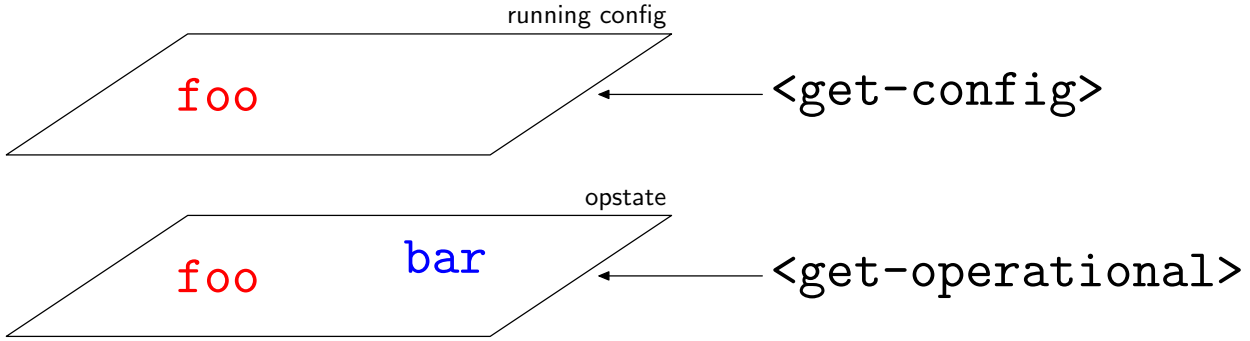
OSD contains all parameters that govern the device or provide information about its instanteneous state, i.e. both `config true` and `config false`.

OSD may be modified indirectly through NETCONF, but possibly also through other management interfaces (CLI, SNMP, I2RS agent…) and/or network protocols.

Config and Operational State

```
leaf foo {
    config true;
    …
}
```

```
leaf bar {
    config false;
    …
}
```



Data Validity and Constraints

Most constraints in YANG are currently imposed on configuration data, but in fact it is OSD where the validity really matters. Common rules are needed for coexistence with other management interfaces.

Options:

1. All constraints on config, nothing on OSD (no change).
2. All constrains on OSD, nothing on config.
3. New YANG statements for specifying constraints on OSD, e.g. `osd:mandatory`.

What It Does Not Solve

Cases where the config and operational values differ in their type.

Typical situations:

- The permitted values for the configurations are a superset of operational values.

  Example: Duplex can be `full|half|auto` in config but operational values are only `full` or `half`.

  This can be handled by the proposed approach - just one leaf definition for both.

- The opposite situation – possible operational values are a superset of config values.

  Example: Interface status can be configured as `up` or `down` but other operational values are possible: `lower-layer-down`, `testing` etc.

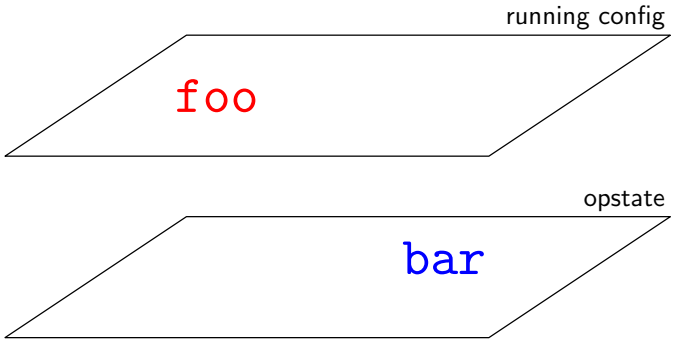
  These still have to be defined as separate leafs.

Various Ideas (Not in the I-D)

❶ Define global XML attribute “inactive” that blocks copying configuration to opstate.

In running:

```
<foo yang:inactive="true">6378</foo>
```



❷ Predefined read-only list entries in OSD?

Example: Main routing table can be a permanent entry of the list `rt:routing-table` and appear only in OSD. Additional configured entries would be appended to this list.