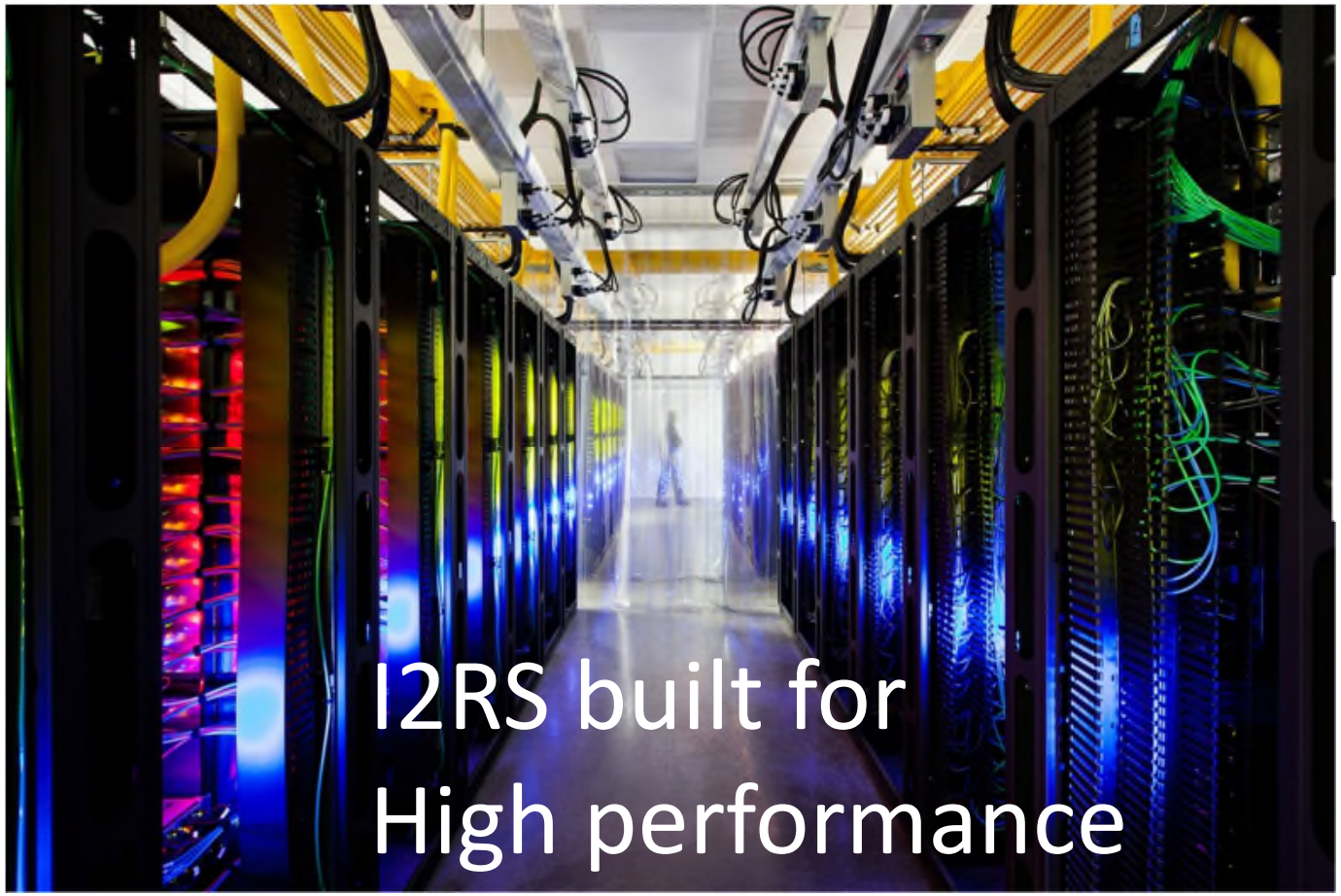
I2RS Protocol



Not the Pizza box CLI

Sue Hares

I2RS Protocol

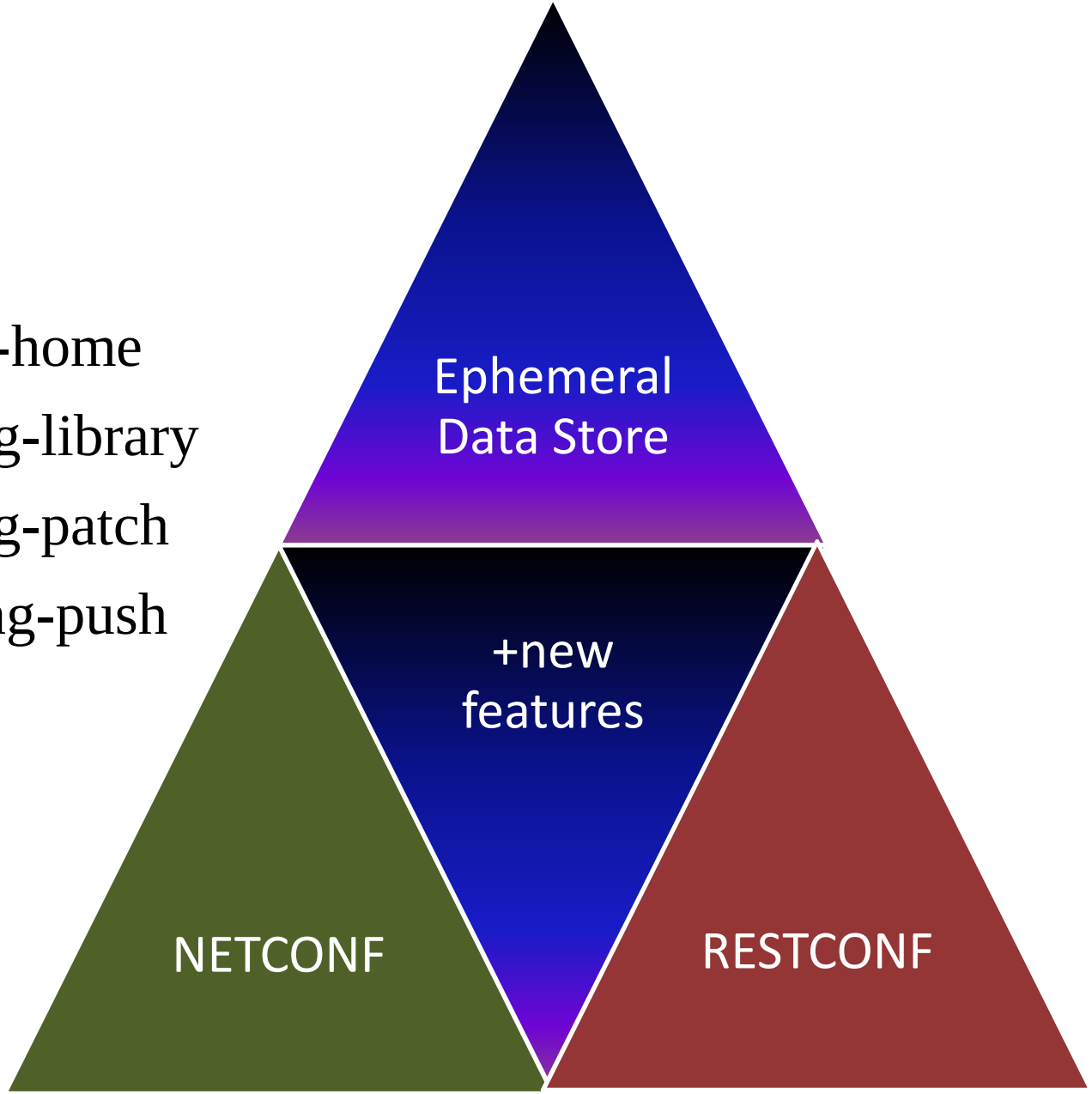
- Re-use Protocol
  - 5 Drafts

draft-ietf-netconf-call-home

draft-ietf-netconf-yang-library

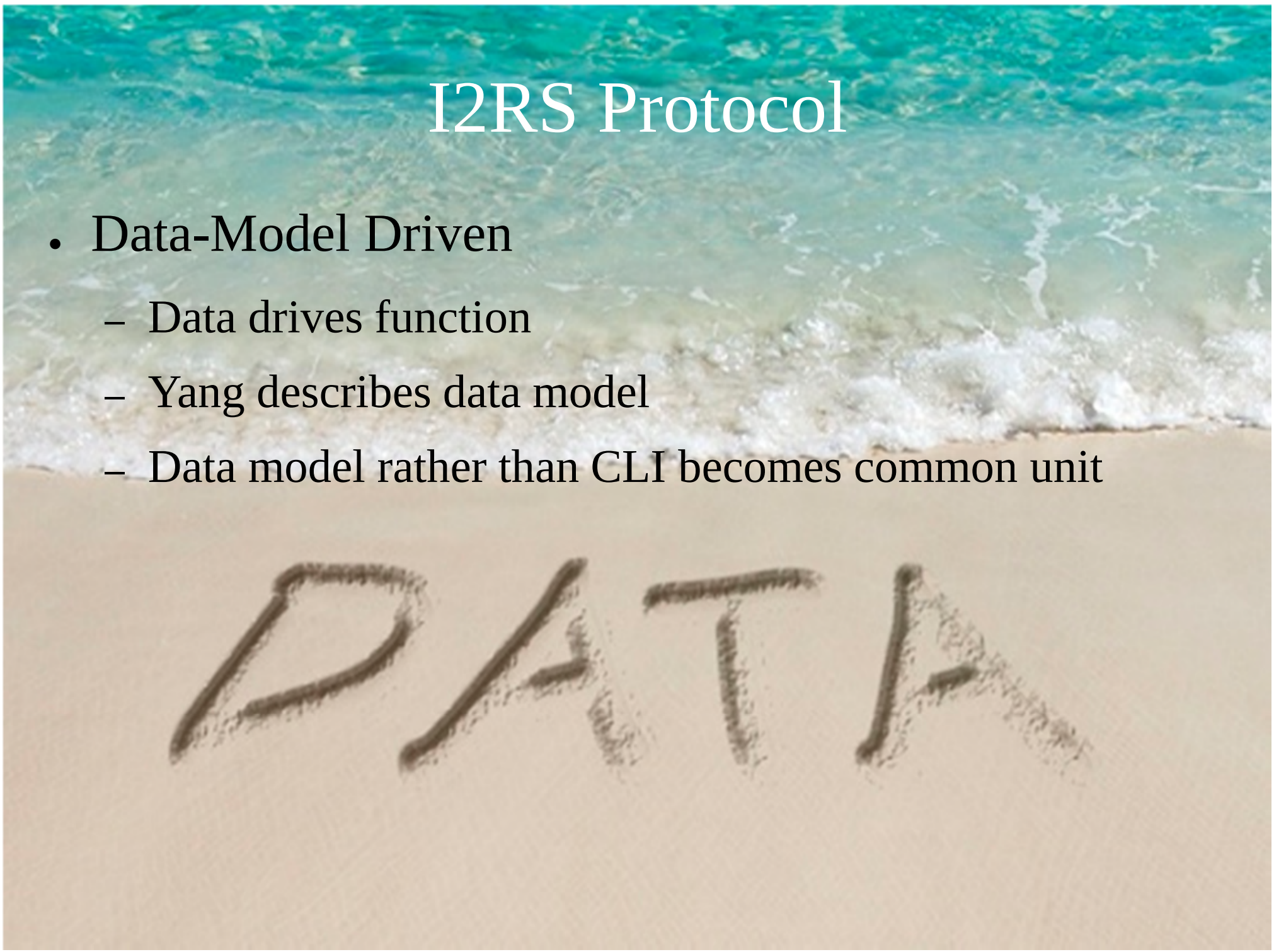
draft-ietf-netconf-yang-patch

draft-ietf-netconf-yang-push



I2RS Protocol

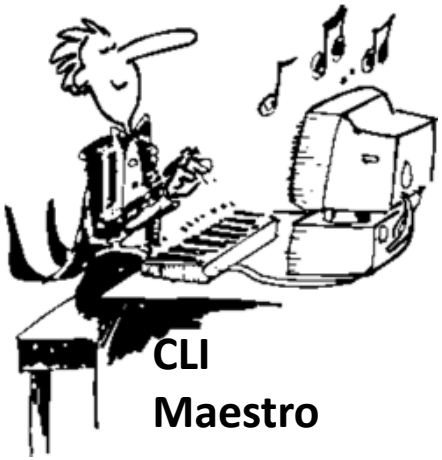
- Data-Model Driven
  - Data drives function
  - Yang describes data model
  - Data model rather than CLI becomes common unit

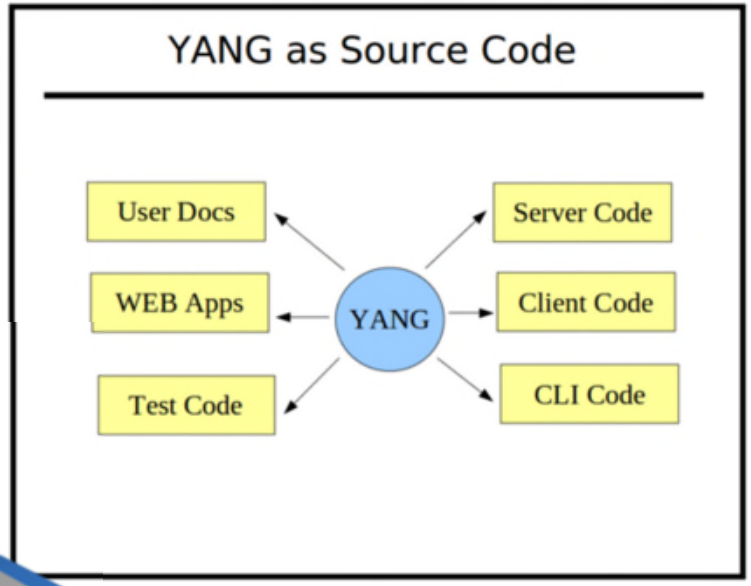


What is Ephemeral Configuration?

The Journey to Ephemeral

- CLI Maestro
- NETCONF-Yang (RFC6244)
- Opstate
- Ephemeral





CLI Maestro

config true;

config false;

operational data

Traditionally CLI configurations have been considered:

- **Candidate –** Added config
- **Running –** What system is running
- **Start-up –** What system reboots to

Operational data – what system creates

RFC6244 Data Model view

client

Netconf server

Meta Data

config

State data

notification

operation

Config DataBase

Intended Config

Ephemeral config

Routing Software

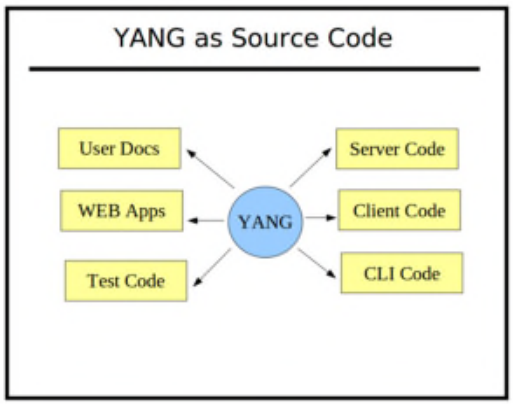
Applied Config

Derived State

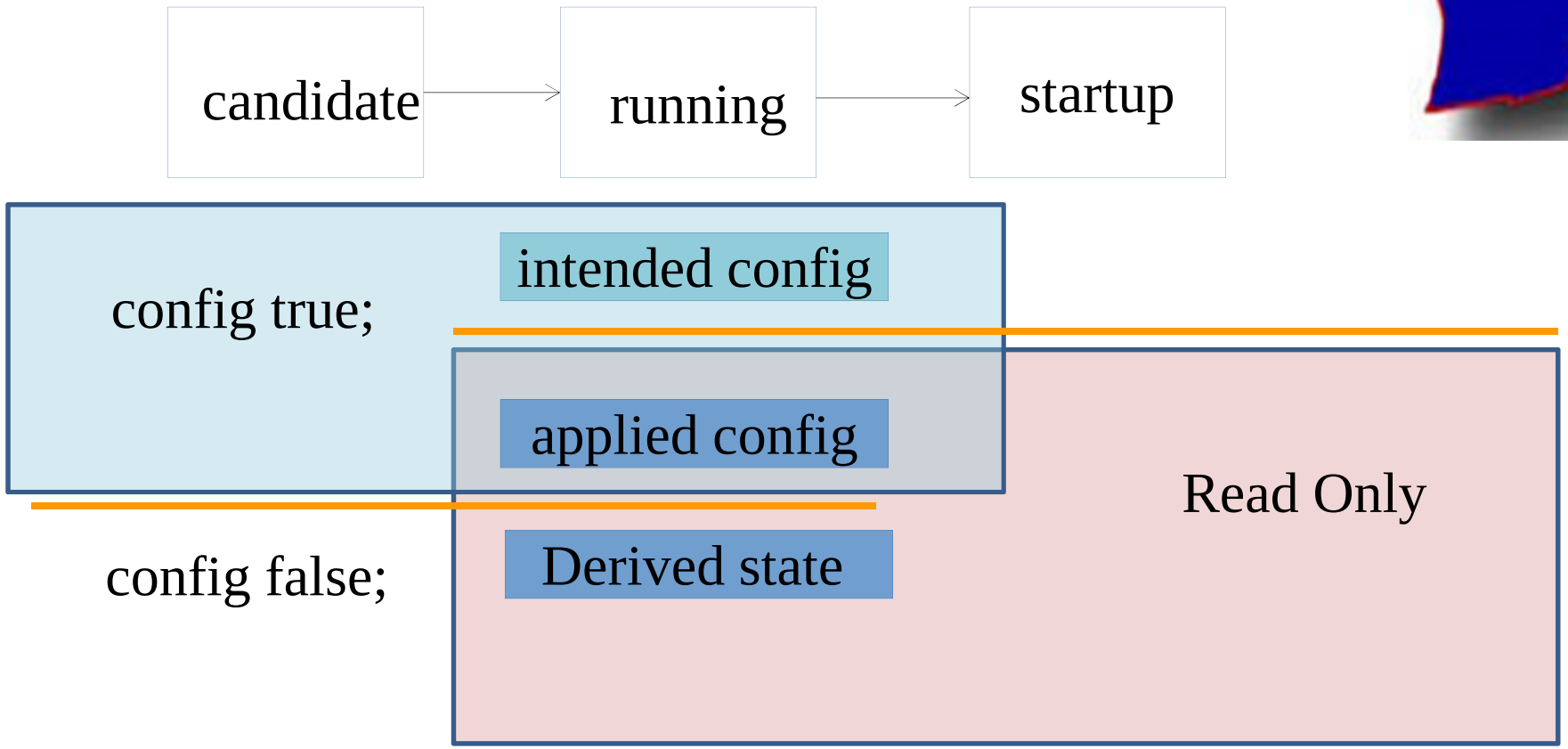
System Software

Applied Config

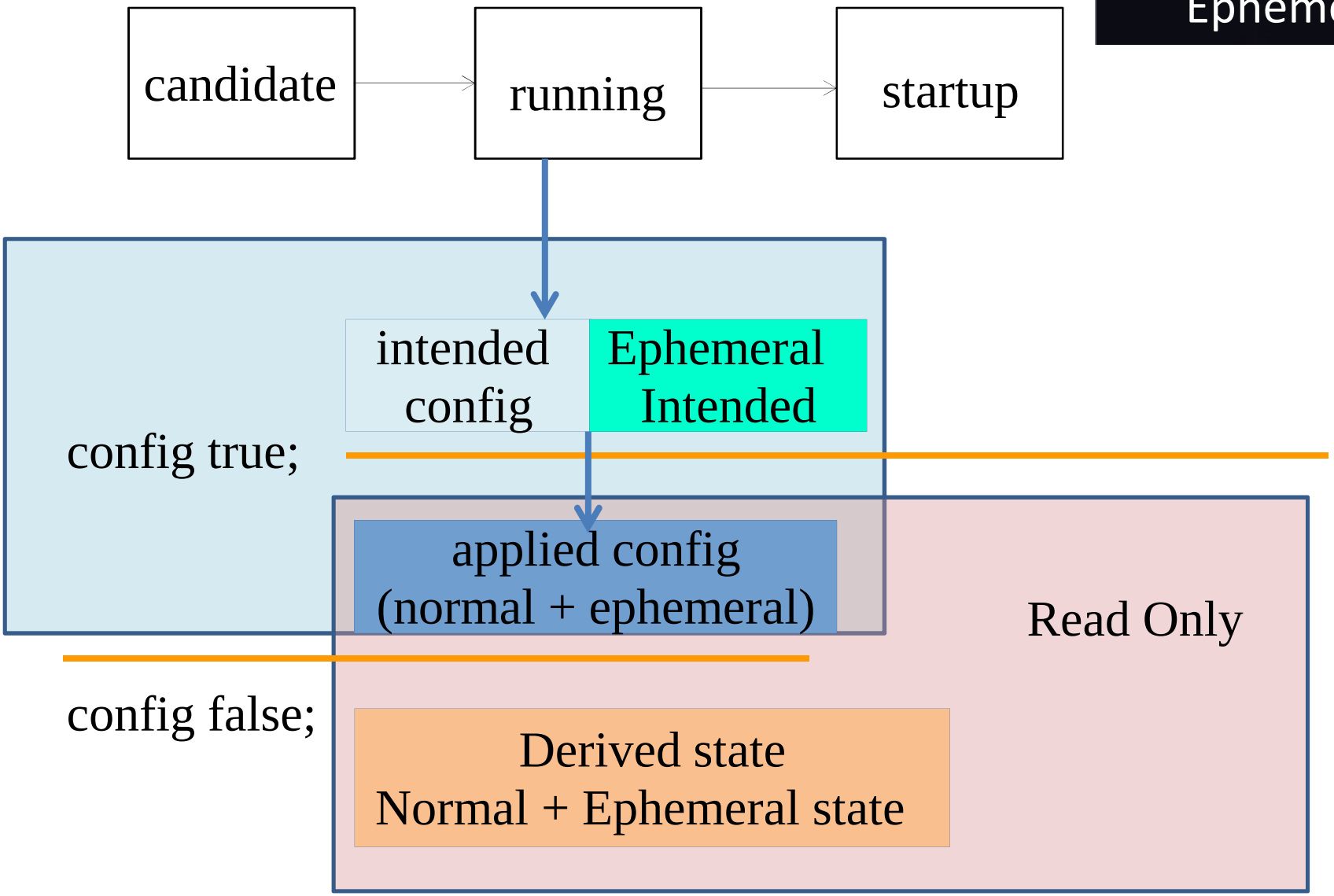
Derived State



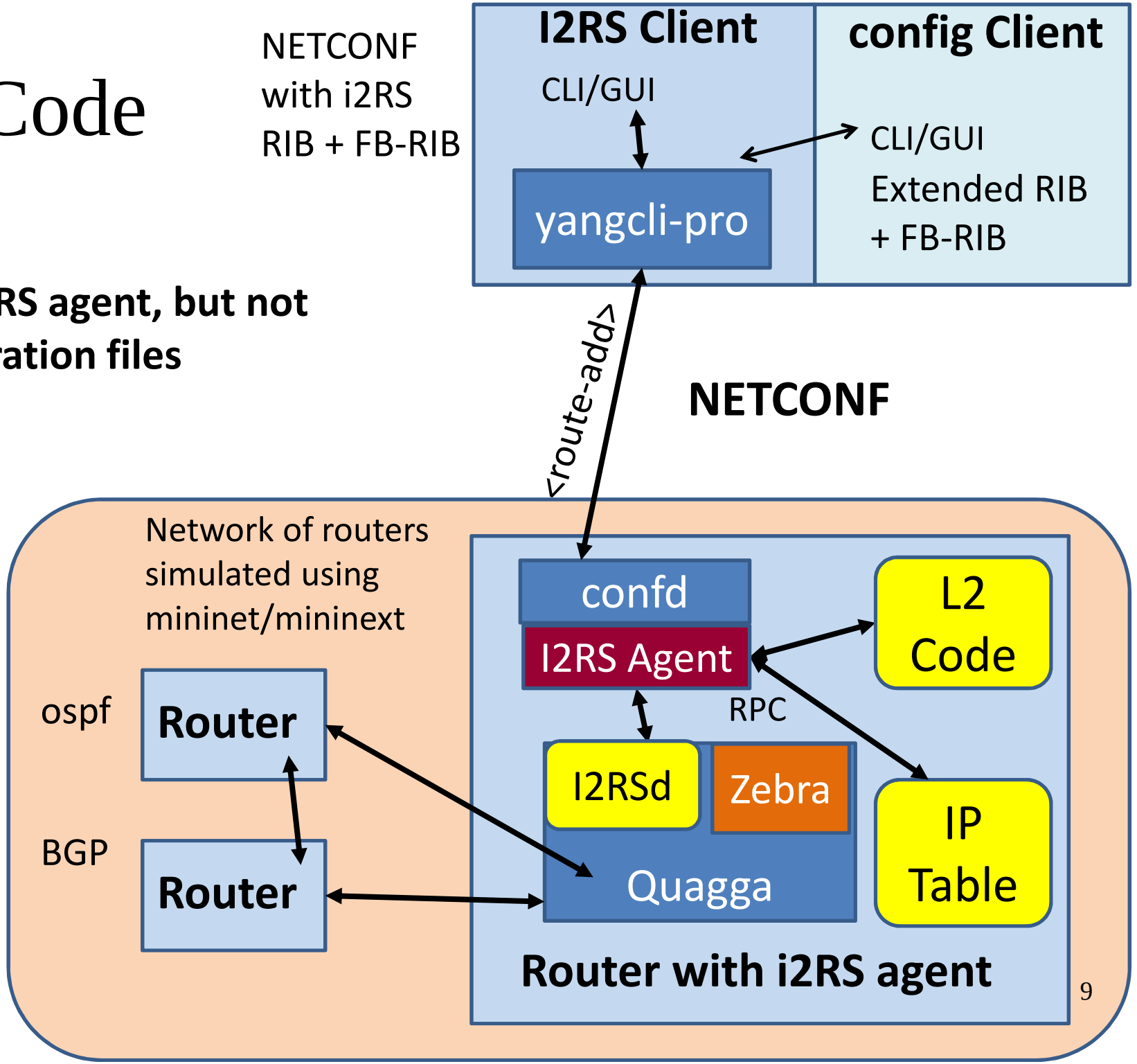
Definitions from ietf-netmod-opstate-req



Ephemeral Additions



I2RS Code


NETCONF
with i2RS
RIB + FB-RIB
I2RS Client
CLI/GUI
yangcli-pro
config Client
CLI/GUI
Extended RIB
+ FB-RIB
Store in I2RS agent, but not
in configuration files
<route-add>
NETCONF
Network of routers
simulated using
mininet/mininext
ospf
Router
BGP
Router
confd
I2RS Agent
RPC
L2
Code
I2RSd
Zebra
Quagga
IP
Table
Router with i2RS agent

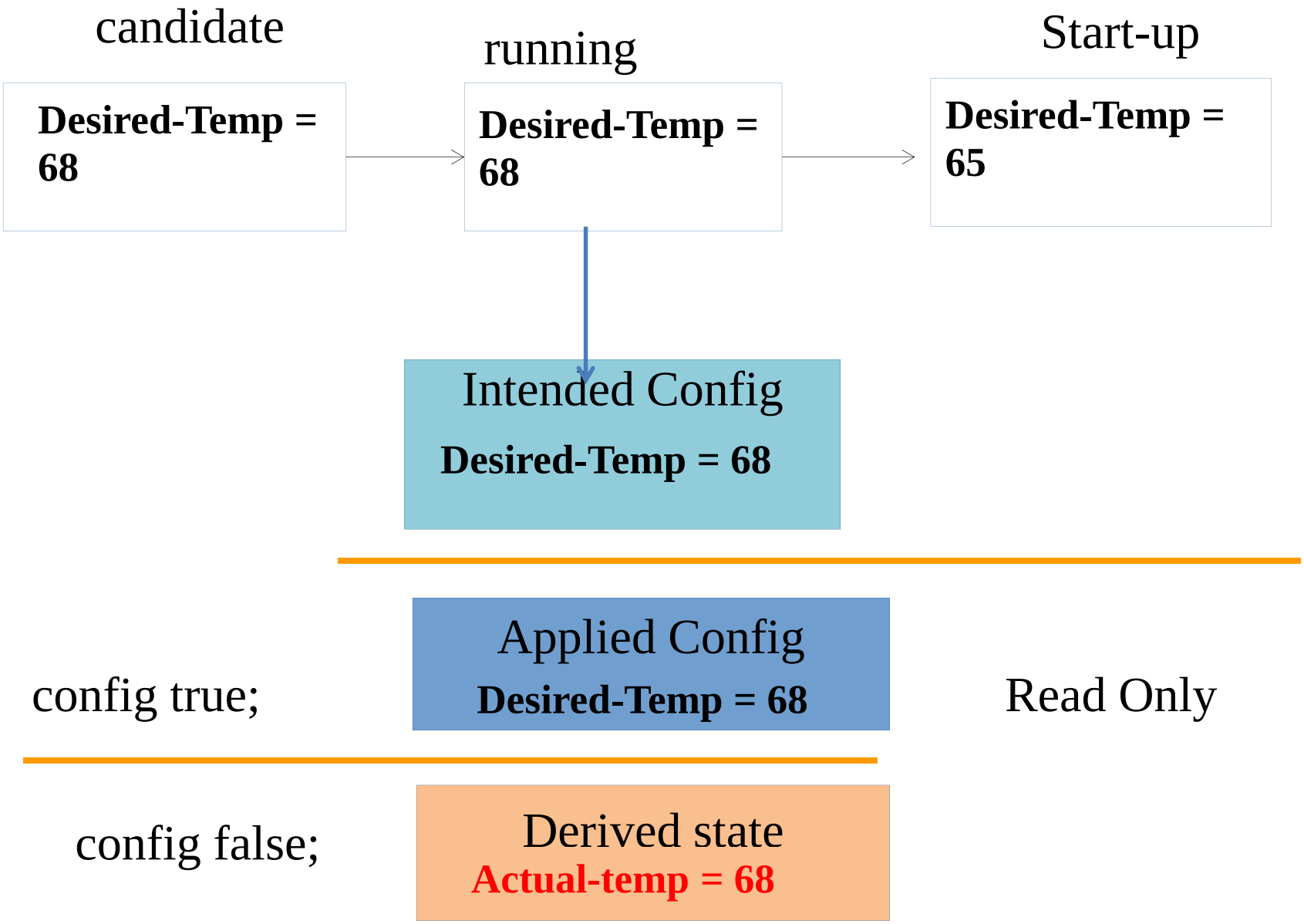
Examples

- Thermometer
- Route add
- Filter Add

Simple Thermostat Example

```
module thermostat {
 // Configuration
    leaf desired-temp {
        type int32;
        units “degrees Celsius”;
        description “The desired temperature”;
    }
 // operational state
    leaf actual-temp {
        type int32;
        config false;
        units “degrees Celsius”;
        description “The measured temperature”;
    }
}
```

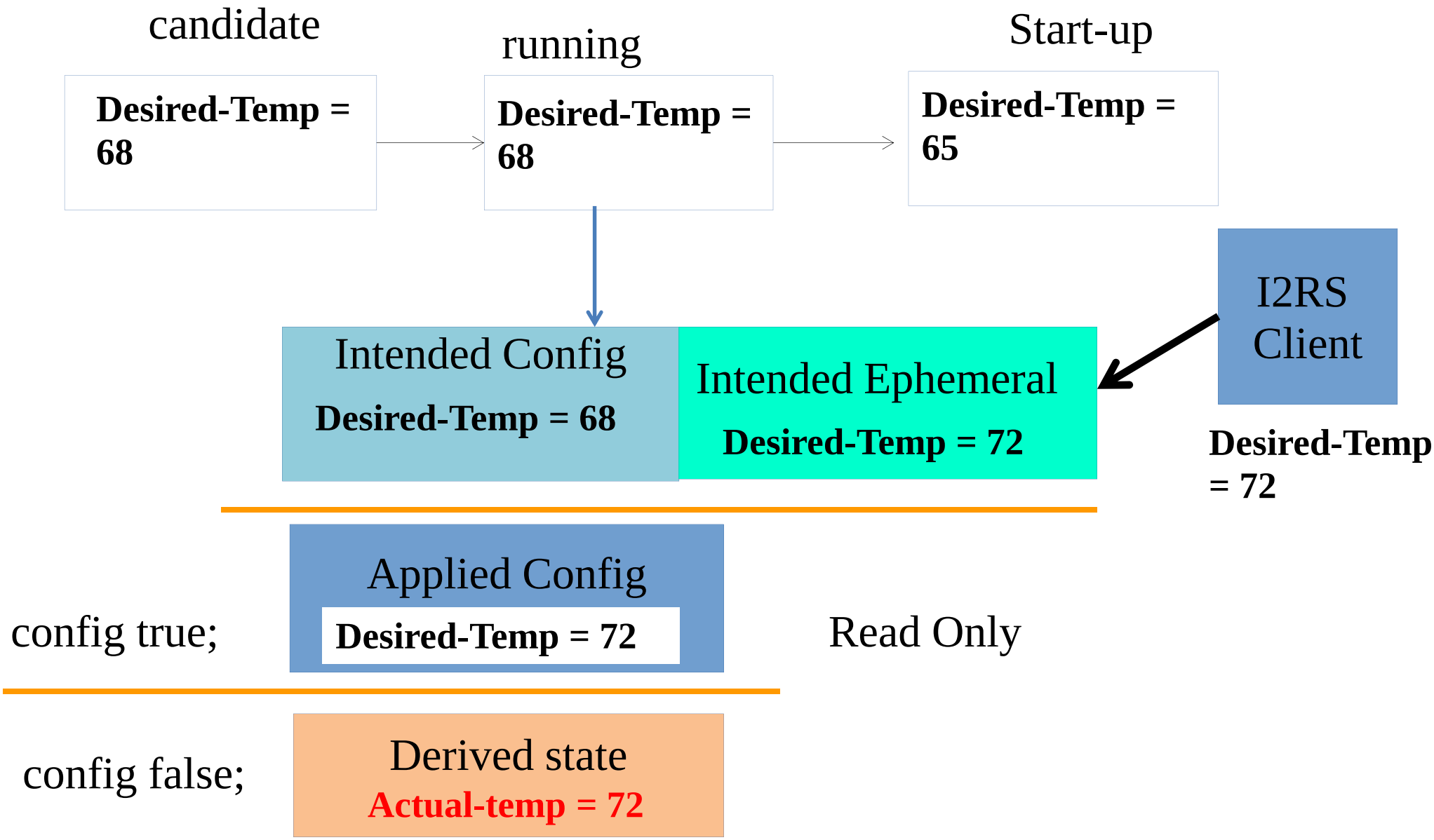
Normal Thermostat Model



Simple Thermostat + ephemeral

```
module thermostat {
//config
    leaf desired-temp {
        type int32;
         ephemeral true;
         units “degrees Celsius”;
        description “The desired temperature”;
    }
//operational State:
    leaf actual-temp {
        type int32;
        config false;
        units “degrees Celsius”;
        description “The measured temperature”;
    }
}
```

Ephemeral Thermostat Model



RESTCONF Example

RESTCONF Running Datastore Edit

PUT /restconf/data/thermostat:desired-temp

{ “desired-temp”: 68 }

RESTCONF Ephemeral Datastore Edit of config=true

PUT /restconf/data/thermostat:desired-temp?context=ephemeral

{ “desired-temp”: 72 }

RESTCONF Temperature Set

PUT /restconf/data/thermostat:desired-temp
Host: example.com
Content-Type: application/yang.data+json
{ “desired-temp”: 68 }

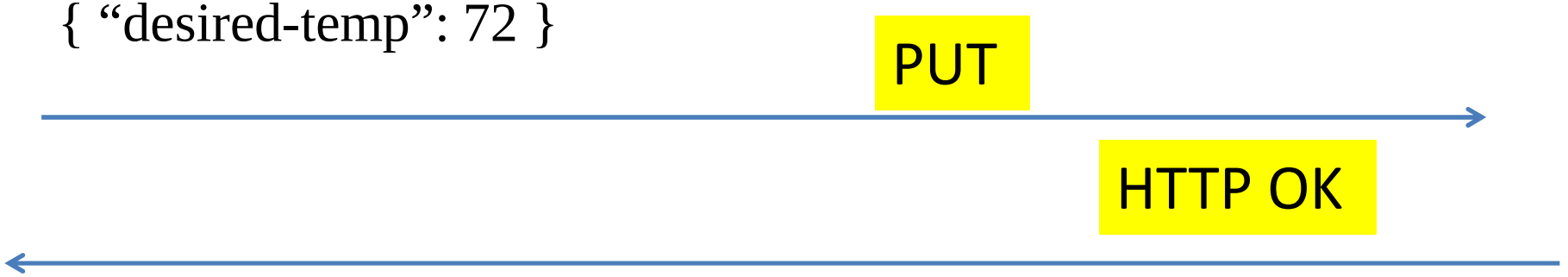
PUT

HTTP OK

HTTP/1.1 204 No Content Date:
Mon, 23 Apr 2016 17:04:00 GMT
Server: example-server
Last-Modified: Mon, 23 Apr 2017
17:04:00 GMT ETag: b27480aeda4

RESTCONF Temperature Set

PUT /restconf/data/thermostat:desired-temp?context=ephemeral
Host: example.com
Content-Type: application/yang.data+json
{ “desired-temp”: 72 }



HTTP/1.1 204 No Content Date:
Mon, 22 Apr 2016 18:04:00 GMT
Server: example-server
Last-Modified: Mon, 23 Apr 2016
18:04:00 GMT ETag: b27480aeda4

NETCONF

```
<rpc-message-id=101>
  <xmlns="urn:ietf:params:xml:ns:base:1.0">
   <edit-config>
      <target>
        <running/>
      </target>
     <config>
        <top xmlsns="http::/exaple.com/schema/1.0/thermostat/config>
            <desired-temp> 68 </desired-temp>
        </top>
      </config>
   <edit-config>
<rpc>
```

RESTCONF Temperature Set

```
<edit-config>
    <target> </running/> </target>
   <config>
     <top xmlsns="http::/exaple.com/schema/1.0/thermostat/config>
            <desired-temp> 68 </desired-temp>
     </top>
  </config>
 </edit-config>
```

rpc

Rcp-reply OK

```
<rpc-rply message-id="101"
xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
<ok/>
</rpc-reply>
```

NETCONF

```
<rpc-message-id=101>
  <xmlns="urn:ietf:params:xml:ns:base:1.0">
   <edit-config>
      <target>
        <ephemeral-datastore/>
      </target>
     <config>
        <top xmlsns="http::/exaple.com/schema/1.0/thermostat/config>
            <desired-temp> 72 </desired-temp>
        </top>
      </config>
   <edit-config>
<rpc>
```

RESTCONF Temperature Set

```
<edit-config>
    <target> <ephemeral-datastore/></target>
   <config>
     <top xmlsns="http::/exaple.com/schema/1.0/thermostat/config>
            <desired-temp> 72 </desired-temp>
     </top>
  </config>
 </edit-config>
```

rpc

Rcp-reply OK

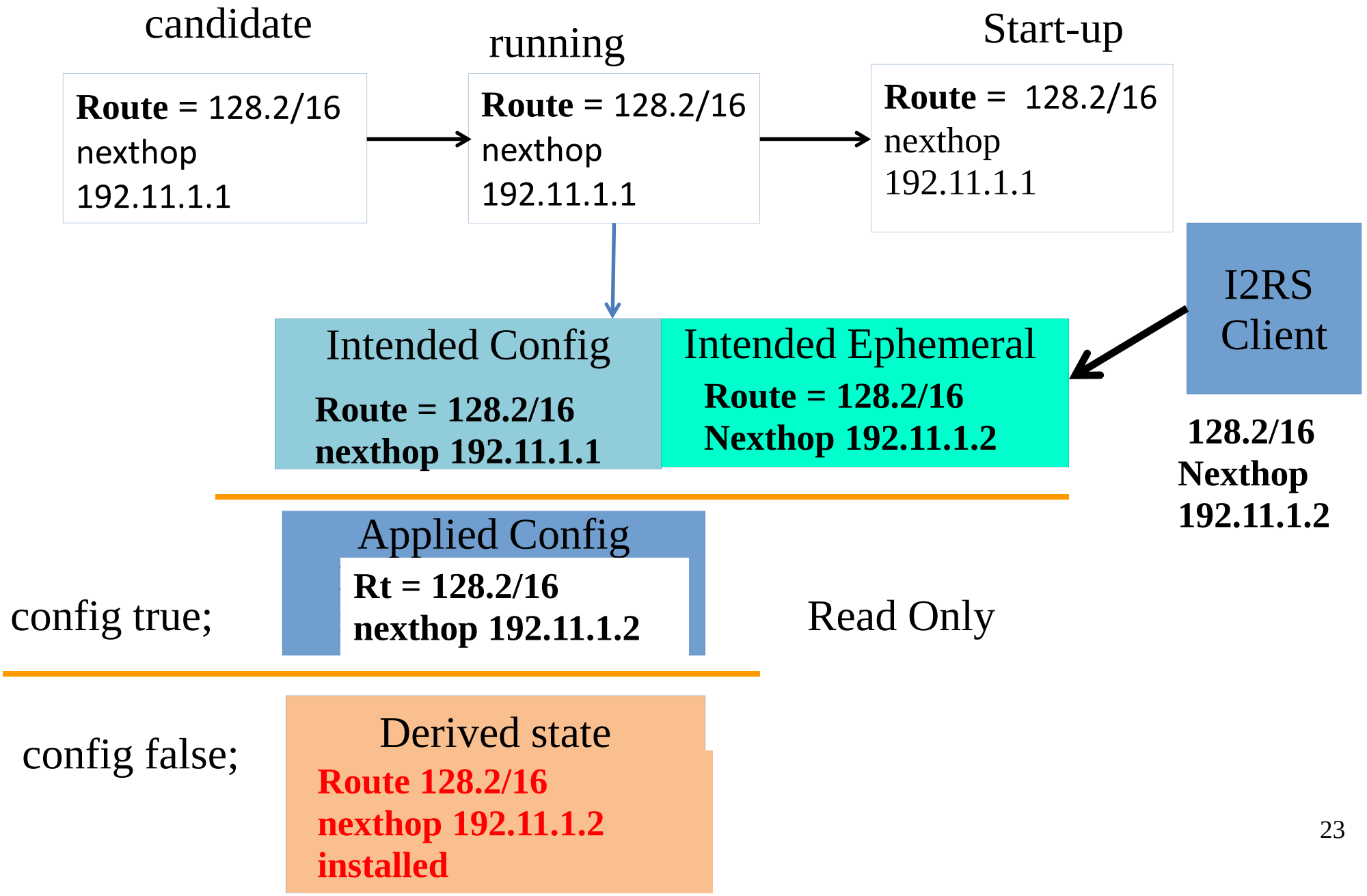
```
<rpc-rply message-id="101"
xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
<ok/>
</rpc-reply>
```

I2RS RIB Route add

- Circumstance
  - 128.2/16 with nexthop 192.11.1.1 – added by netconf config
  - 128.2/16 with nexthop 192.11.1.2 – added by I2RS Client

- How I2RS adds route
  - rpc route-add
  - (RIB, route-list {prefix, attributes, vendor-attributes, nexthop)

Ephemeral Route



Route

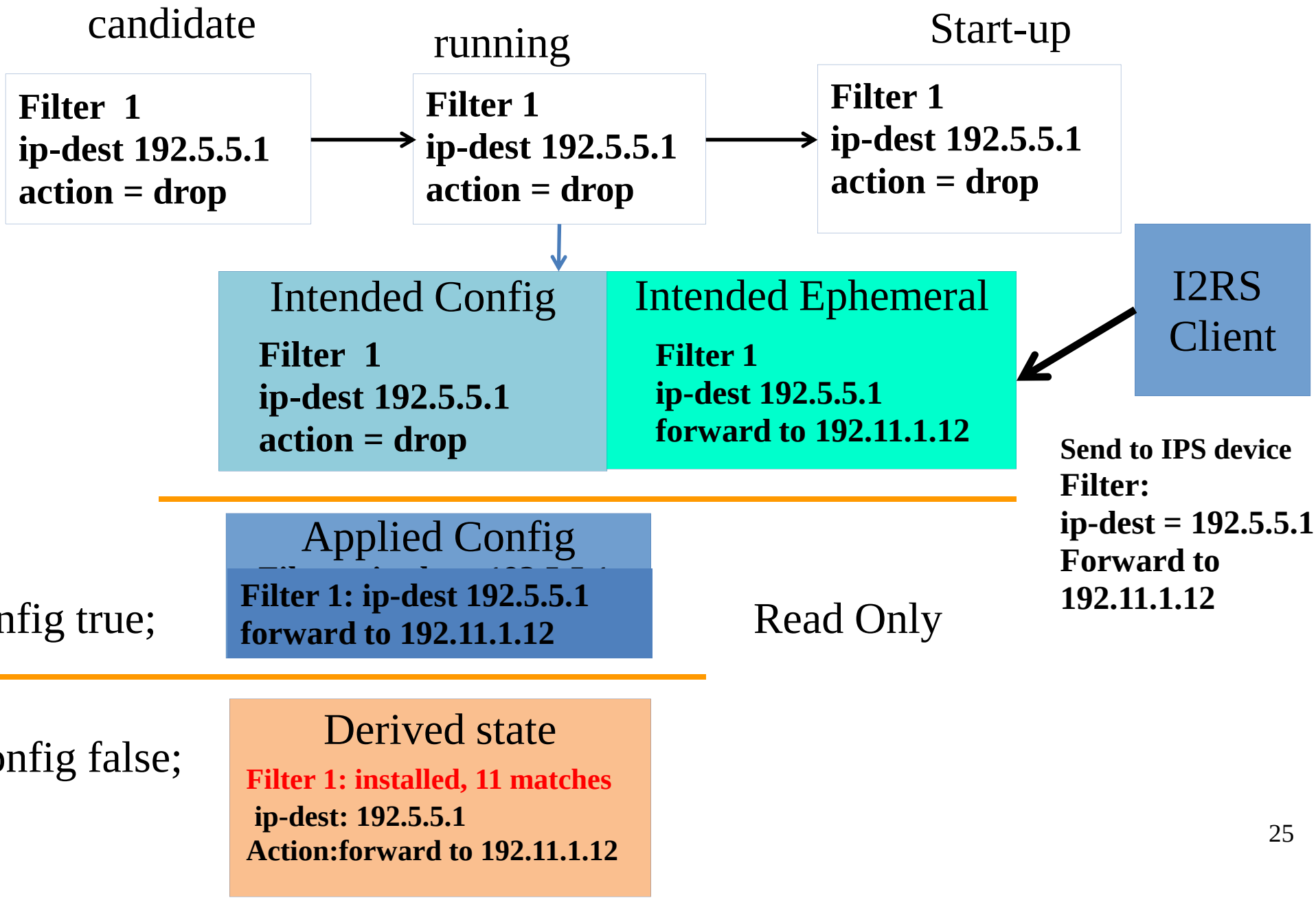
```
module i2rs-rib {
    …
    container routing-instance {
     …
      list rib-list {
       …
         list route-list {
             key “route-index”;
             uses route;
         }
    }
}
```

operational data

Extensions

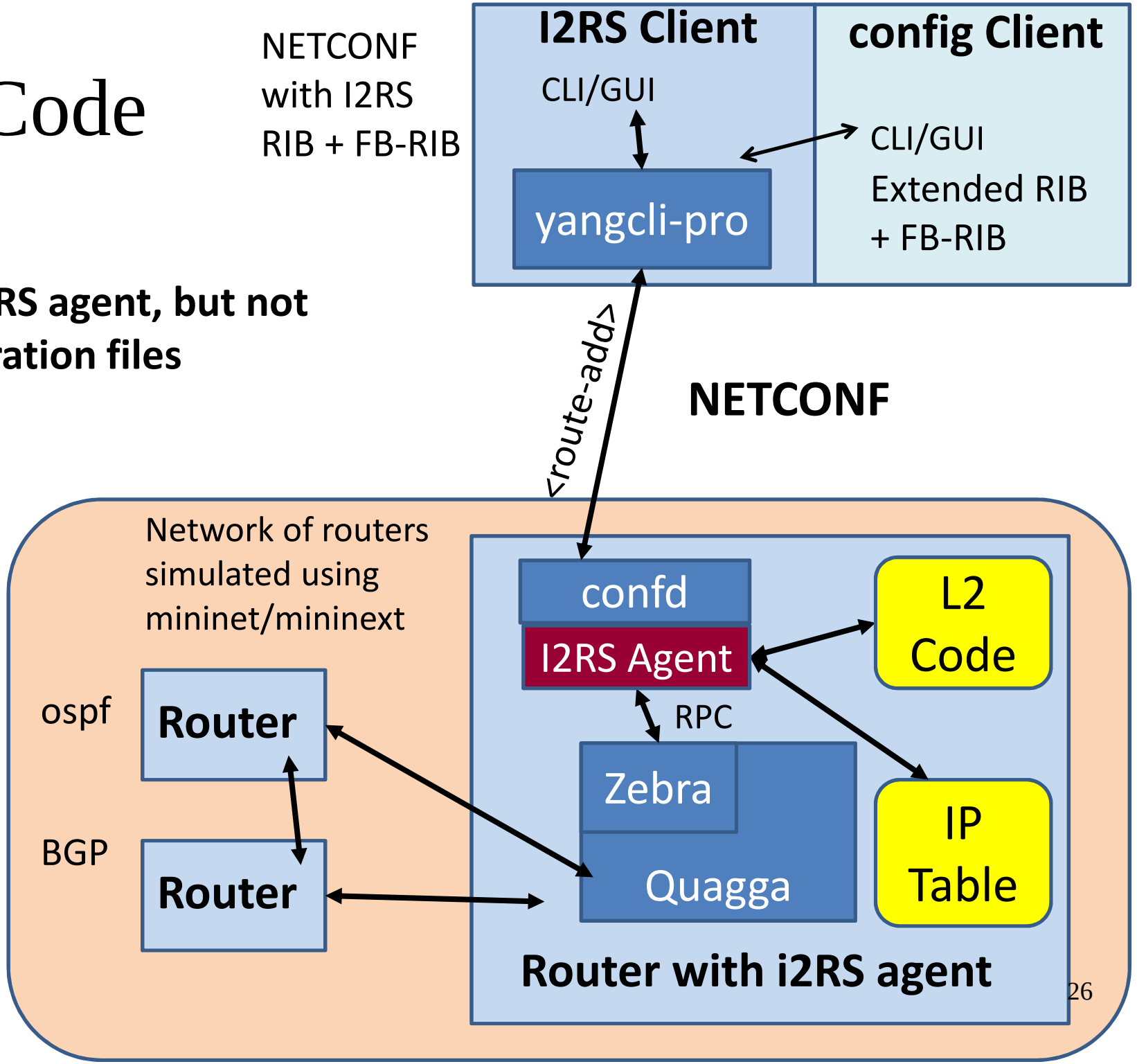
```
grouping route  {
  description
    “The common attribute used for all routes;”
   uses routeg-prefix;
    container nexthop {
           uses nexthop;
        }
        container route-statistics {
            leaf route-state {
              type route-state-def;
              config false;    /* operational state */
             }
            leaf route-installed state {
              type route-installed-state def;
              config false;
             }
            leaf route-reason {
             type route-reason-def;
              config false;
             }
         }
      container router-attributes {
          uses router-attributes;
       }
     container route-vendor-attributes
         uses route-vendor attributes;
   }
```

Ephemeral Filter



I2RS Code

NETCONF
with I2RS
RIB + FB-RIB



Store in I2RS agent, but not in configuration files

Git Hub information for Code

- https://github.com/i2rs-wg
  - WG documents, code, yang modules
- https://github.com/i2rs-wg/hackathon-code
  - Interim work repository @ietfs
- https://github.com/i2rs-wg/code-implementations
  - Ready for you to try
  - ConfD requires Cisco Dev listen

Capability Specification for NETCONF/RESTCONF

NETCONF (1)

- **Capability: ephemeral-datastore**
- *Overview:*
  - Not in intended to survive a reboot, and Never locked
  - Normal Case: Priority of Ephemeral Pane higher than configuration Pane. Error if two clients write same variable (priority scopes error)
  - No Rollback on ephemeral
  - Ephemeral under non-ephemeral; No non-ephemeral under Ephemeral
  - NETCONF <hello> - but no non-ephemeral under ephemeral modules, sub-trees, node
- **Dependencies:**
  - Yang: ephemeral flag, ephemeral-validation
  - Yang modules – must support notification of write conflicts (Config/ephemeral and Priority)

NETCONF (2)

- **New operations :**
  - Link-ephemeral <target-config>
  - Bulk-write – [Not sure if need or if rpc better approach]
- **Modifications**
  - <get-config> <get> - target changes
  - <edit-config> - <merge-priority> <replace-priority>
    - <default-operations>: <merge-priority> or <replace-priority>
    - <error-option> - “all-or-nothing” == “rollback-on-error
  - <unlock> <lock> - not supported
  - <confirmed commit> - not supported
  - <close-session> <kill-session> - target change
  - <Writeable-running> and <candidate> – support ephemeral (?)
  - Validate – supports ephemeral data store with three key words:

    Syntax, reduced, full-check

Route Add rpc

- Screen shot of route add rpc added here

Filter-Add RPC here

- Screen shot of filter-add rpc added here

RESTCONF (1)

- **Capability: ephemeral-context**
- *Overview:*
  - Same as netconf except RESTCONF Context
- **Dependencies:**
  - Yang: ephemeral flag, ephemeral-validation
  - Yang modules – must support notification of write conflicts (Config/ephemeral and Priority)
  - I2RS Yang modules support: Yang patch and Yang module library

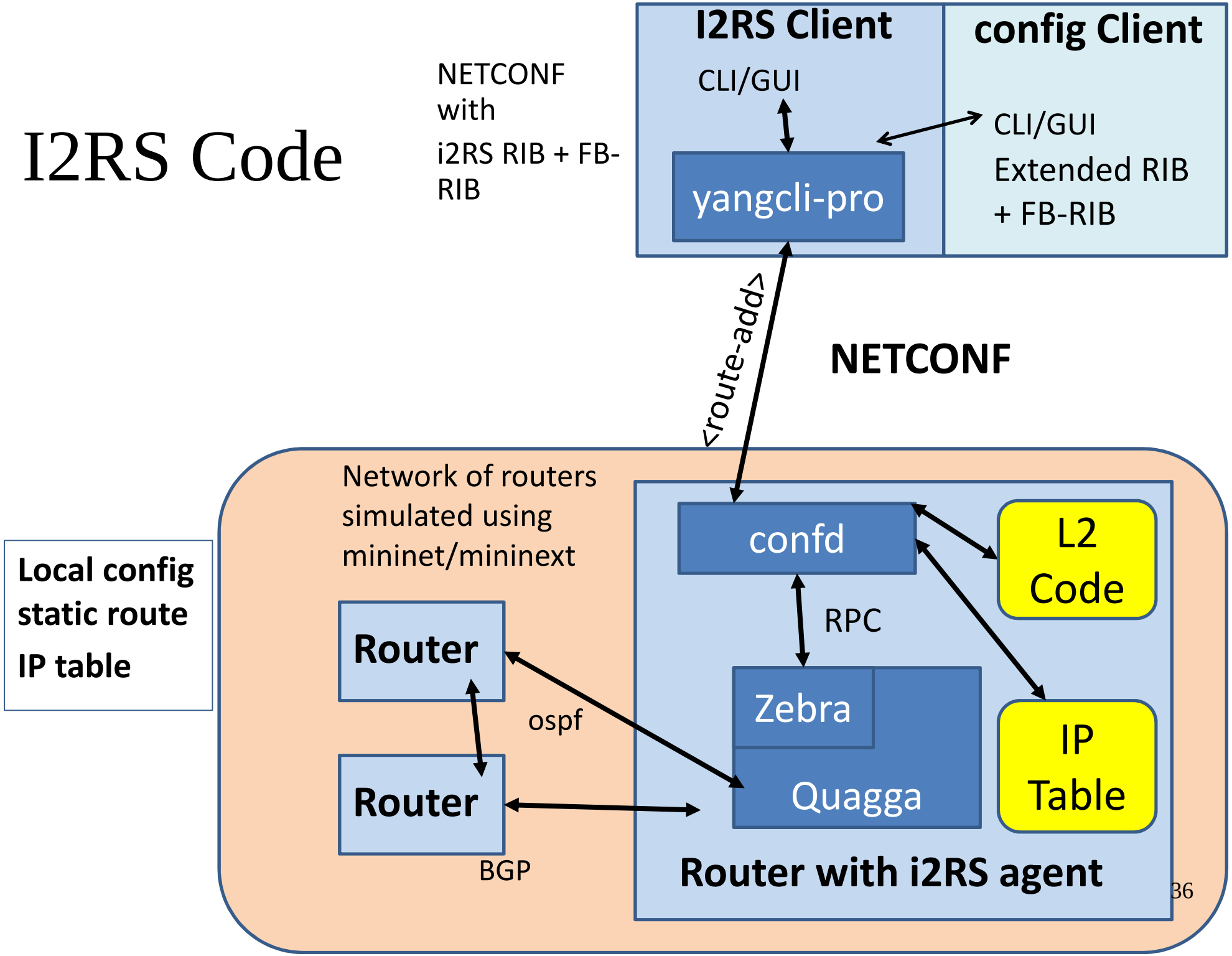
RESTCONF (2)

- **Data resources**
  - +restconf/data – ephemeral data tree with edit collision features of timestamp and Entity Tag
  - Assumption: Entity can be split to client-priority
- **Modifications**
  - Options: provide indication of ephemeral state in data modules, sub-modules [ietf-netconf-yang-library]
  - HEAD – returns ephemeral or config context
  - GET - determines if ephemeral or config
  - POST/PUT/PATCH - context=ephemeral:

    uses ephemeral rules + validity + priority + no config below ephemeral
  - DELETE – ephemeral context
  - Query – Allows to filter by ephemeral
  - Error/Notifications – must interact with pub/sub push [ietf-netconf-yang-push]
  - Log and traceability -

Hands-on questions

I2RS Code


NETCONF
with
i2RS RIB + FB-RIB
I2RS Client
CLI/GUI
yangcli-pro
config Client
CLI/GUI
Extended RIB
+ FB-RIB
<route-add>
NETCONF
Network of routers
simulated using
mininet/mininext
Local config
static route
IP table
confd
L2
Code
RPC
Router
Router
ospf
BGP
Zebra
Quagga
IP
Table
Router with i2RS agent