
OPEN
Compute Project
OCP U.S. SUMMIT 2017
Santa Clara, CA

Experiences in operating a mixed OCP/non-OCP data center network

Starsky H.Y. Wong, Facebook.
wong@fb.com

OPEN HARDWARE. OPEN SOFTWARE. OPEN FUTURE.

OPEN Compute Project

Agenda

1. Overview of Facebook Network
2. Robotron: Our Top-Down Network Management
3. Applying to FBOSS (Wedges + Backpack)
4. Takeaways

Scale of Facebook Community

~1.86 Billion

on Facebook Monthly

~1.2 Billion

on WhatsApp Monthly

~600 Million

on Instagram Monthly

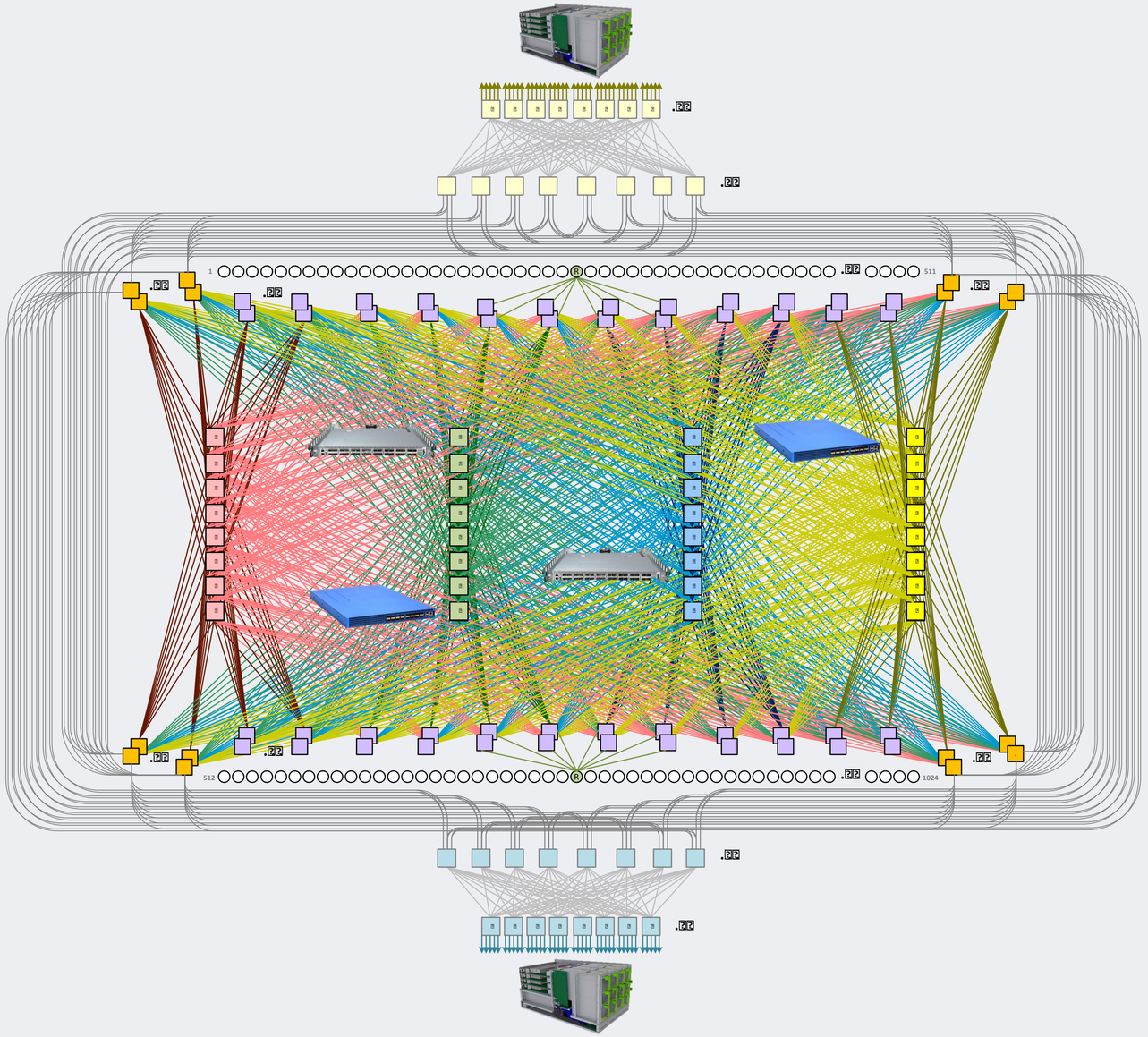
~1 Billion

on Messenger Monthly

Network Management at Facebook

Goals

- Keep FB network healthy
- Support FB network evolution



Network Management at Facebook

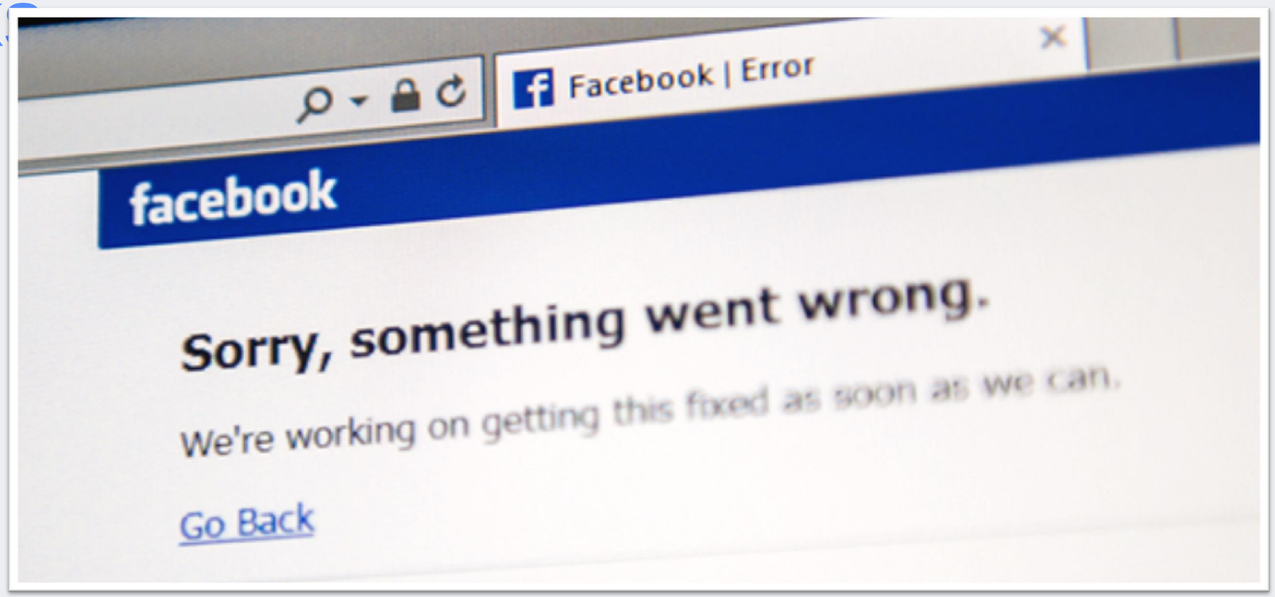
Example Management Tasks

- Router/switch turnup
- Circuit turnup
- Circuit migration
- Router/switch rename
- ...etc.

Network Management at Facebook

Example Management Tasks

- Router/switch turnup
- Circuit turnup
- Circuit migration
- Router/switch rename
- ...etc.



Network Management at Facebook

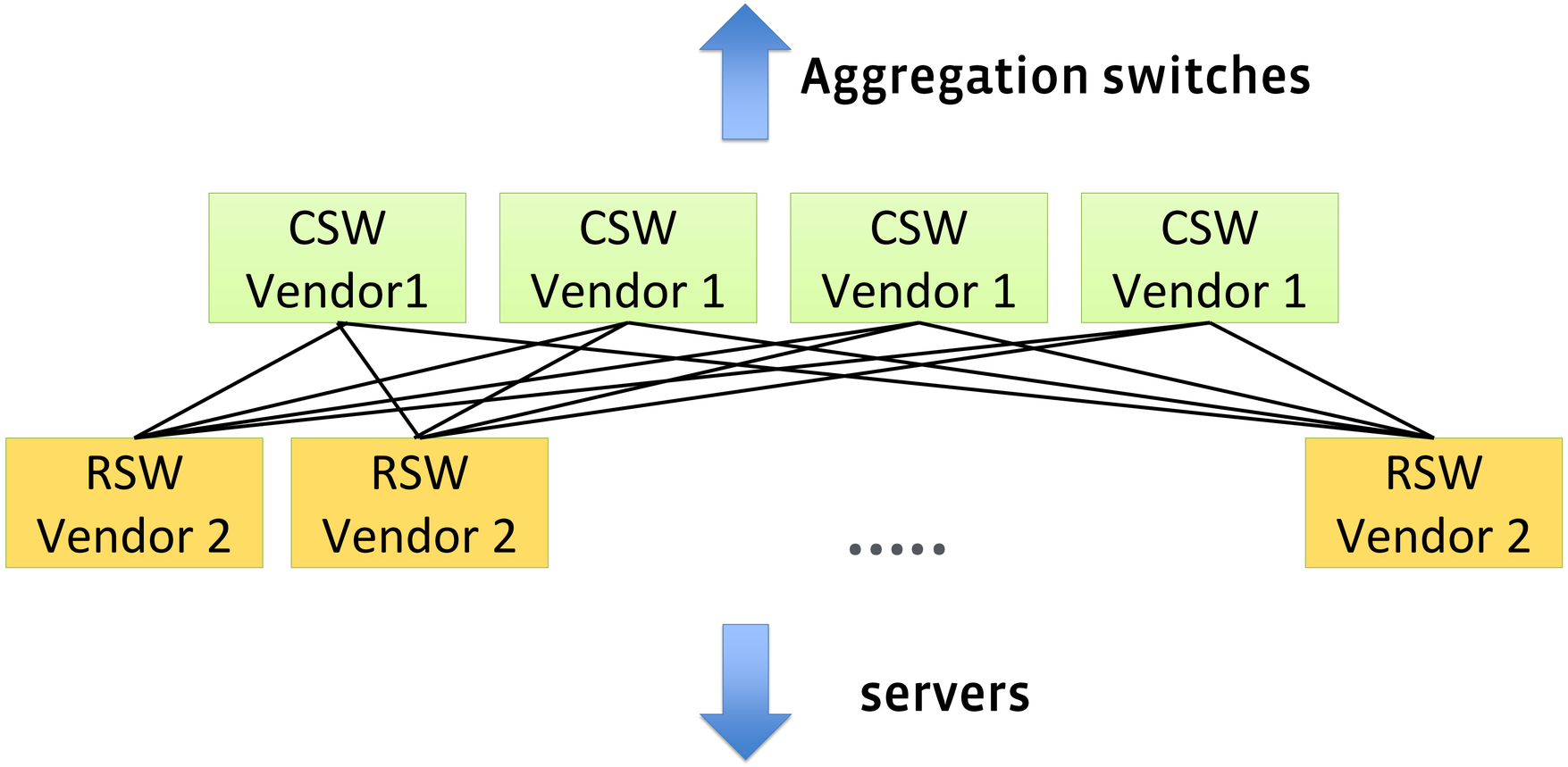
Challenges

- Multi Vendors

Multi Vendors in Data Center

Illustrative example (Not an actual topology)

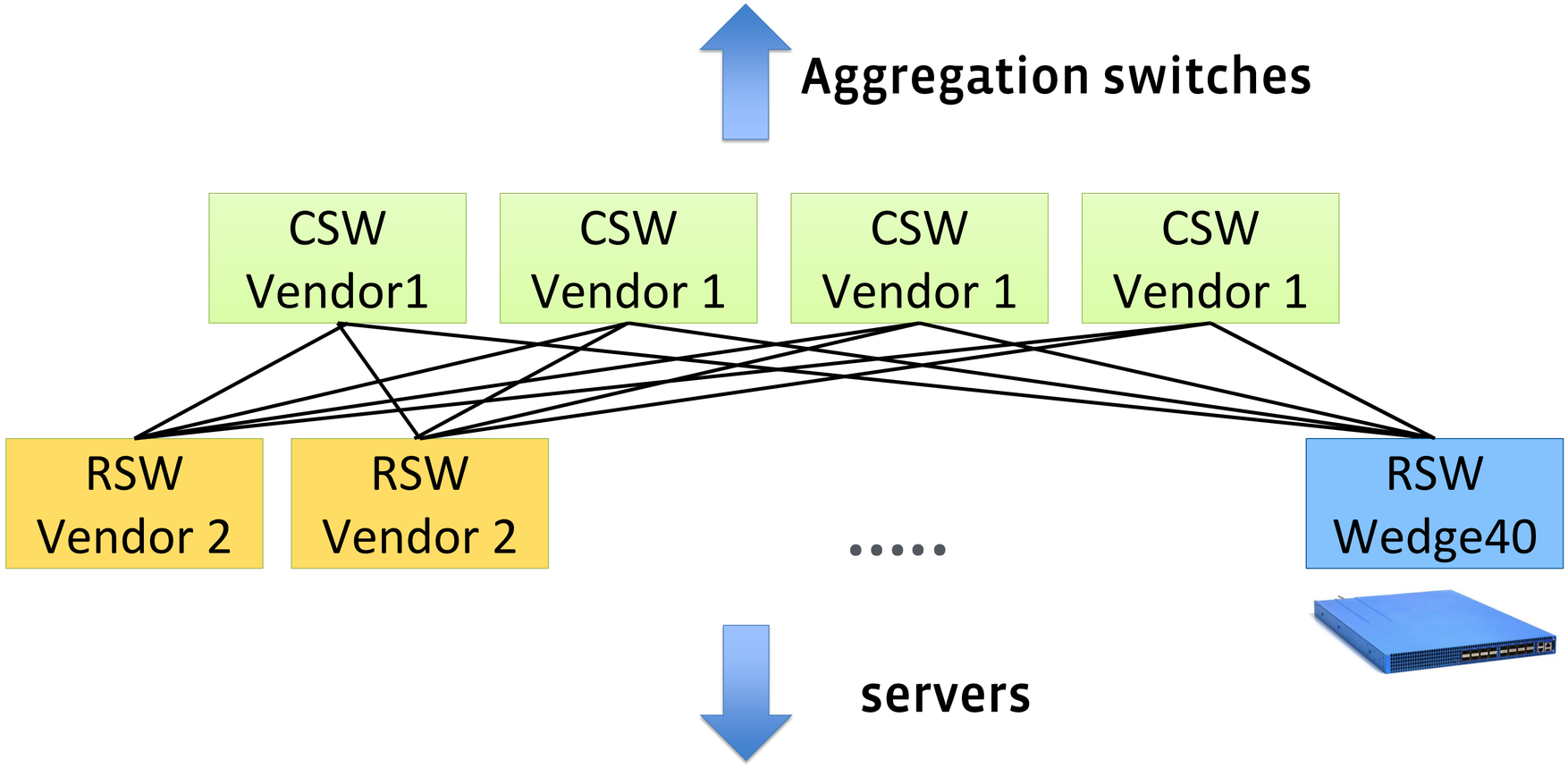
CSW: Cluster switch
RSW: Rack switch



Multi Vendors in Data Center

Illustrative example (Not an actual topology)

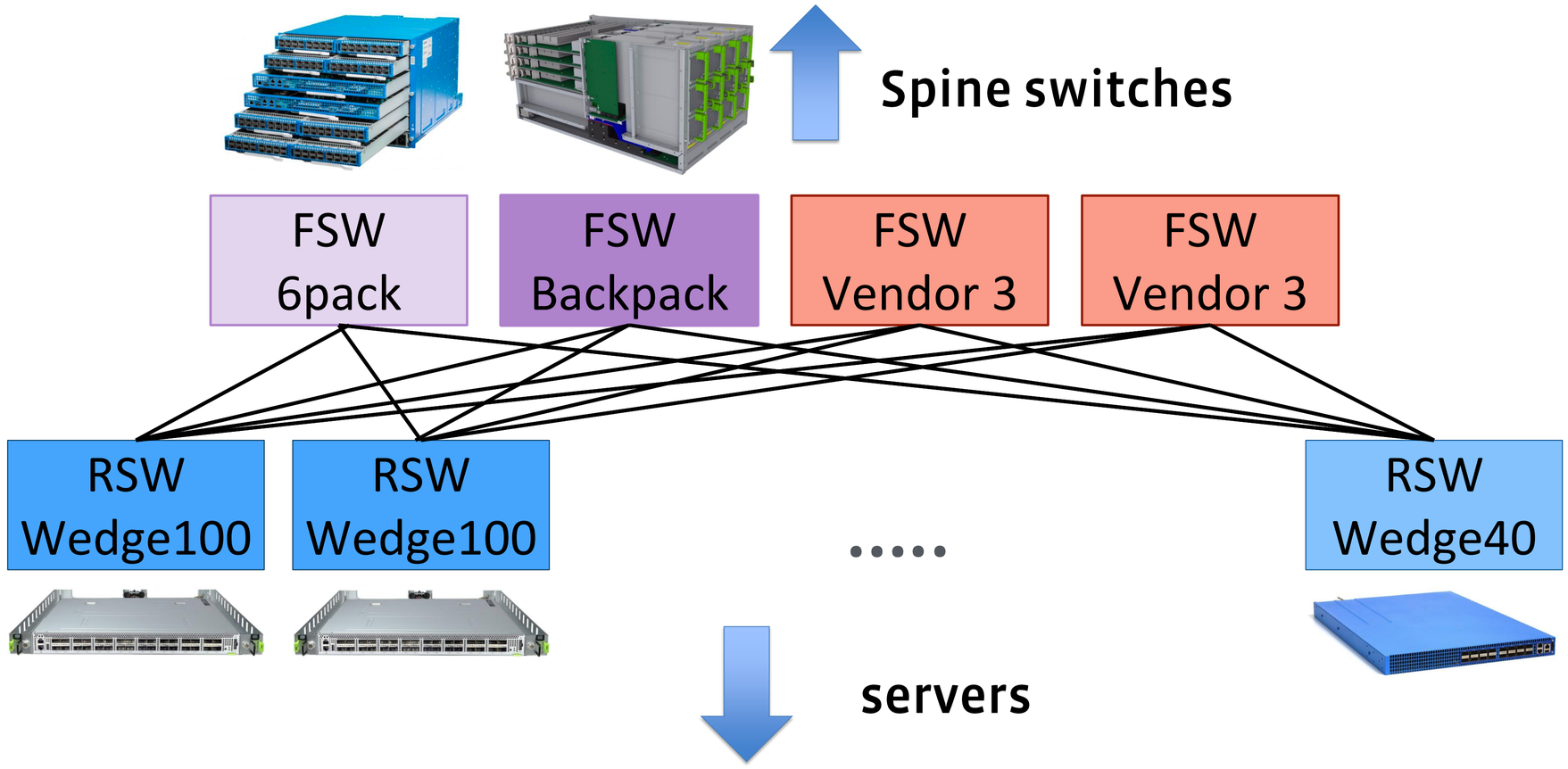
CSW: Cluster switch
RSW: Rack switch



Multi Vendors in Data Center

Illustrative example (Not an actual topology)

FSW: Fabric switch
RSW: Rack switch



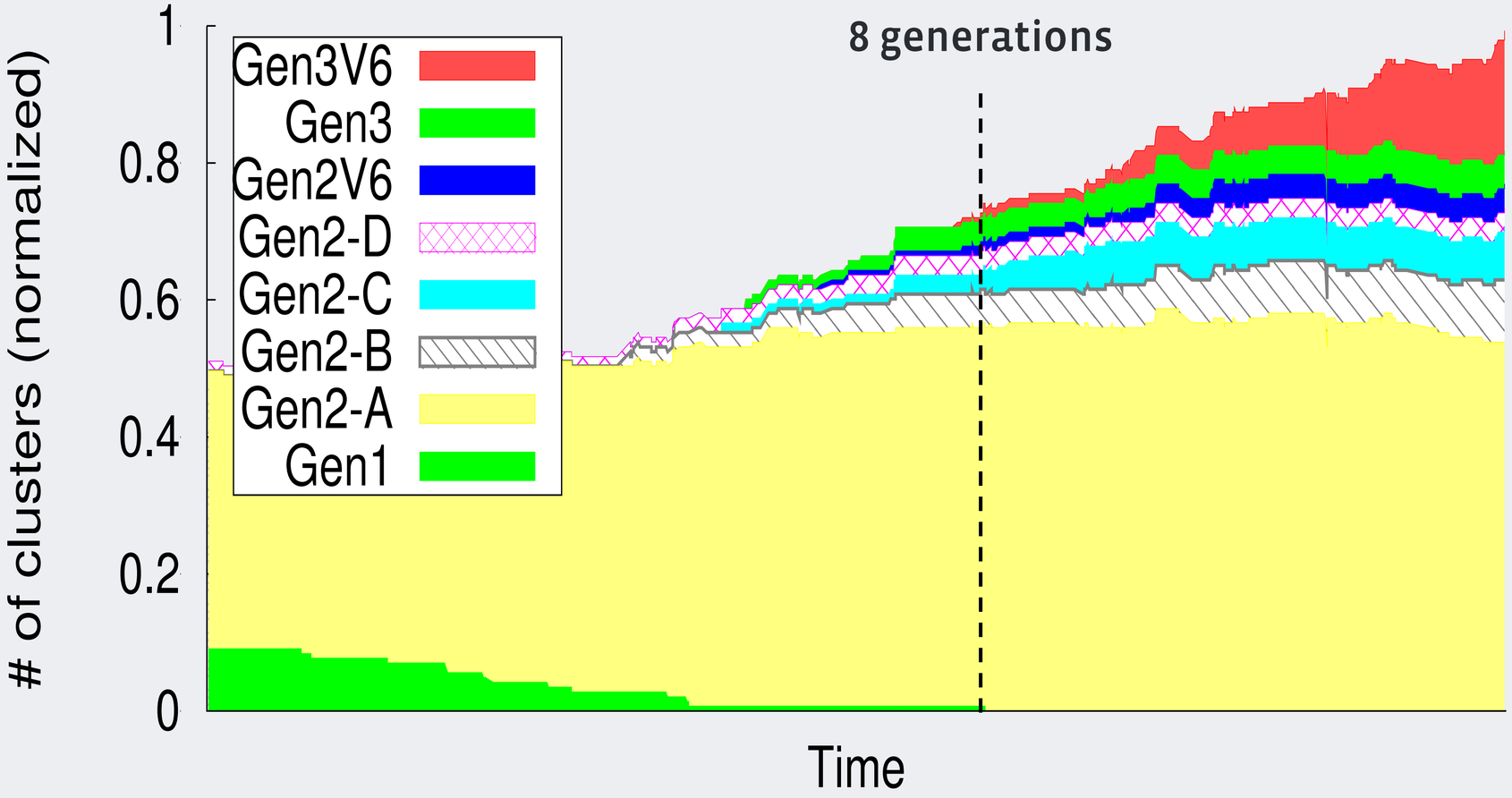
Network Management at Facebook

Challenges

- Multi Vendors
- Versioning

Hybrid architectures in Data Center

Multiple versions of FB cluster architectures co-exist



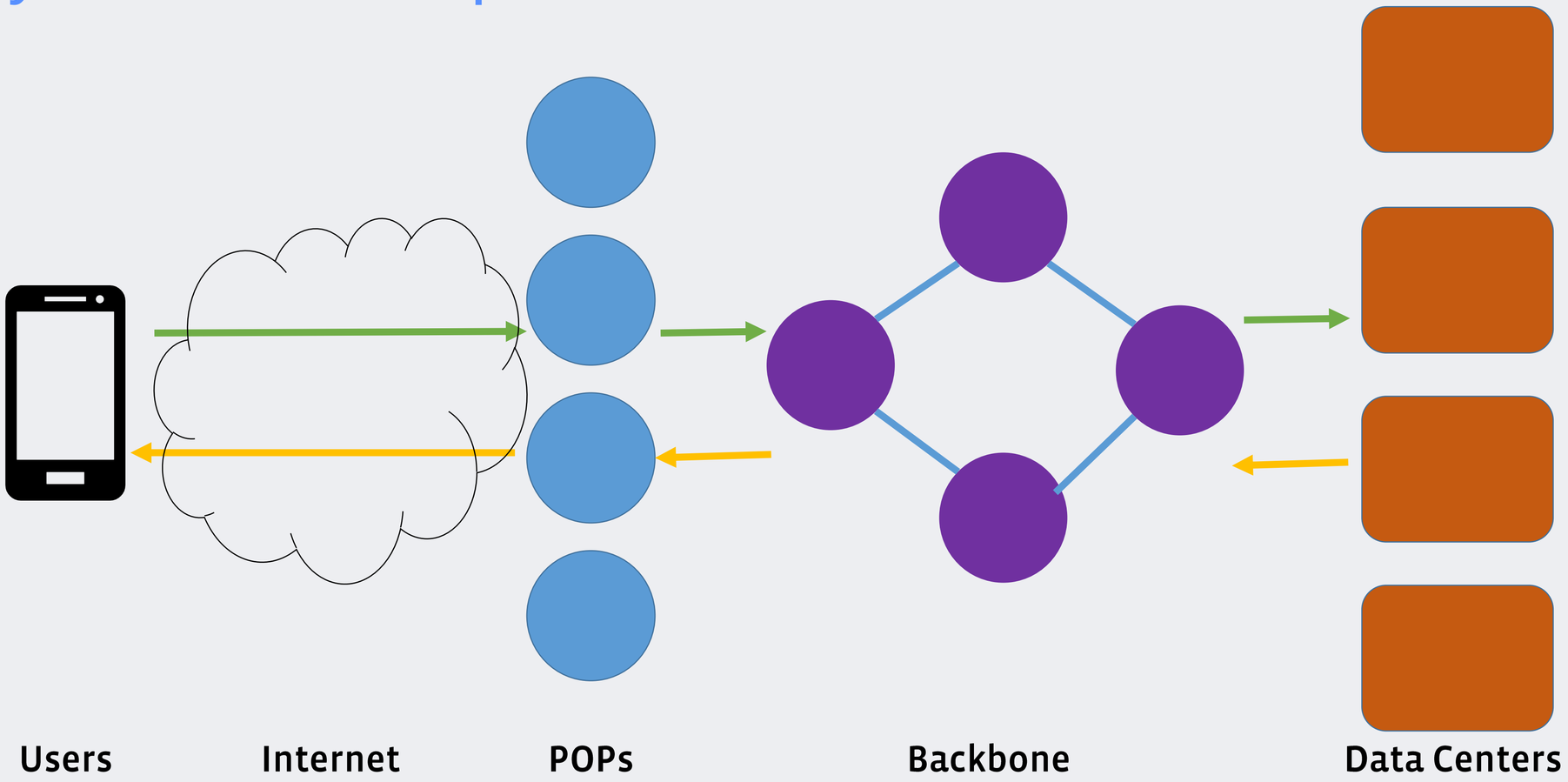
Network Management at Facebook

Challenges

- Multi Vendors
- Versioning
- Distributed Configurations
- Dependency

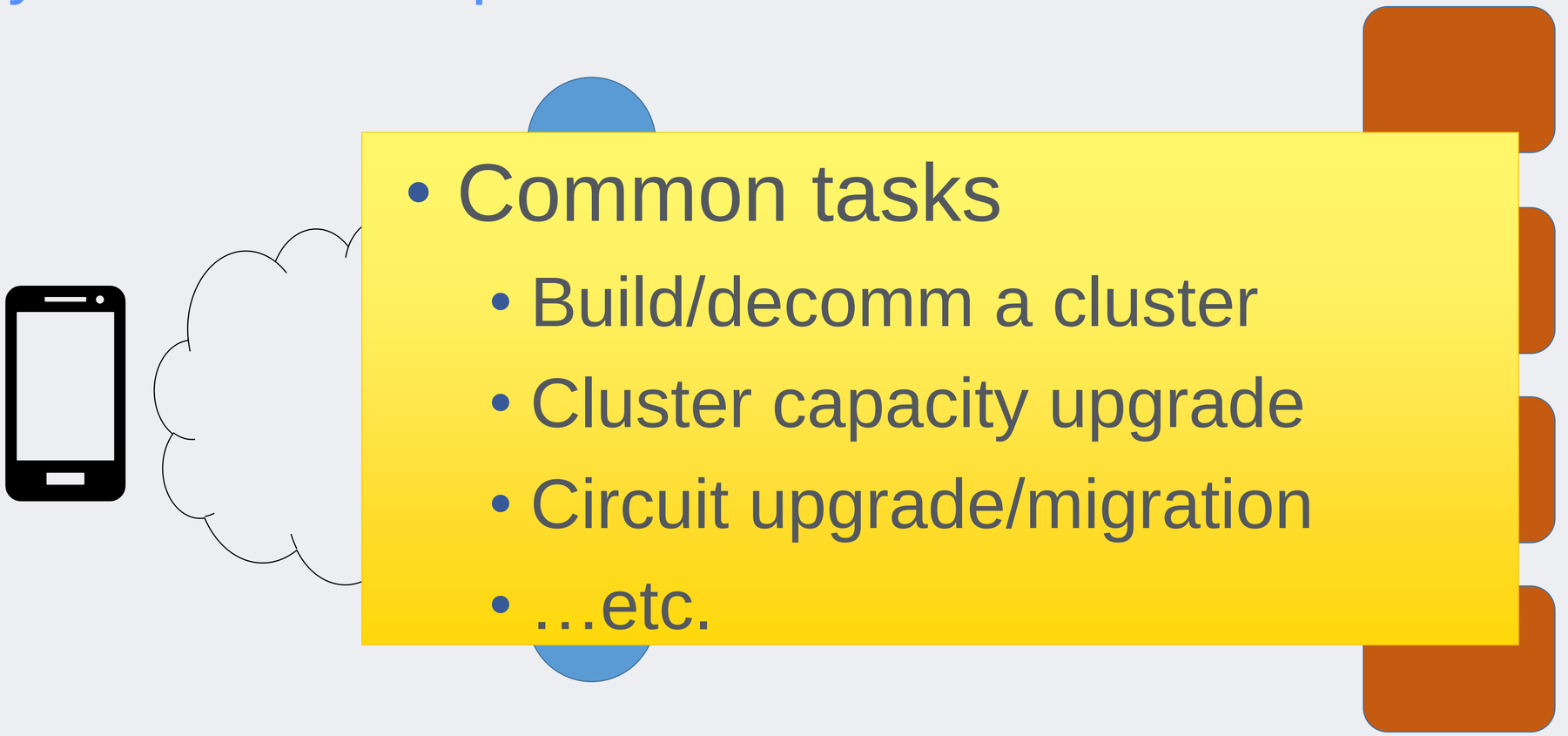
Overview of Facebook's Network

Lifecycle of user requests



Overview of Facebook's Network

Lifecycle of user requests

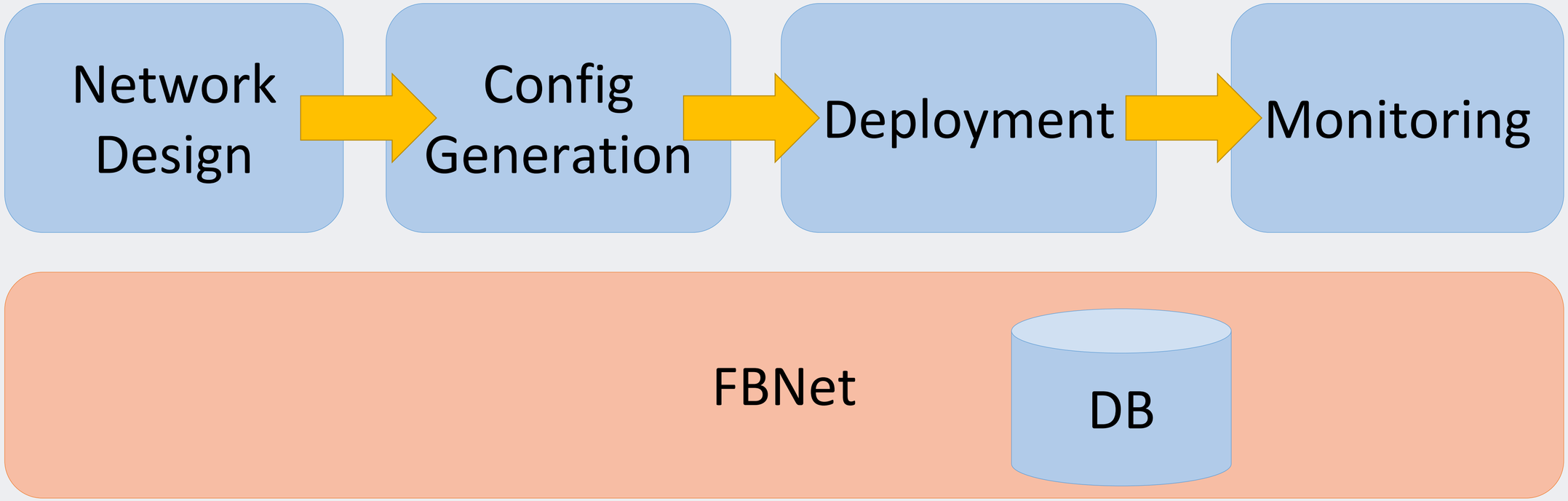


Agenda

1. Overview of Facebook Network
2. Robotron: Our Top-Down Network Management
3. Applying to FBOSS (Wedges + Backpack)
4. Takeaways

Robotron: “Top-Down” Network Management System@FB

Overview

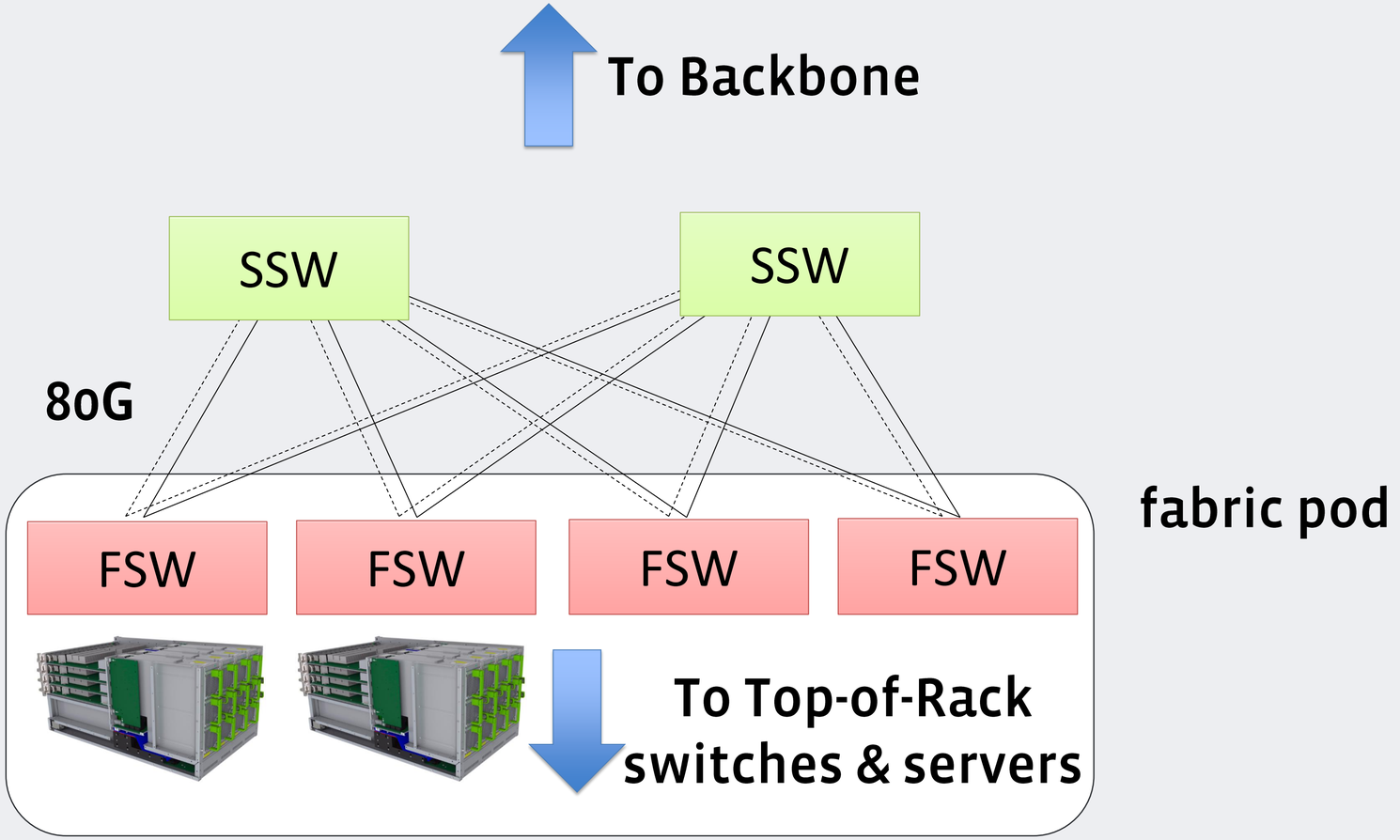


https://research.fb.com/publications/robotron-top-down-network-management-at-facebook-scale/

FBNet: Modeling the Network

Illustrative example of fabric pod (Not an actual topology)

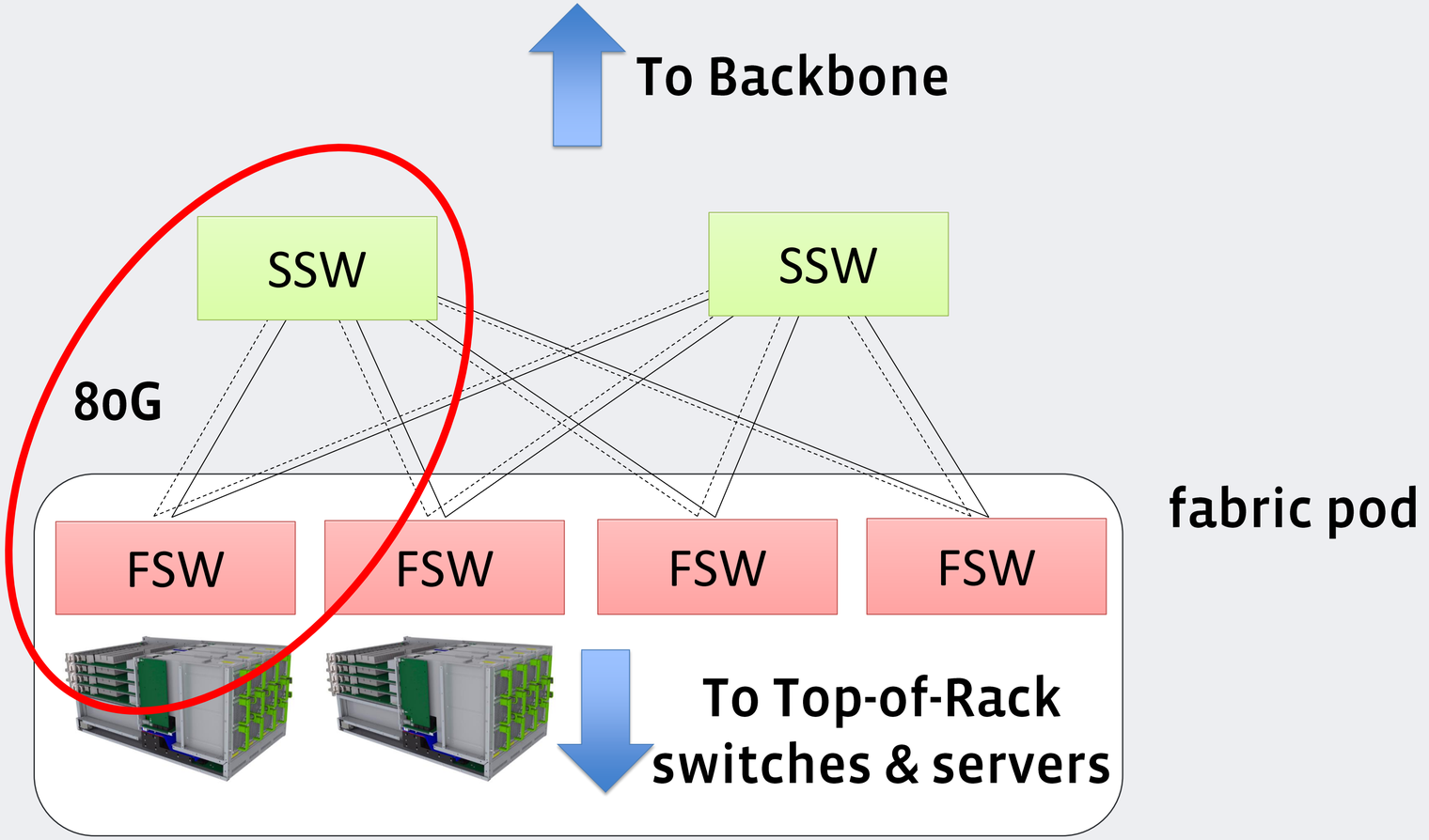
SSW: Spine switch
FSW: Fabric switch
BGP session



FBNet: Modeling the Network

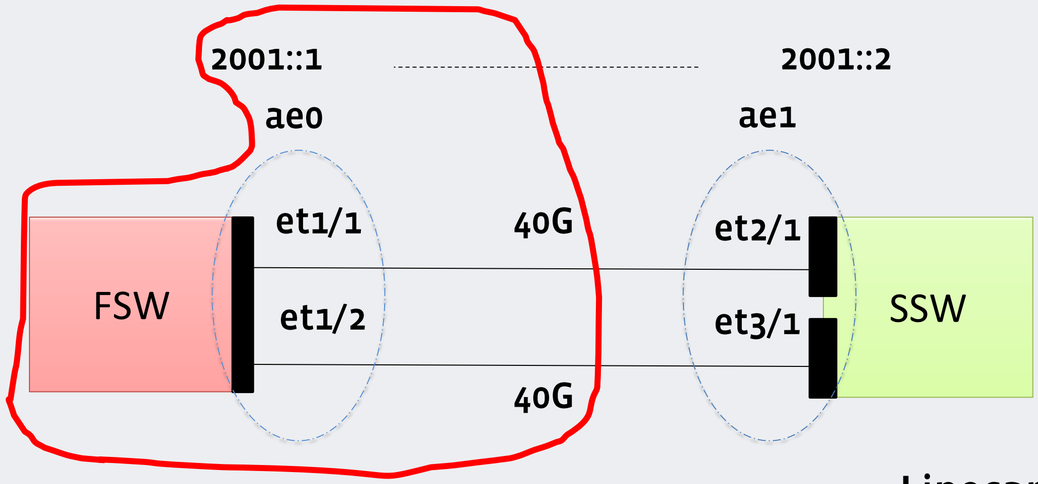
Illustrative example of fabric pod (Not an actual topology)

SSW: Spine switch
FSW: Fabric switch
BGP session



FBNet: Modeling the Network

Object



BGP session

Linecard

Circuit

Networkswitch

Linecard

PhysicalInterface

PhysicalInterface

Circuit

AggregatedInterface

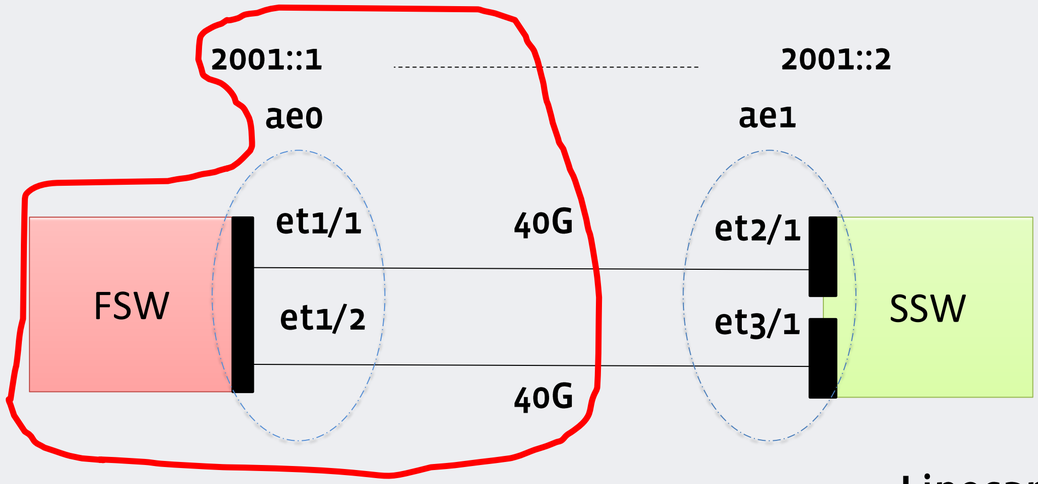
V6Prefix

BgpV6Session

Circuit

FBNet: Modeling the Network

Value Fields



Networkswitch

name=FSW

Linecard

slot=1
model=X

PhysicalInterface

name=et1/1

PhysicalInterface

name=et1/2

AggregatedInterface

name=ae0

V6Prefix

prefix=2001::1

Circuit

speed=40G

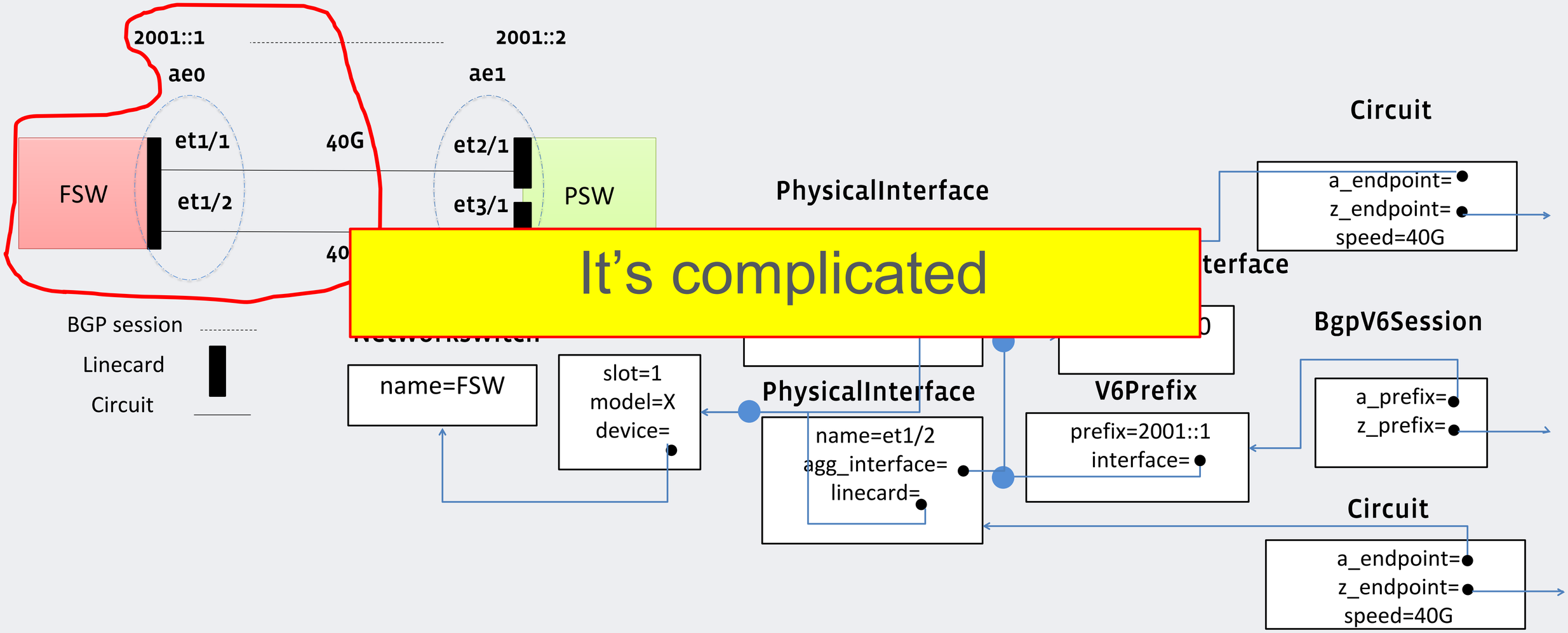
BgpV6Session

Circuit

speed=40G

FBNet: Modeling the Network

Relationship Fields



FBNet Model Snippet

```
class Physical_Interface(Interface):
  linecard = related_to(Linecard)
  agg_interface = related_to(
      AggregatedInterface)
```

FBNet Model Snippet

Related models

```
class Physical_Interface(Interface):
  linecard = related_to(Linecard)
  agg_interface = related_to(
      AggregatedInterface)
```

FBNet Model Snippet

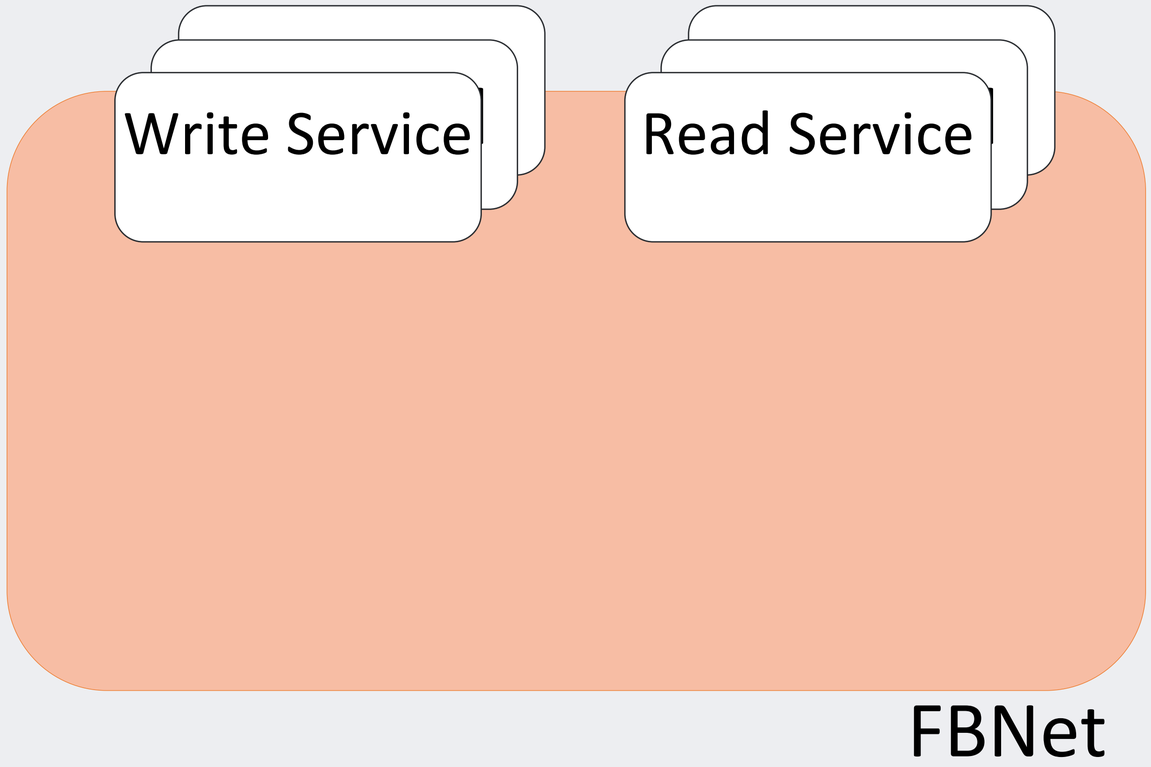
Model inheritance

```
class Physical_Interface(Interface):
  linecard = related_to(Linecard)
  agg_interface = related_to(
      AggregatedInterface)
```

FBNet: Architecture

API Layer

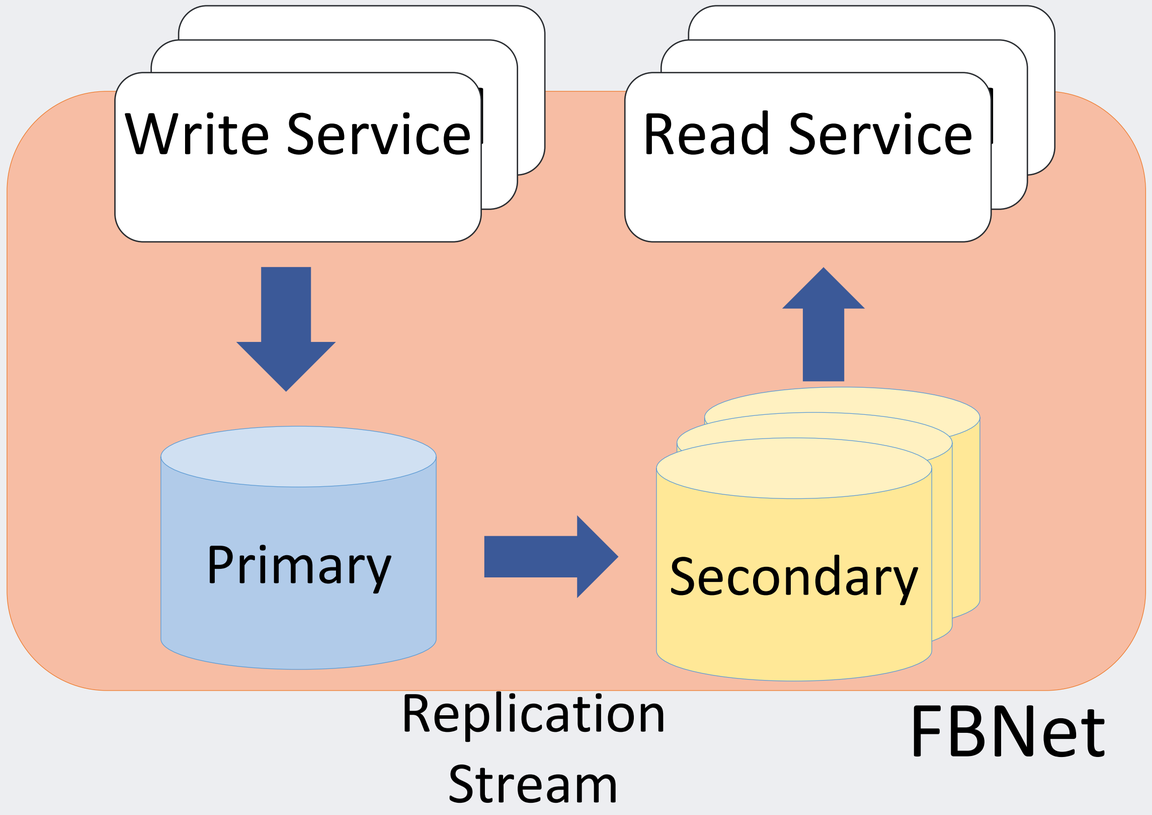
- RPC services
  - Read: fine-grained per-model query
  - Write: task-based
- High Availability: Multiple replicas per DC



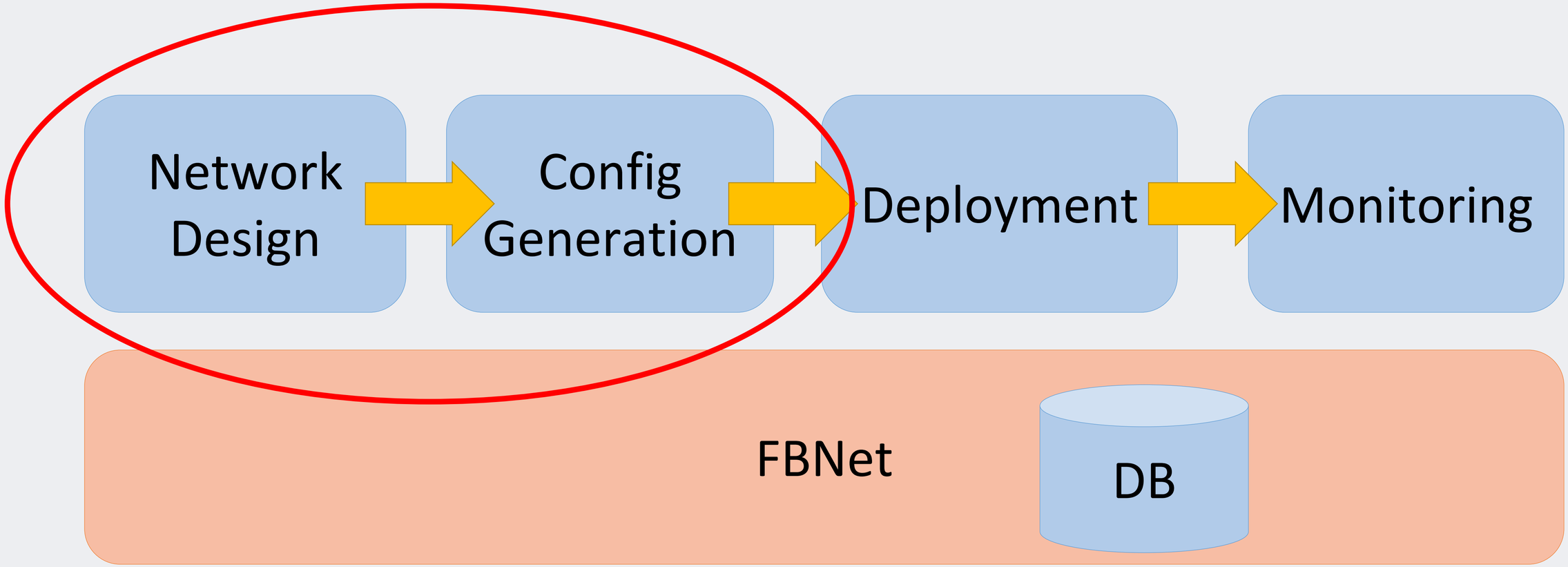
FBNet: Architecture

Storage Layer

- 1 primary, multiple secondary DBs
- Scalability: 1 secondary DB per DC



Robotron’s management life cycle



Network Design

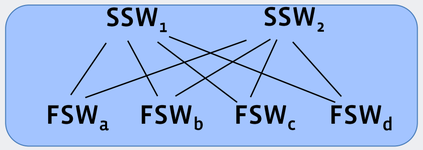
Design intent → FBNet objects

Template for a fabric pod

```
Cluster(
  devices={
    SSW: DeviceSpec(
      hardware=["Switch_Vendor1",
      a_device=SSW,
      z_device=FSW,
      pifs_per_agg=2,
      ip=V6)
  ]
)
```

94 objects across 7 models, 100+ relationship fields

FBNet objects



NetworkSwitches: 6
Circuits: 16
PhysicalInterfaces: 32
AggregatedInterfaces: 16
V6Prefixes: 16
BgpV6Sessions: 8

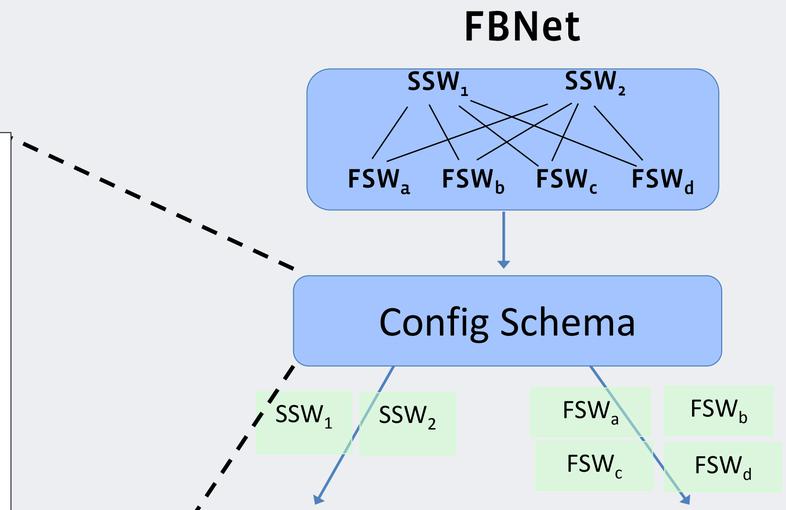
Config Generation

FBNet objects → Device configs

```
Class Device {
  set <AggregatedInterface> aggs,
}

Class AggregatedInterface {
  str name,
  int number,
  str v4_prefix,
  str v6_prefix,
  set <PhysicalInterface> pifs,
}

Class PhysicalInterface {
  str name,
}
```



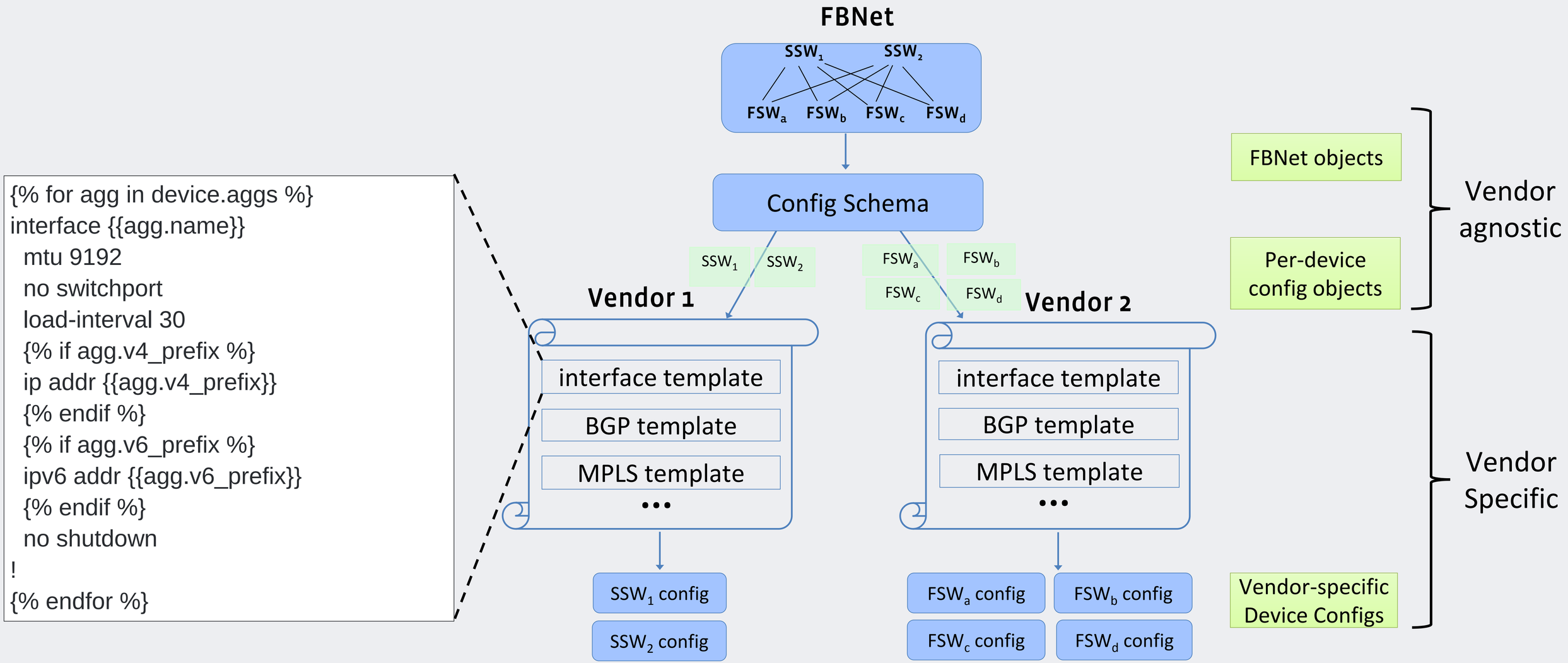
FBNet objects

Per-device config objects

Vendor agnostic

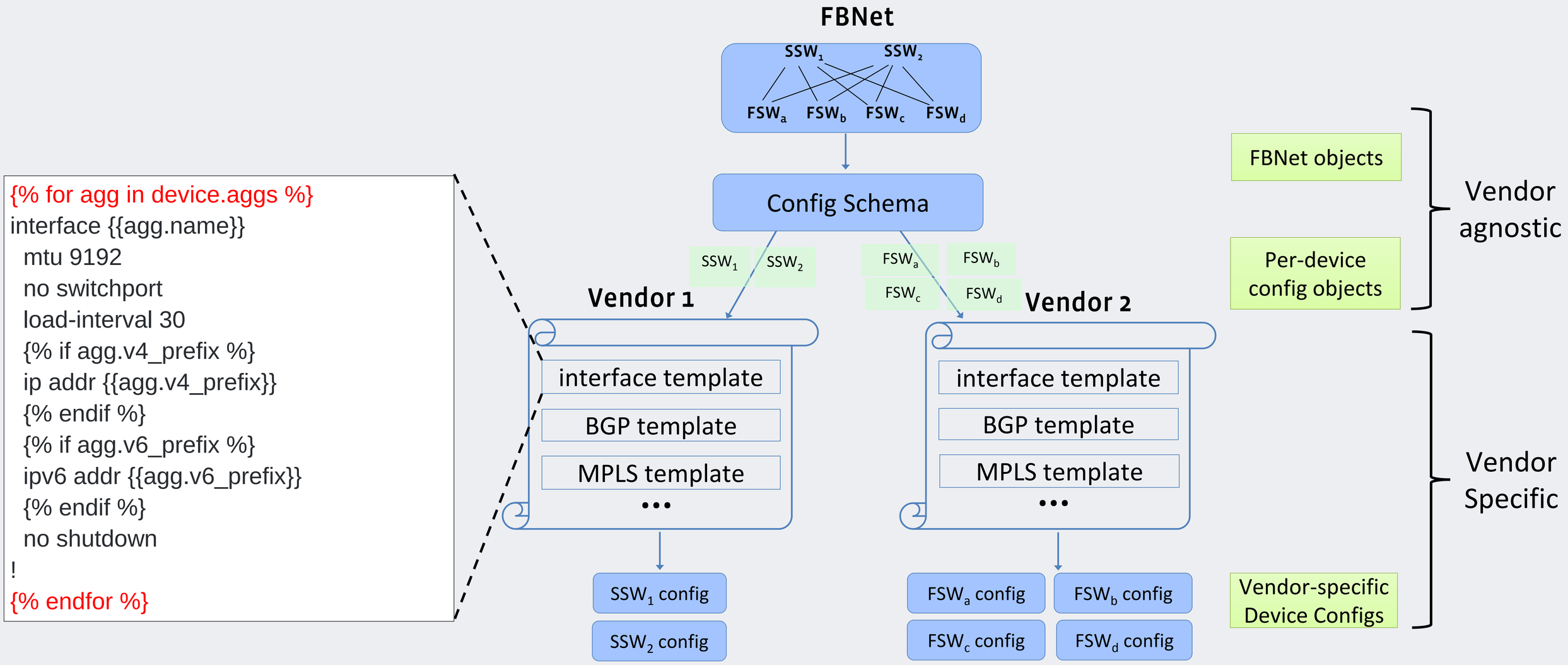
Config Generation

FBNet objects → Device configs



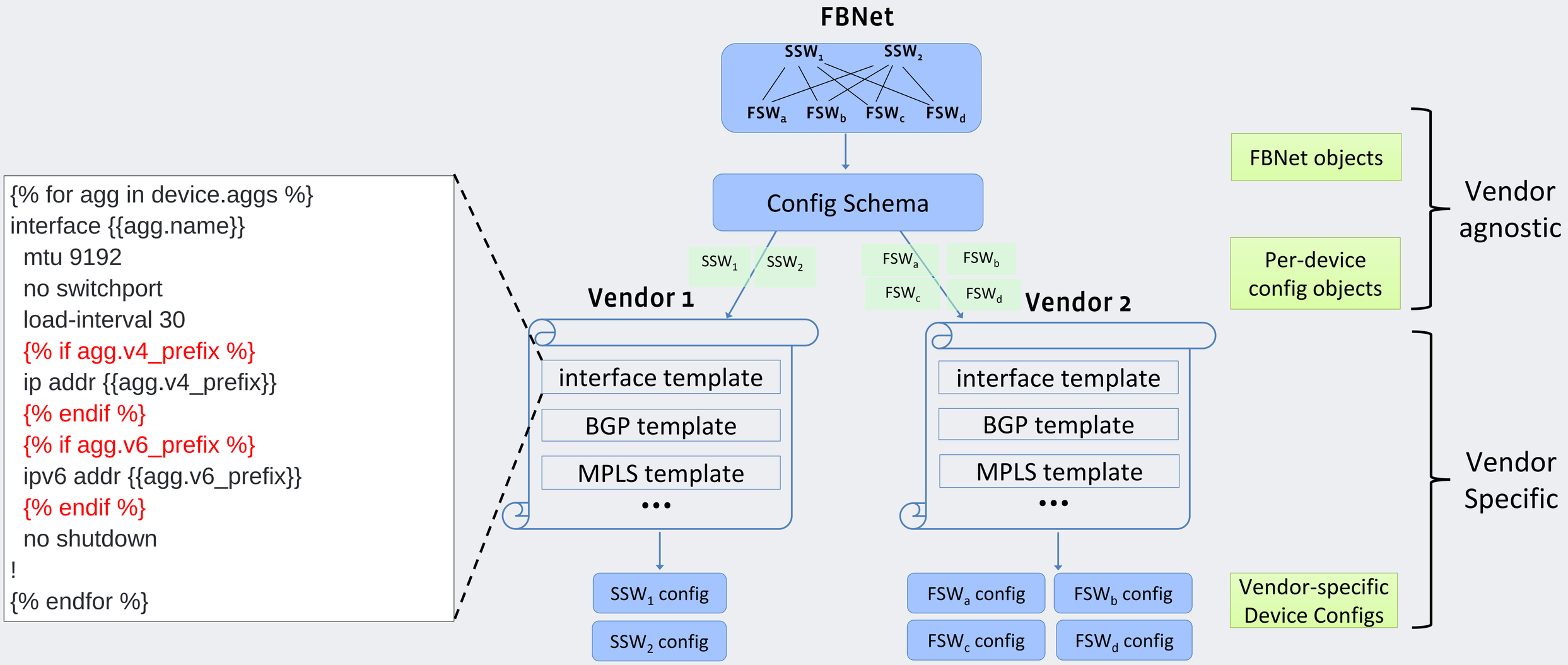
Config Generation

FBNet objects → Device configs



Config Generation

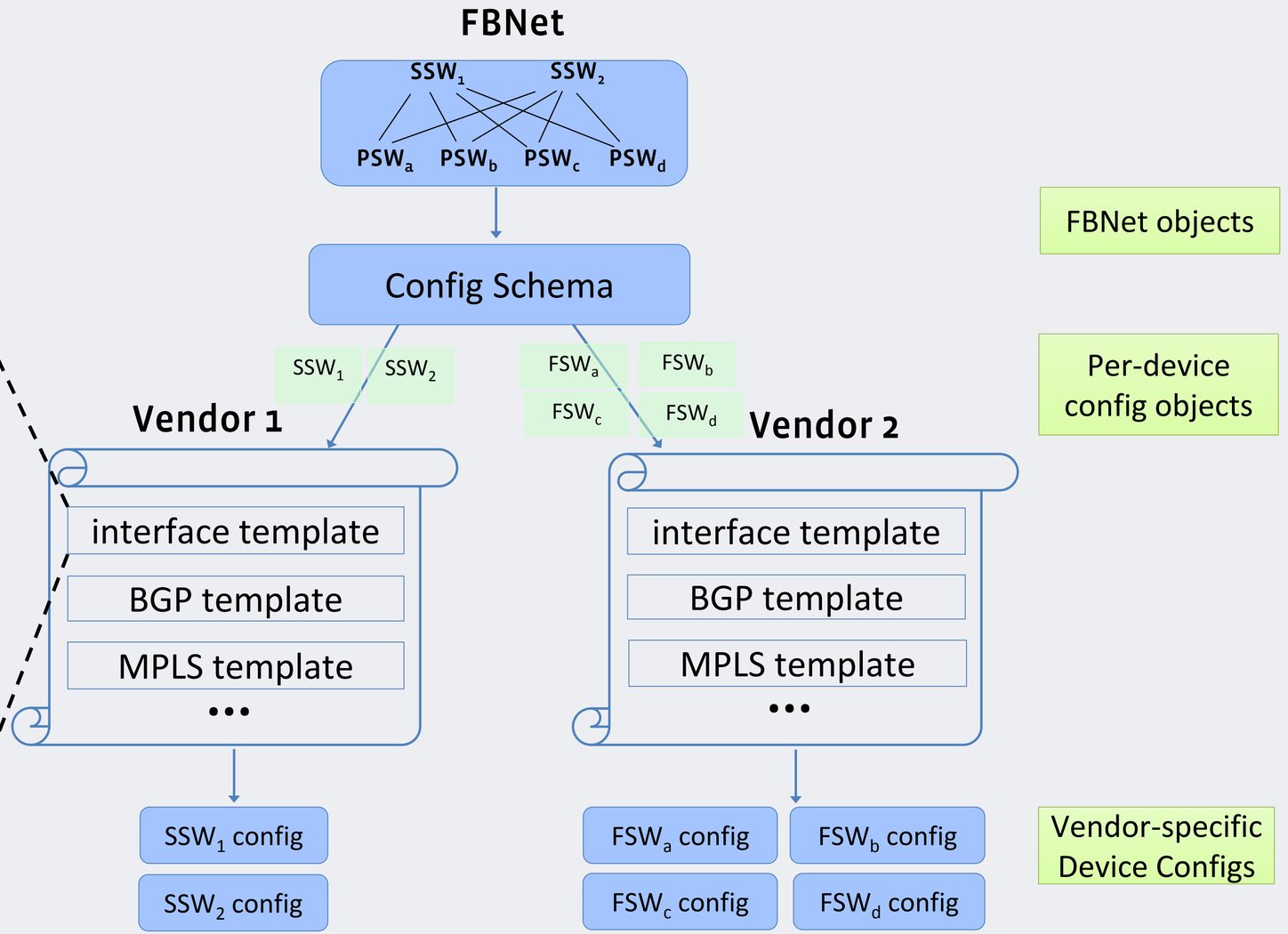
FBNet objects → Device configs



Config Generation

FBNet objects → Device configs

```
{% for agg in device.aggs %}
interface {{agg.name}}
  mtu 9192
  no switchport
  load-interval 30
  {% if agg.v4_prefix %}
  ip addr {{agg.v4_prefix}}
  {% endif %}
  {% if agg.v6_prefix %}
  ipv6 addr {{agg.v6_prefix}}
  {% endif %}
  no shutdown
!
{% endfor %}
```



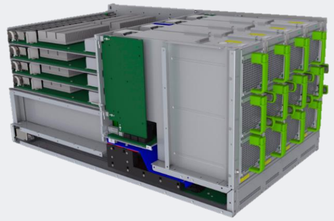
Agenda

1. Overview of Facebook Network
2. Robotron: Our Top-Down Network Management
3. Applying to FBOSS (Wedges + Backpack)
4. Takeaways

Is FBOSS managed differently ?

- Yes, at lower layer
  - Similar to support a new vendor/hardware
  - e.g., how to control, how to monitor, etc.
- Abstraction
  - Model: identify vendor agnostic concepts
  - Implement vendor specific APIs

How do we specialize for FBOSS?



New Model

New template

Configerator (fb infra)

New collector

Network Design

Config Generation

Deployment

Monitoring

FBNet

DB

FBOSS Modeling example

Support Backpack

NetworkSwitch

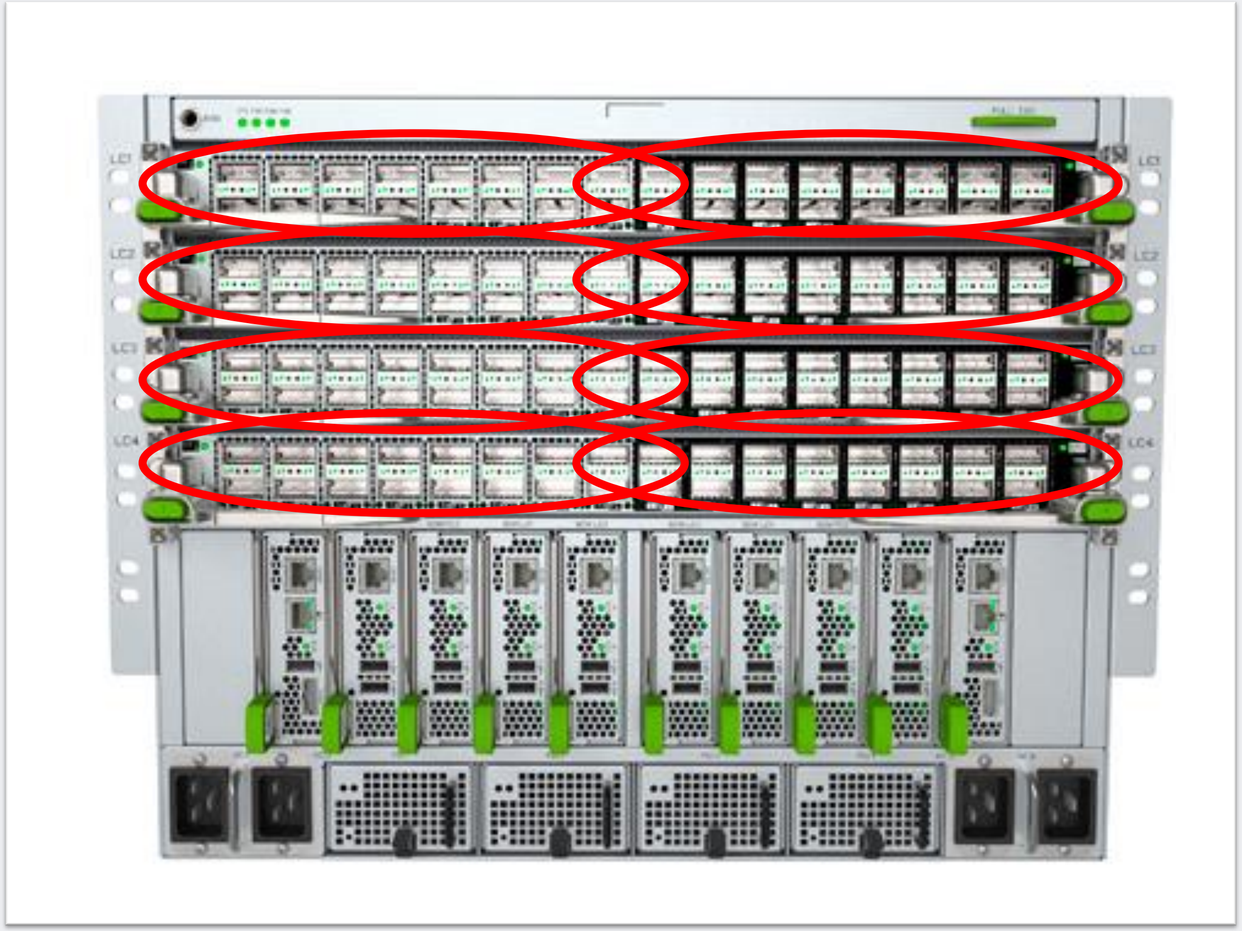
- ipv4
- ipv6
- mgmt_ipv4
- mgmt_ipv6
- chassis_model
- etc.

FBOSS Modeling example

Support Backpack

NetworkSwitch

- ipv4
- ipv6
- mgmt_ipv4
- mgmt_ipv6
- chassis_model
- etc.

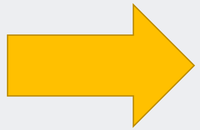


FBOSS Modeling example

Support Backpack

NetworkSwitch

- ipv4
- ipv6
- mgmt_ipv4
- mgmt_ipv6
- chassis_model
- etc.

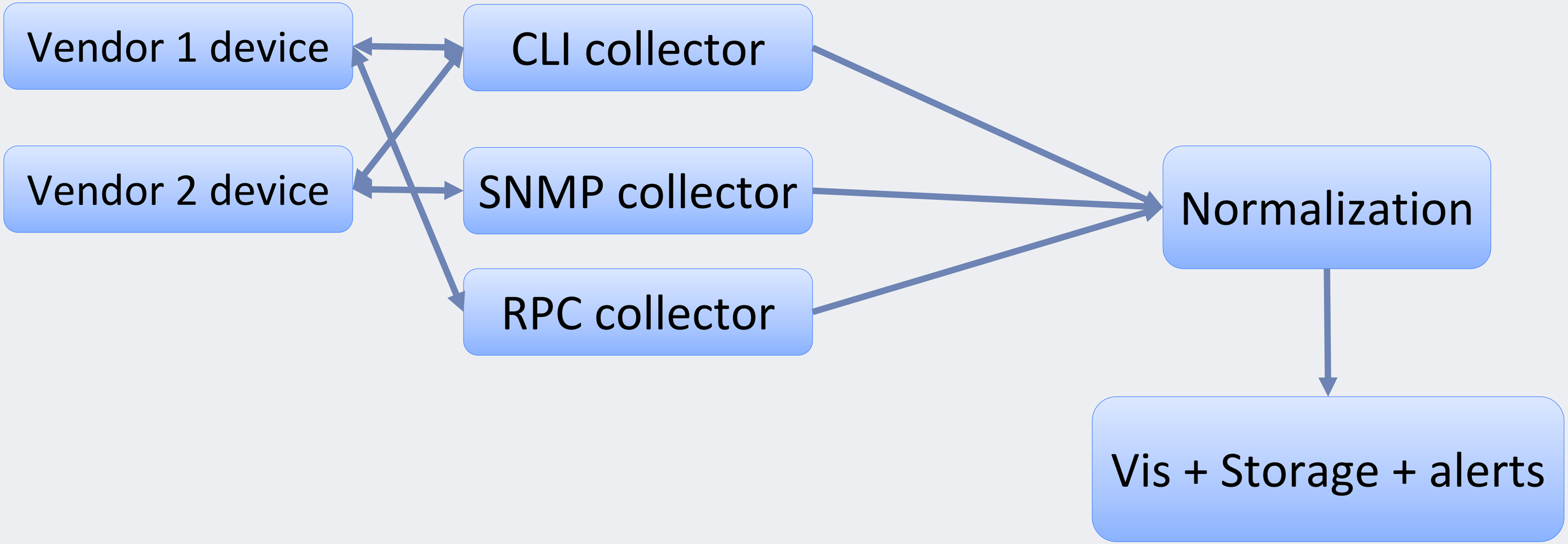


NetworkSwitch

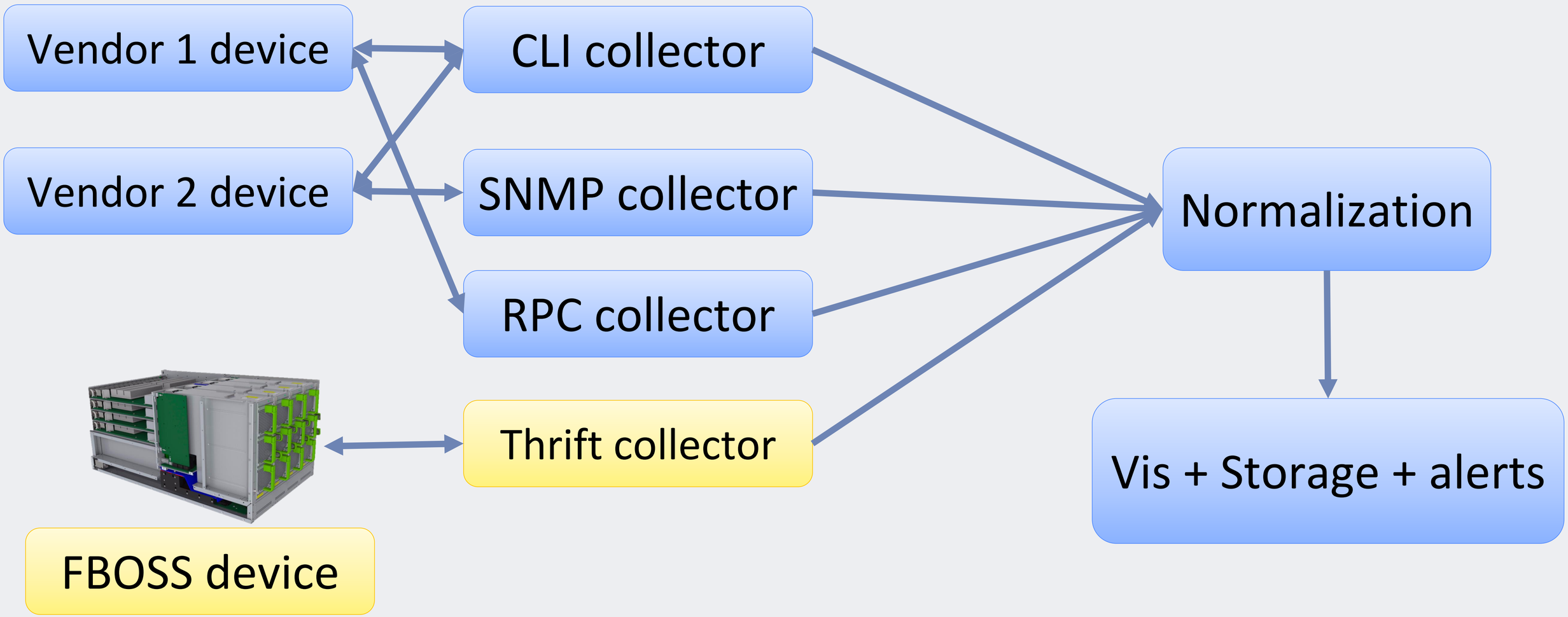
NetworkSubSwitch

- ipv4
- ipv6
- mgmt_ipv4
- mgmt_ipv6
- chassis_model
- etc.

FBOSS Monitoring example



FBOSS Monitoring example



Agenda

1. Overview of Facebook Network
2. Robotron: Our Top-Down Network Management
3. Applying to FBOSS (Wedges + Backpack)
4. Takeaways

Experience 1: Network engineers need time to adapt

- Fboss initially only communicate thru thrift
- Eng ctivity dec
- Dev

- Lesson: CLI is important
- Ongoing challenge: CLIs can be brittle

Experience 2: Coupling changes is key

1. An e
   to g
2. The
3. The

t forgot

- Lesson:
  - Network design, config generation and deployment should be tightly coupled
  - Disaggregated switches make thing possible and better
- Ongoing challenge
  - Atomicity
  - Conflict resolution

Experience 3: Emergency Fallba

- Engin e
  device
    - SSH
    - Mak
    - Log
- Not as
- Needed upon emergencies
- Alerts and active audits with config monitoring to detect

- Lesson: Bypassing mechanism is needed, but…
- Ongoing challenges:
  - Quickly/reliably detect and alert emergency changes
  - Reduce instances over time
  - Safely revert such activities

Summary

- Share experience on how to manage a hybrid network at large scale
  - FBOSS is different, so do other new vendors
  - Abstraction + normalization

- Open challenges:
  - Network engineers need time to adapt
  - Atomicity and conflict resolution across management tasks
  - Emergency manual fallback mechanisms is still needed

Questions?


OPEN
Compute Project