>>> network .toCode()

Hands-on with Closed-Loop Network Automation

ESNOG30

Christian Adell Querol

26-10-2023

>>> network.toCode()

~~Hands-on with Closed-Loop Network Automation~~

ESNOG30

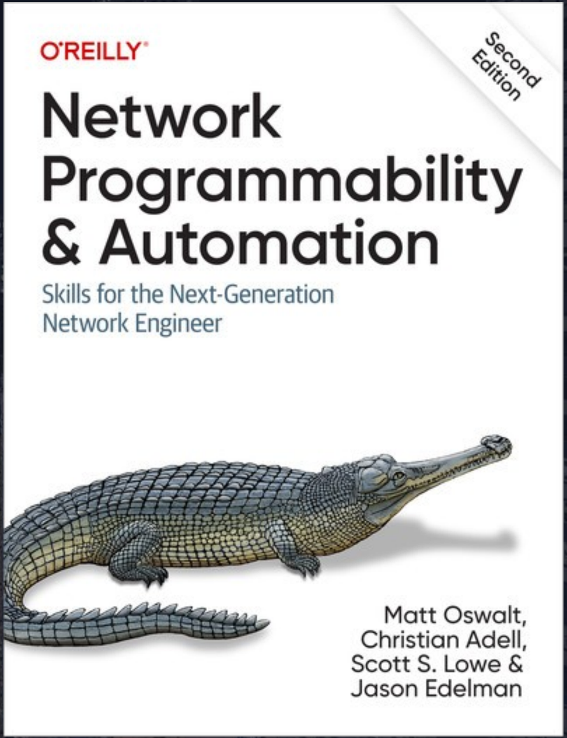
Christian Adell Querol

26-10-2023

>>> network .toCode()

Skills for the Next-Generation Network Engineer

ESNOG30



O'REILLY

Network Programmability & Automation

Skills for the Next-Generation Network Engineer

Second Edition

Matt Oswalt, Christian Adell, Scott S. Lowe & Jason Edelman

Christian Adell Querol

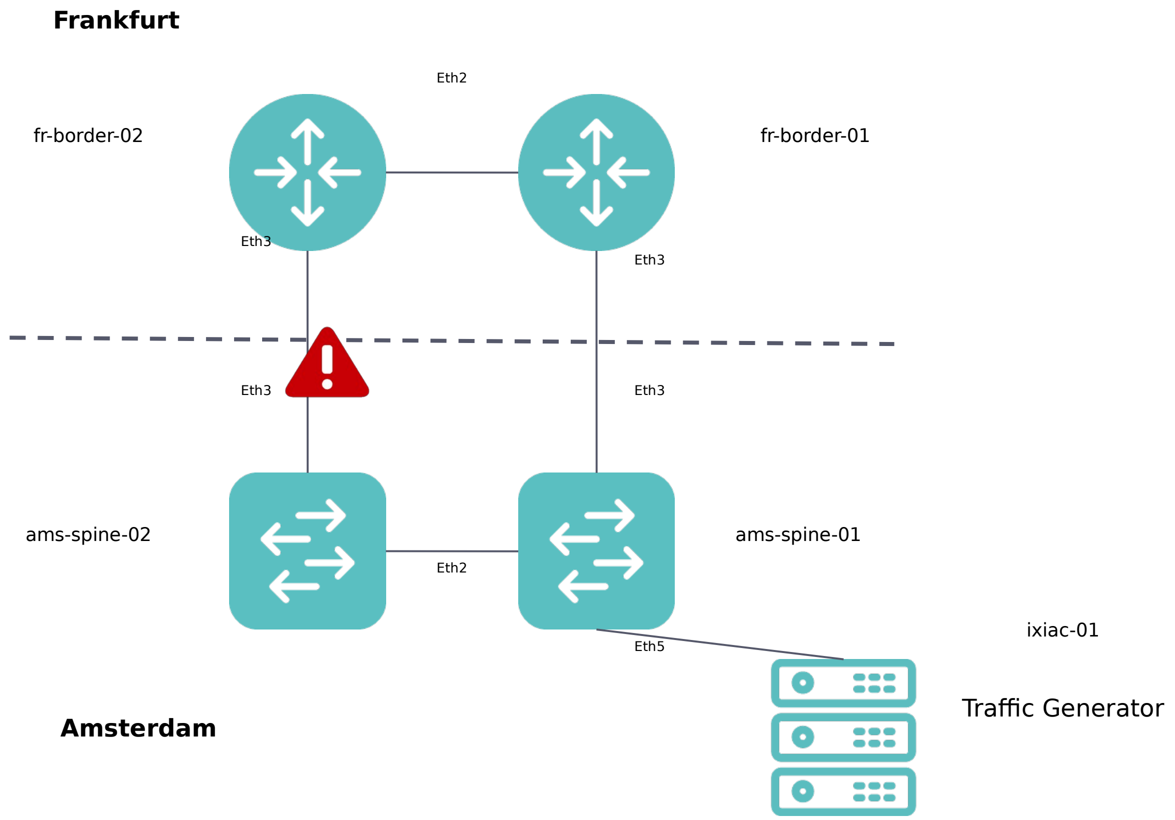
26-10-2023

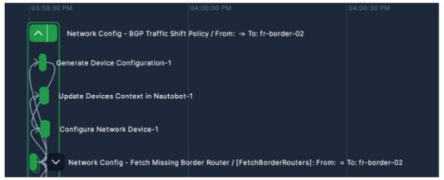
Yo he venido a hablar de mi ~~libro~~

Paco Umbral

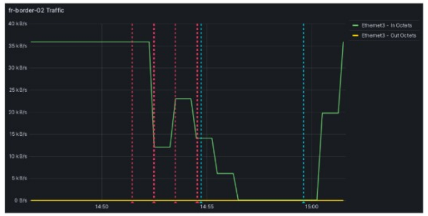
Automated Incident Response

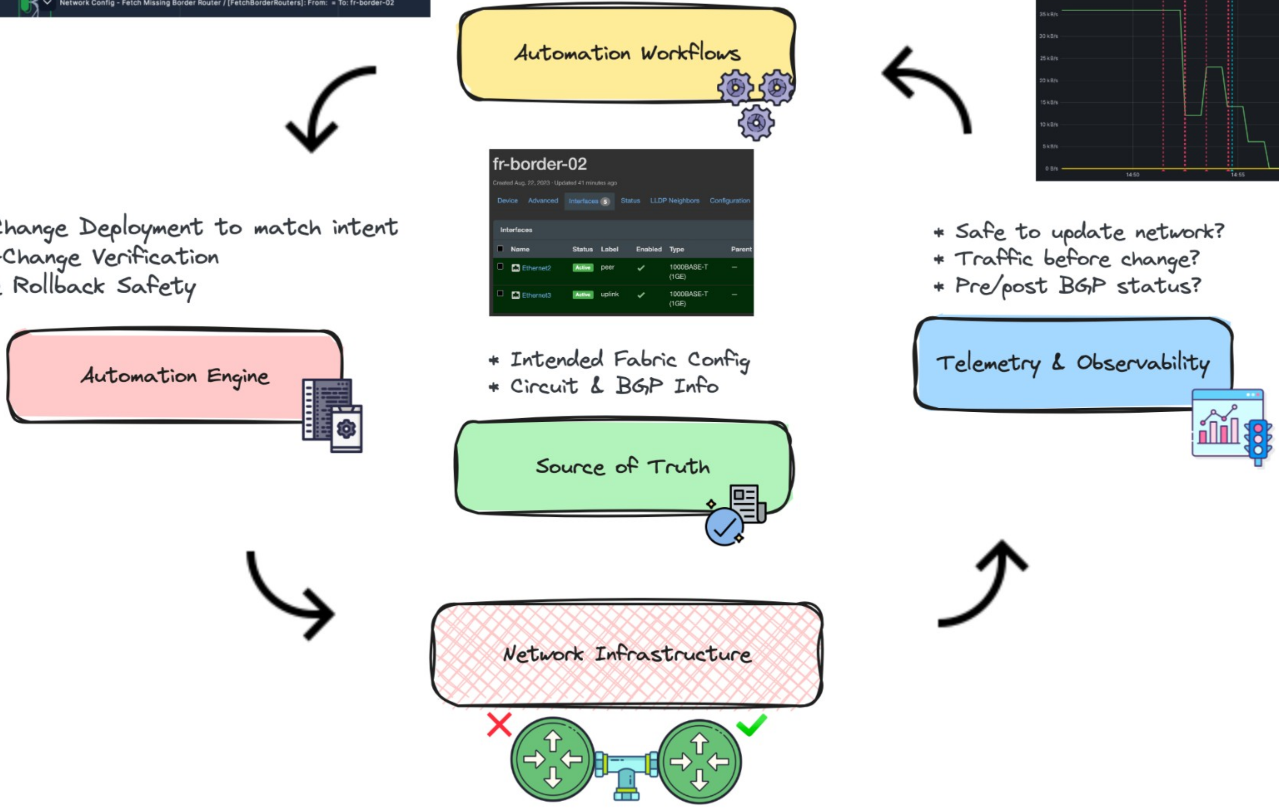
The Problem: Interface Flapping

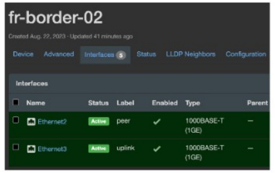



```
03:59:30 PM                    04:00:00 PM                    04:00:30 PM
Network Config - BGP Traffic Shift Policy / From:  -> To: fr-border-02
Generate Device Configuration-1
Update Devices Context in Nautobot-1
Configure Network Device-1
Network Config - Fetch Missing Border Router / [FetchBorderRouters]: From:  = To: fr-border-02
```

* BGP Shift Automation
* Automated RCA Reports
* Assurance Test Automation







fr-border-02

Created Aug. 22, 2023 · Updated 41 minutes ago

Device | Advanced | Interfaces 5 | Status | LLDP Neighbors | Configuration

Interfaces

	Name	Status	Label	Enabled	Type	Parent
☐	Ethernet2	Active	peer	✓	1000BASE-T (1GE)	—
☐	Ethernet3	Active	uplink	✓	1000BASE-T (1GE)	—

* Safe Change Deployment to match intent
* State-Change Verification
* Change Rollback Safety

* Safe to update network?
* Traffic before change?
* Pre/post BGP status?

* Intended Fabric Config
* Circuit & BGP Info

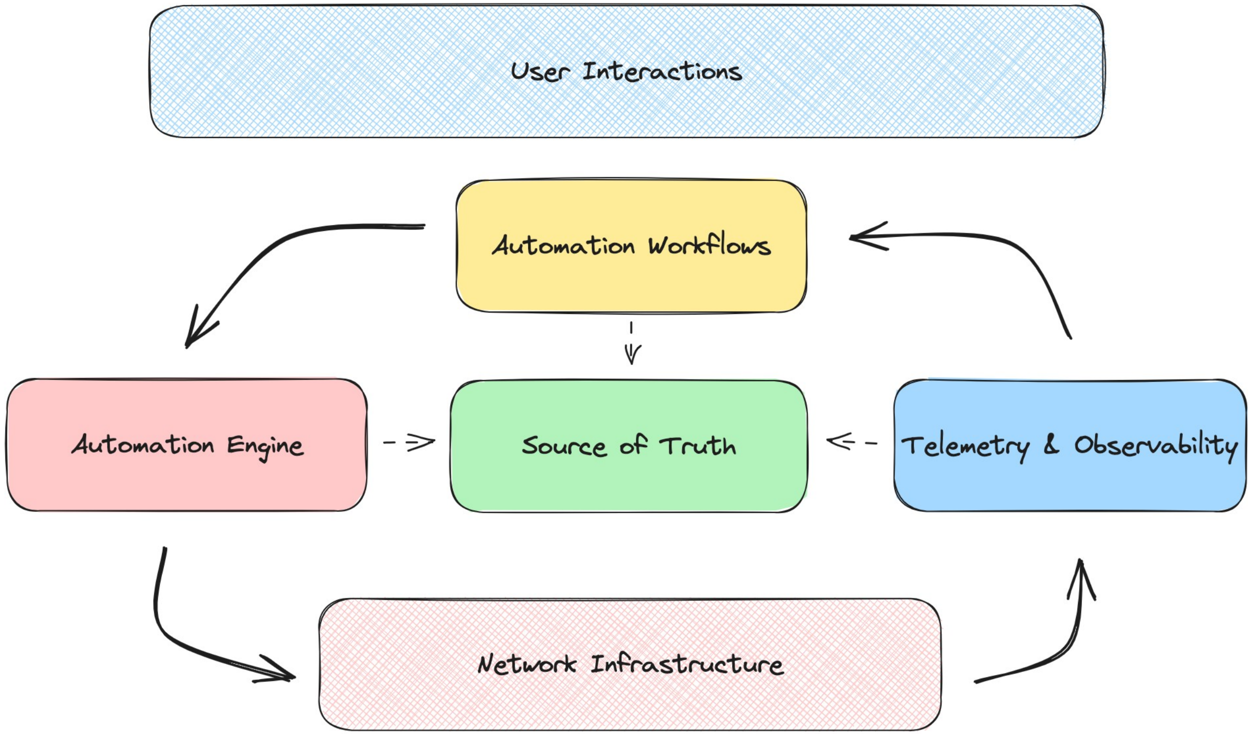
>>> Let’s see it in action!

Skills for building this solution

I love it when a plan comes together

Colonel John "Hannibal" Smith

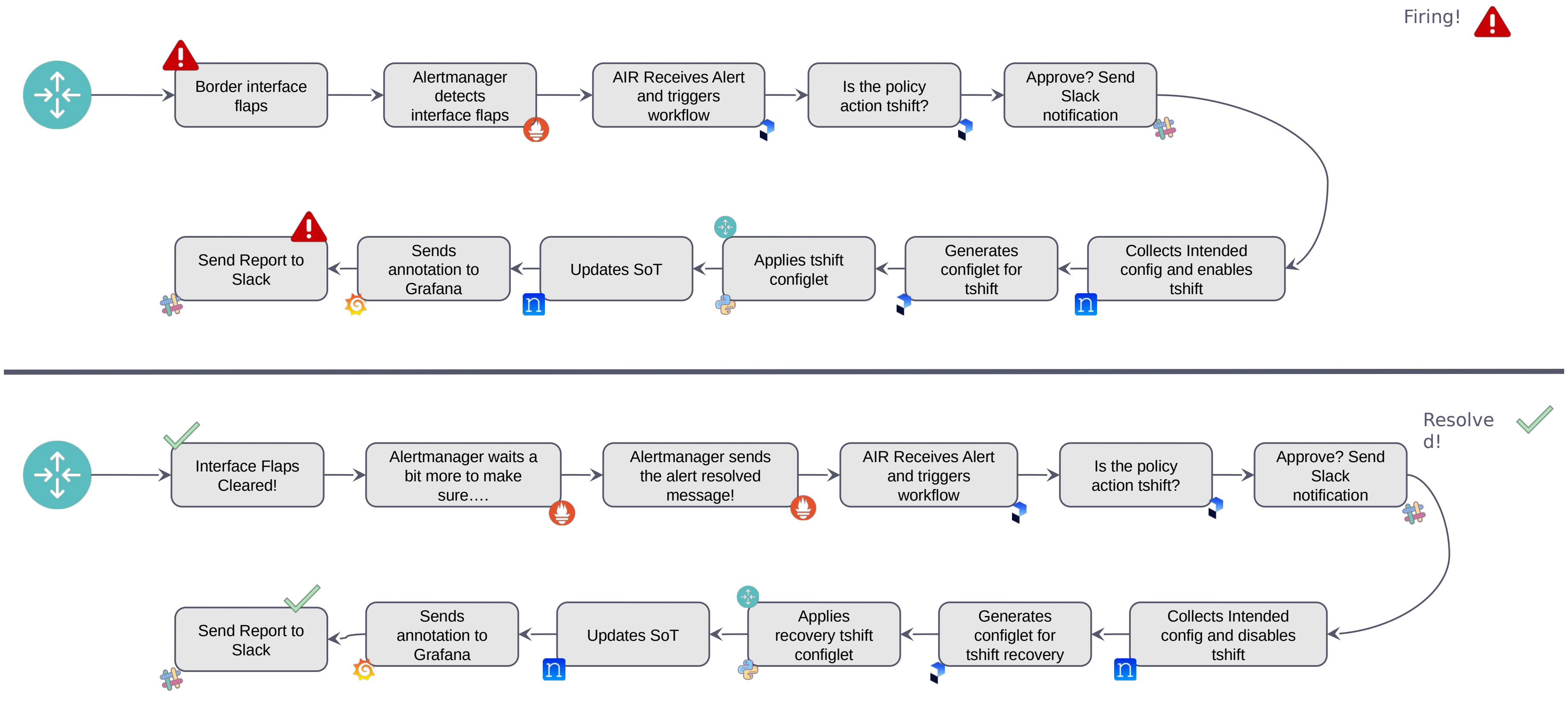
>>> Use a Reference Architecture



A system is not the sum of its parts,
it’s their interactions

Russell L. Ackoff

Determine the workflow execution

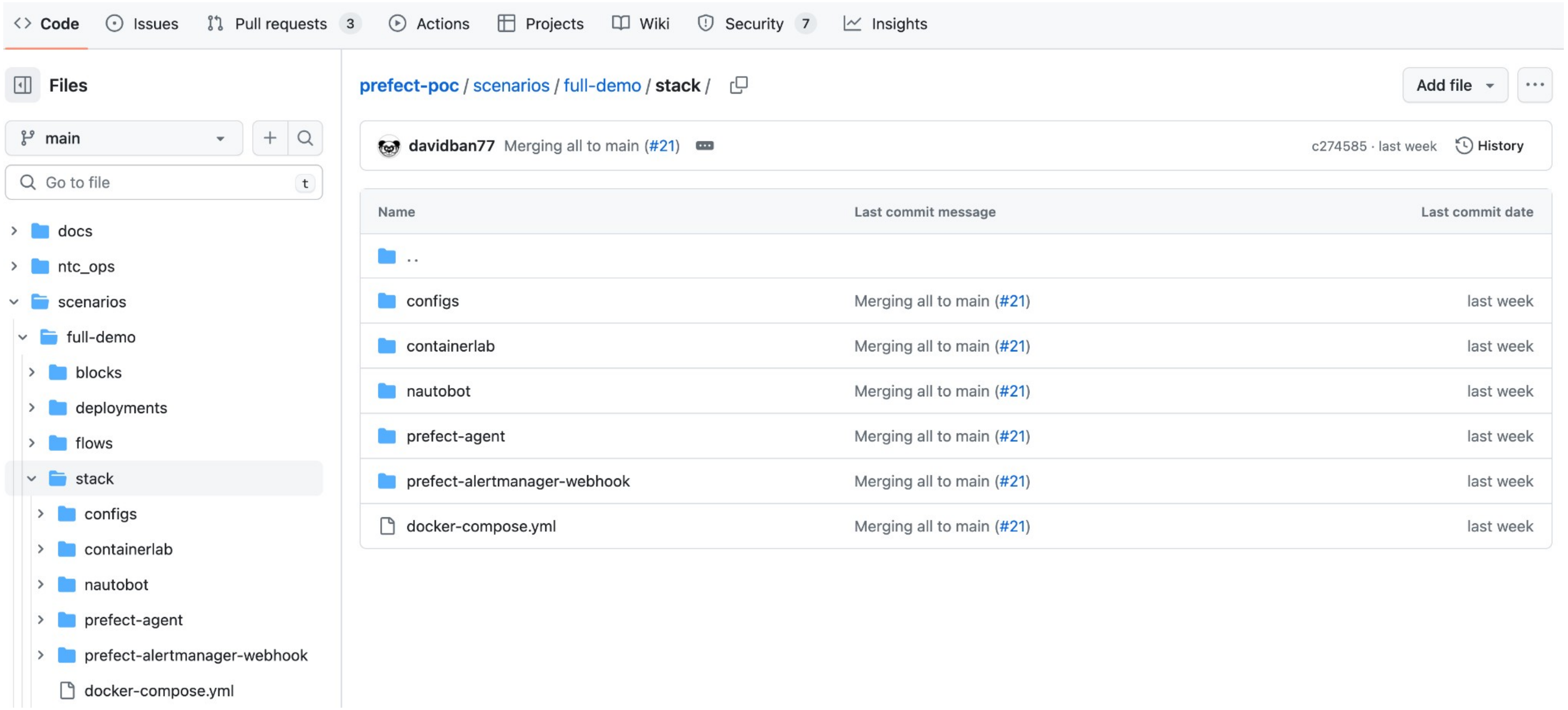

Firing!
Border interface flaps
Alertmanager detects interface flaps
AIR Receives Alert and triggers workflow
Is the policy action tshift?
Approve? Send Slack notification
Collects Intended config and enables tshift
Generates configlet for tshift
Applies tshift configlet
Updates SoT
Sends annotation to Grafana
Send Report to Slack
Resolve d!
Interface Flaps Cleared!
Alertmanager waits a bit more to make sure….
Alertmanager sends the alert resolved message!
AIR Receives Alert and triggers workflow
Is the policy action tshift?
Approve? Send Slack notification
Collects Intended config and disables tshift
Generates configlet for tshift recovery
Applies recovery tshift configlet
Updates SoT
Sends annotation to Grafana
Send Report to Slack

***If you want to go fast, go alone.
If you want to go far, go together.***

**African
Proverb**

***… and if you want to go reliably,
go with continuous integration***

>>> Get to work with Git and CI/CD



prefect-poc / scenarios / full-demo / stack /

davidban77 Merging all to main (#21) c274585 · last week

- docs
- ntc_ops
- scenarios
  - full-demo
    - blocks
    - deployments
    - flows
    - stack
      - configs
      - containerlab
      - nautobot
      - prefect-agent
      - prefect-alertmanager-webhook
      - docker-compose.yml

Name	Last commit message	Last commit date
..		
configs	Merging all to main (#21)	last week
containerlab	Merging all to main (#21)	last week
nautobot	Merging all to main (#21)	last week
prefect-agent	Merging all to main (#21)	last week
prefect-alertmanager-webhook	Merging all to main (#21)	last week
docker-compose.yml	Merging all to main (#21)	last week

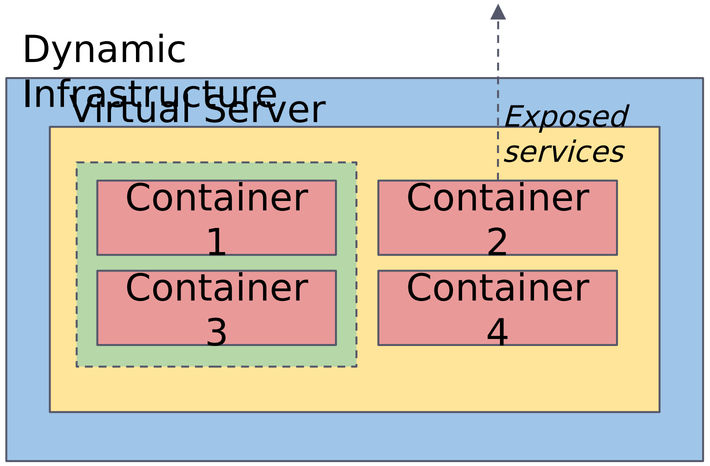
I choose a lazy person to do a hard job. Because a lazy person will find an easy way to do it.

Bill Gates

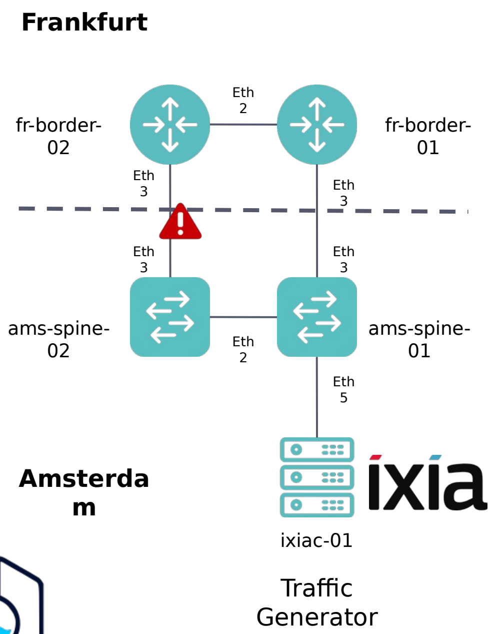
Embrace Dynamic Infrastructure

```
resource "digitalocean_droplet" "ntc-ops_vm" {
  image = "ubuntu-22-04-x64"
  name = format("%s-%s", "ntc-ops", var.user)
  region = var.vm_region
  size = var.vm_size
  ssh_keys = [
    data.digitalocean_ssh_key.terraform.id
  ]
  tags = [
    "ntc-ops-vm"
  ]

connection {
  host = self.ipv4_address
  user = "root"
  type = "ssh"
  private_key = file(var.pvt_key)
  timeout = "2m"
}
provisioner "file" {
  source = var.pub_ssh_key
  destination = "/tmp/temp.pub"
}
```

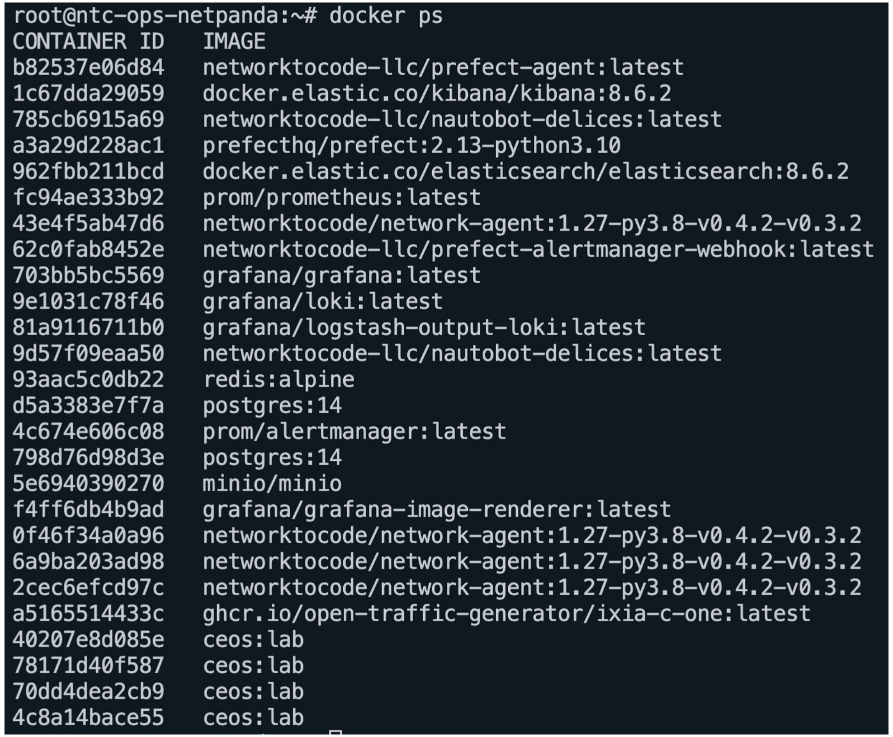


>>> Spin up a network dev environment



```yaml
---
name: full-demo
prefix: "" # Empty string to not add a prefix to the containers
mgmt:
  network: full-demo
  ipv4-subnet: 172.24.77.0/24
topology:
  kinds:
    ceos:
      image: ceos:lab
  nodes:
    fr-border-01:
      kind: ceos
      mgmt-ipv4: 172.24.77.11
      startup-config: startups/fr-border-01.conf
      publish:
        - tcp/50051
        - tcp/80
        - tcp/443
        - udp/161
```

>>> Containerization makes your life easier

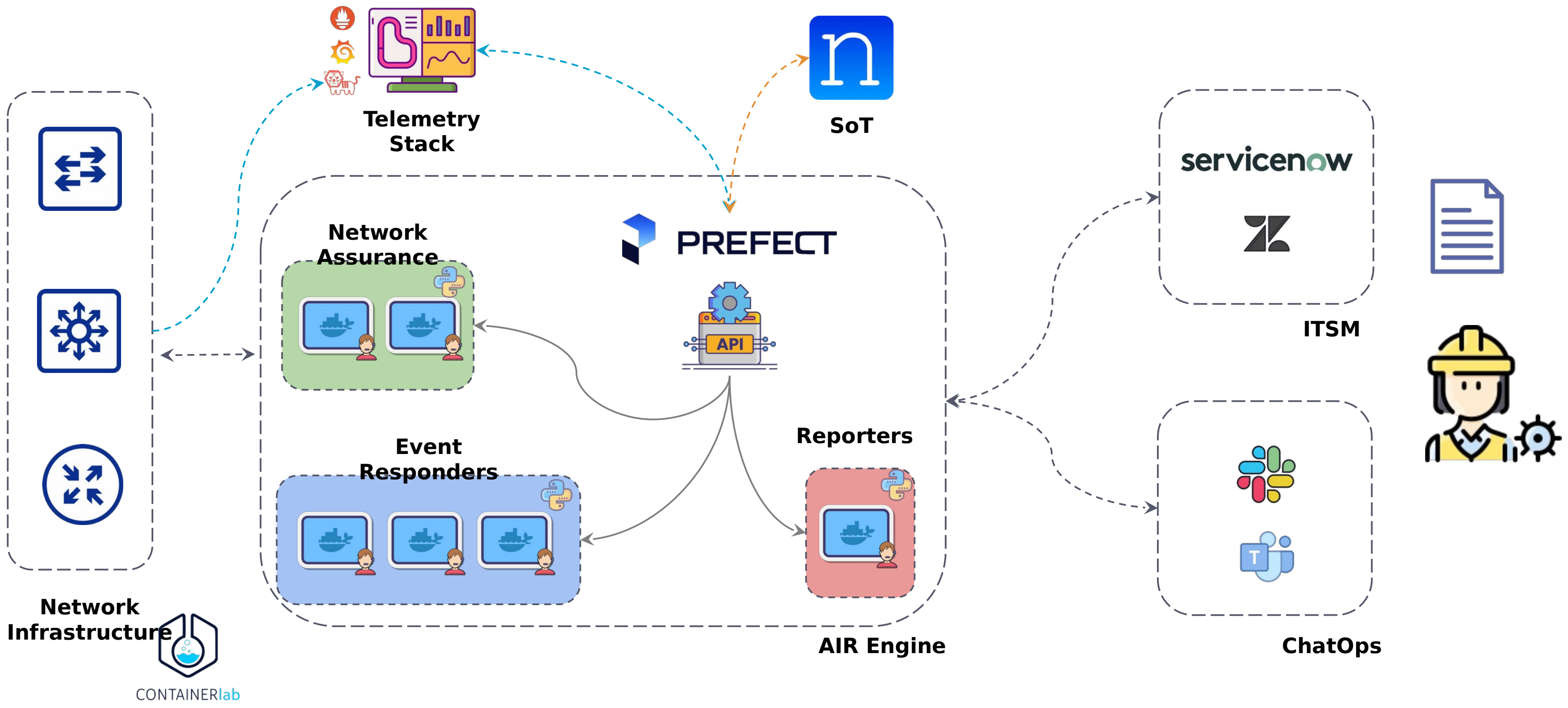


```
root@ntc-ops-netpanda:~# docker ps
CONTAINER ID   IMAGE
b82537e06d84   networktocode-llc/prefect-agent:latest
1c67dda29059   docker.elastic.co/kibana/kibana:8.6.2
785cb6915a69   networktocode-llc/nautobot-delices:latest
a3a29d228ac1   prefecthq/prefect:2.13-python3.10
962fbb211bcd   docker.elastic.co/elasticsearch/elasticsearch:8.6.2
fc94ae333b92   prom/prometheus:latest
43e4f5ab47d6   networktocode/network-agent:1.27-py3.8-v0.4.2-v0.3.2
62c0fab8452e   networktocode-llc/prefect-alertmanager-webhook:latest
703bb5bc5569   grafana/grafana:latest
9e1031c78f46   grafana/loki:latest
81a9116711b0   grafana/logstash-output-loki:latest
9d57f09eaa50   networktocode-llc/nautobot-delices:latest
93aac5c0db22   redis:alpine
d5a3383e7f7a   postgres:14
4c674e606c08   prom/alertmanager:latest
798d76d98d3e   postgres:14
5e6940390270   minio/minio
f4ff6db4b9ad   grafana/grafana-image-renderer:latest
0f46f34a0a96   networktocode/network-agent:1.27-py3.8-v0.4.2-v0.3.2
6a9ba203ad98   networktocode/network-agent:1.27-py3.8-v0.4.2-v0.3.2
2cec6efcd97c   networktocode/network-agent:1.27-py3.8-v0.4.2-v0.3.2
a5165514433c   ghcr.io/open-traffic-generator/ixia-c-one:latest
40207e8d085e   ceos:lab
78171d40f587   ceos:lab
70dd4dea2cb9   ceos:lab
4c8a14bace55   ceos:lab
```

We are stubborn on vision.
We are flexible on details.

Jeff Bezos

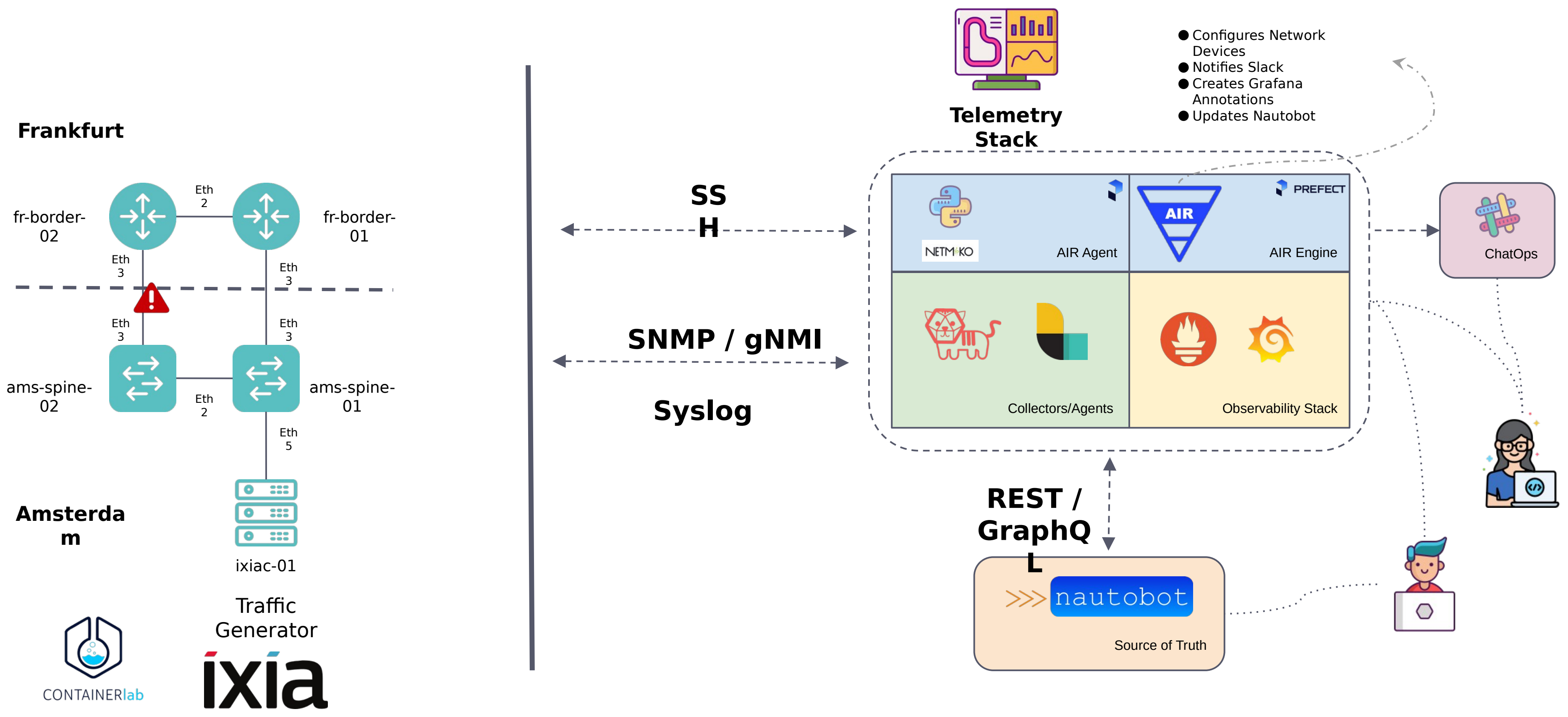
Tooling Agnostic



Give me a place to stand and I will move the earth.

Archimedes

>>> It's all about interactions via APIs



Programming isn't about what you know;
it's about what you can figure out.

Chris Pine

>>> Configure Devices with Python and Jinja2

```python
@flow(
  name="Network Config - Configure Device",
  description="Flow to configure a device.",
  flow_run_name="Device: {device_name}",
)
def configure_device(device_name: str, config_template: str | None = None) -> bool:
"""Configure device."""
  logger = get_run_logger()
  logger.info(f"Retrieving device {device_name} data from Nautobot...")
  device = get_device(device_name)
  # Get interfaces information
  logger.info(f"Retrieving device {device_name} interfaces from Nautobot...")
  device_interfaces = get_device_interfaces.submit(device_name=device_name)
  process_device_interfaces.submit(device_interfaces)
  logger.info(f"Device {device_name} interfaces retrieved from Nautobot!")
  # Add interfaces to config_vars
  config_vars = generate_config_vars.submit(device=device,
device_interfaces=device_interfaces)
  # Get Template data
  template_file = Path(__file__).parent / "tasks" / "network" / "templates" /
f"{device.platform}.j2"
  # Generate the configuration
  config = net_device.generate_config.submit(
    device=device.name,
    device_type=device.platform,
    template=template_file.read_text(),
    config_vars=config_vars,
    wait_for=[device, device_interfaces],
  )
```

```jinja
{% if config_data.bgp is defined %}
router bgp {{ config_data.bgp.asn }}
  bgp log-neighbor-changes
{%   if config_data.bgp.router_id is defined %}
  router-id {{ config_data.bgp.router_id }}
{%   endif %}
{%   for redis in config_data.bgp.redistribute | default([]) %}
{%     if redis.route_map is defined %}
  redistribute {{redis.type}} route-map {{ redis.route_map }}
{%     else %}
  redistribute {{redis.type}}
{%     endif %}
{%   endfor %}
{%   for neighbor in config_data.bgp.neighbors | default([]) %}
  neighbor {{ neighbor.ip }} remote-as {{ neighbor.asn }}
{%     if neighbor.description is defined %}
  neighbor {{ neighbor.ip }} description {{ neighbor.description }}
{%     endif %}
{%     if neighbor.route_map is defined %}
  neighbor {{ neighbor.ip }} route-map {{ neighbor.route_map.name }}
{{ neighbor.route_map.direction | default("in") }}
{%     endif %}
{%     if neighbor.tshift is defined %}
{%       if neighbor.tshift.enabled %}
  neighbor {{ neighbor.ip }} route-map {{ neighbor.tshift.route_map }}
{{ neighbor.tshift.direction | default("in") }}
{%       endif %}
{%     endif %}
{%     if neighbor.max_routes is defined %}
  neighbor {{ neighbor.ip }} maximum-routes {{ neighbor.max_routes }}
{%     endif %}
{%   endfor %}
{% endif %}
```

Did we have a demo ongoing?

>>> How to get started?

Why not?

Christian Adell

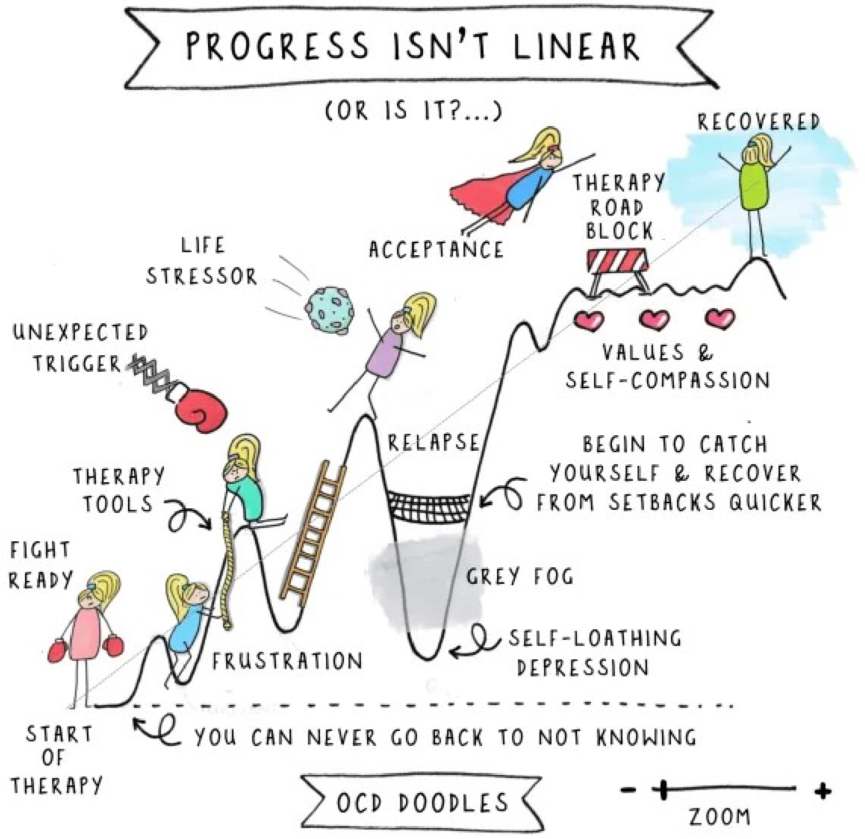
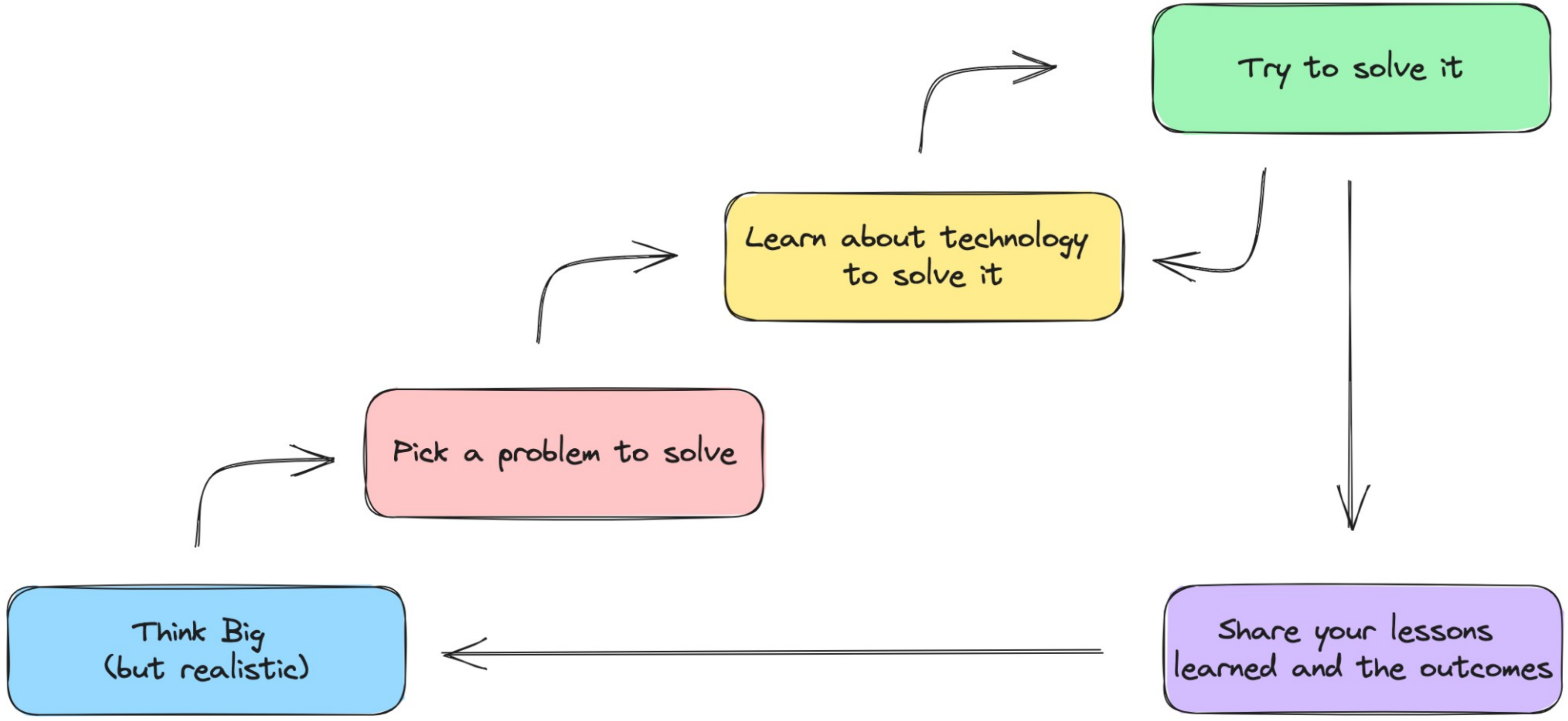


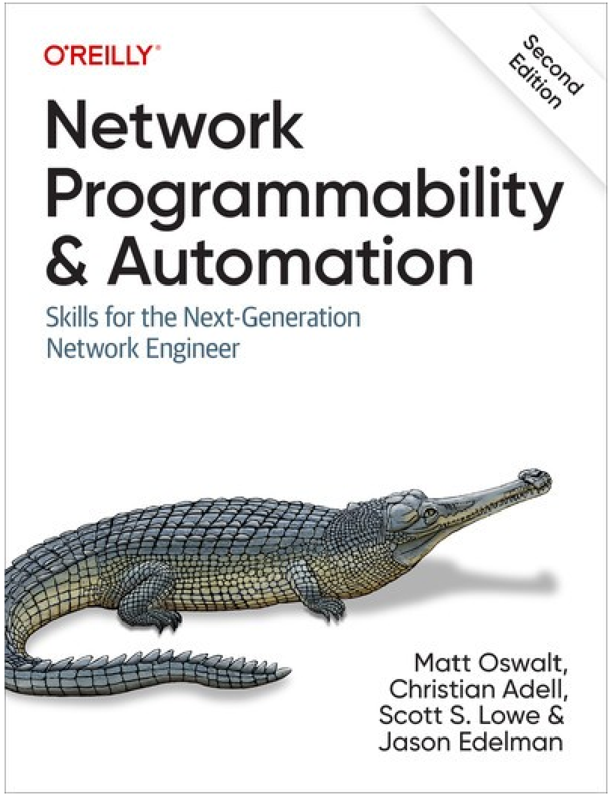
Image credits: https://navigatinguncertaintyblog.wordpress.com/

>>> It’s an iterative process


Try to solve it
Learn about technology to solve it
Pick a problem to solve
Think Big (but realistic)
Share your lessons learned and the outcomes

>>> Book: Network Programmability & Automation, 2n Edition

- Programming skills with **Python and Go**: data types, conditionals, loops, functions, and more
- New **Linux-based networking** technologies and cloud native environments, and how to use them to bootstrap development environments for your network projects
- **Data formats and models**: JSON, XML, YAML, Protobuf, and YANG
- **Jinja** templating for creating network device configurations
- A holistic approach to **architecting network automation services**
- The role of **application programming interfaces (APIs) in network automation**
- **Source control with Git** to manage code changes during the automation process
- **Cloud-native technologies** like Docker and Kubernetes
- How to automate network devices and services using **Ansible**, **Nornir**, and **Terraform**
- Tools and technologies for developing and **continuously integrating network automation**


O'REILLY
Second Edition
Network Programmability & Automation
Skills for the Next-Generation Network Engineer
Matt Oswalt, Christian Adell, Scott S. Lowe & Jason Edelman

Get your free book in the final contest tomorrow!

>>> network .toCode()

Thanks!