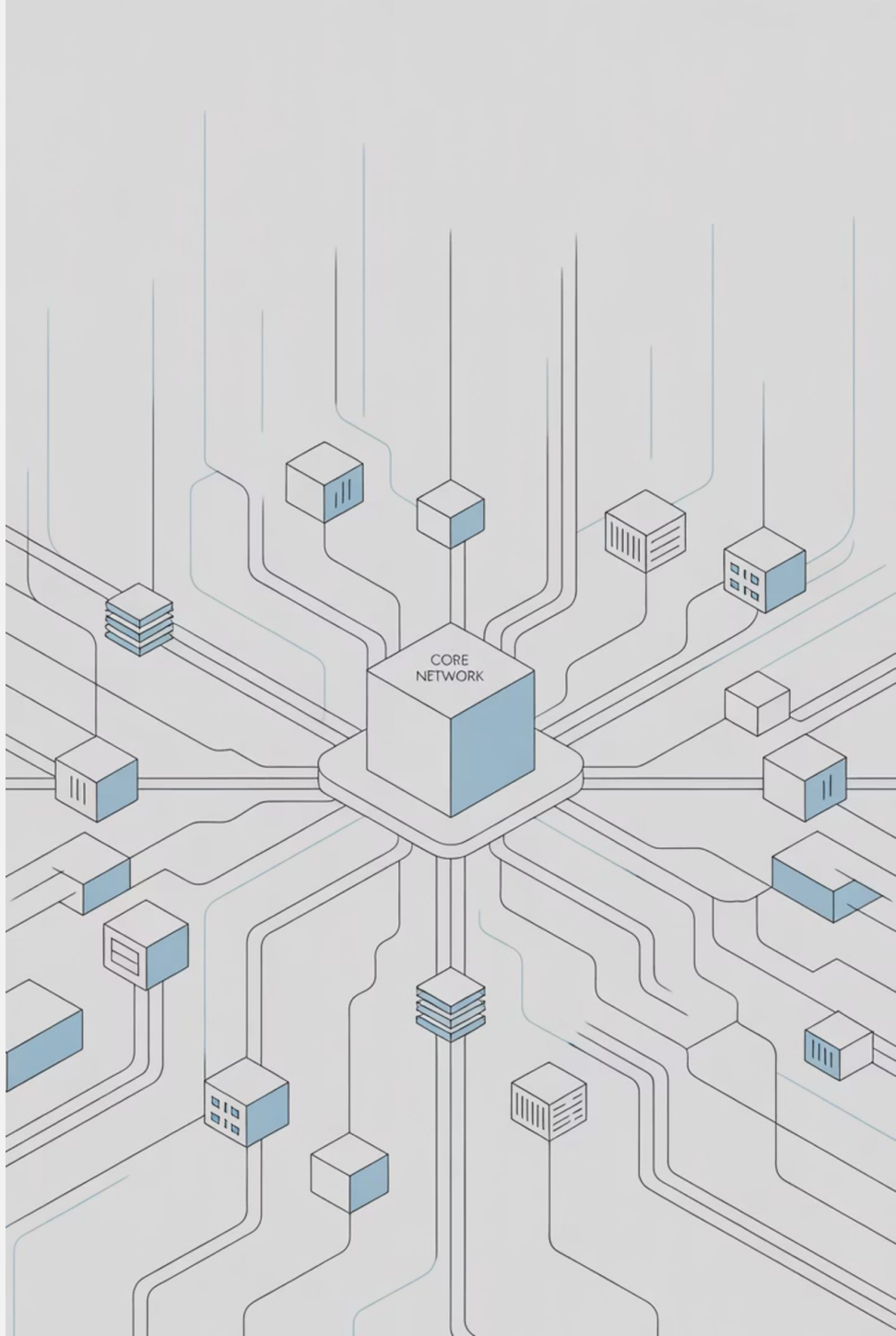
Building a Service Layer in a Vendor Agnostic Way

A practical approach to bridging the gap between business services and network implementation



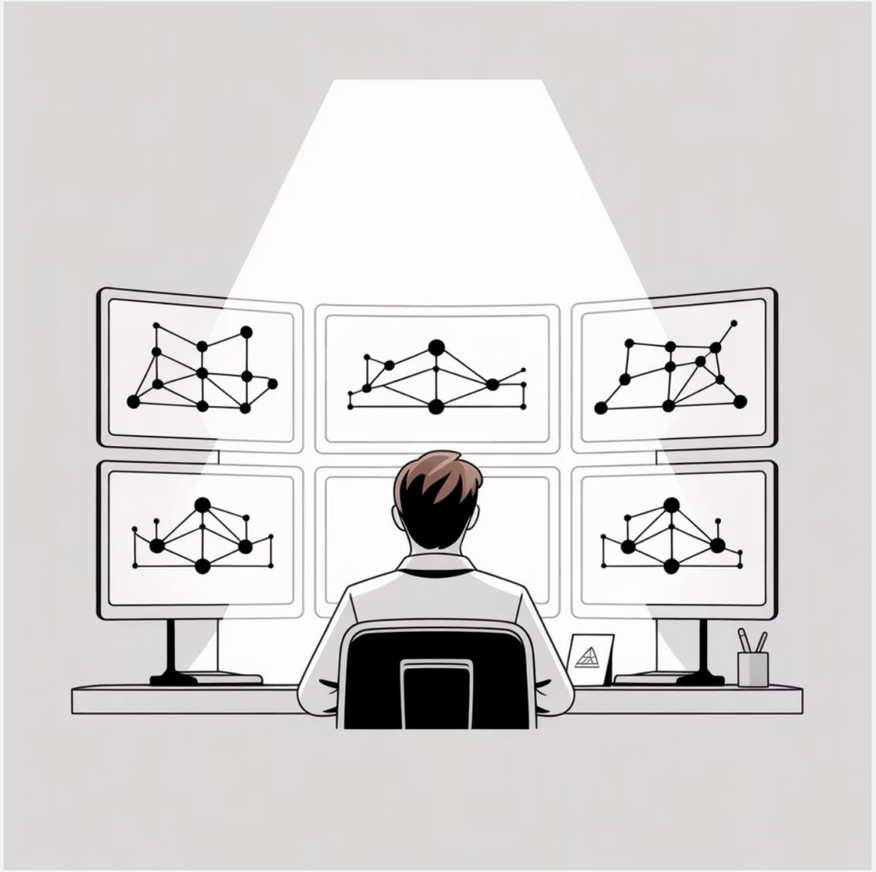
About me - Baptiste Girard

ISP Experience

With years of experience in the ISP domain, I deeply understand operational pain points and the challenges of network automation.

OpsMill & Infrahub

Currently working at OpsMill, the company behind Infrahub, developing solutions for network automation challenges

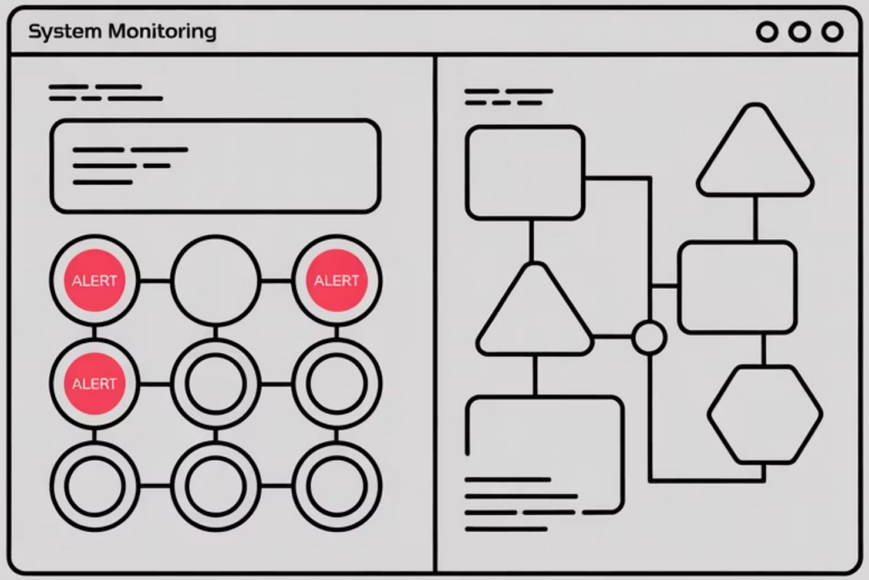


The Monitoring Paradox

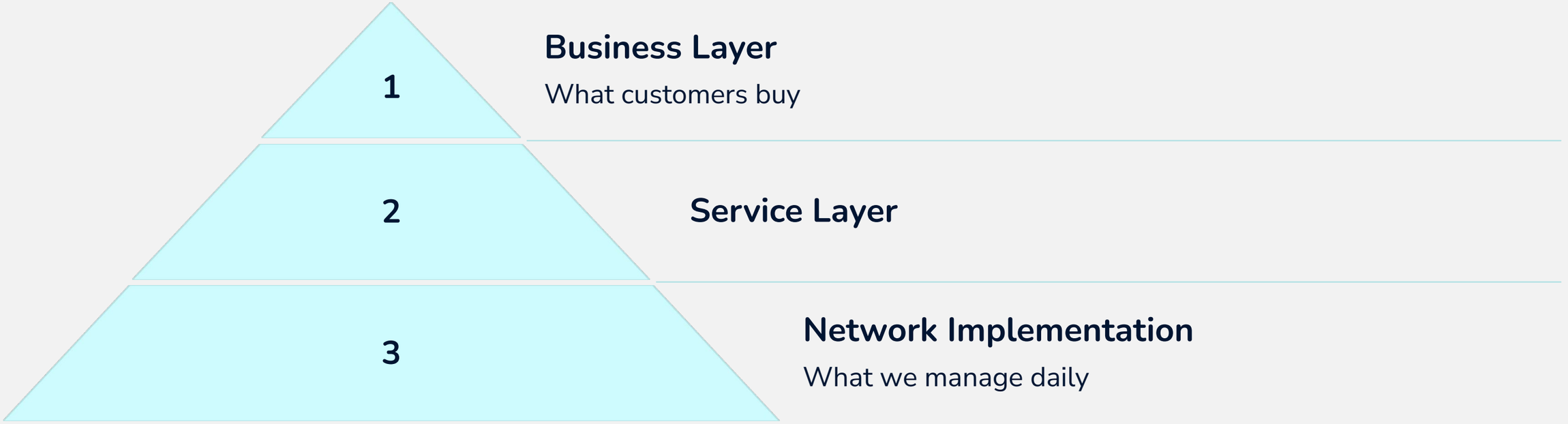
> "I don't need a source of truth. I already have my monitoring."

But monitoring only tells you an interface is **down**... It doesn't tell you if that interface **SHOULD** be down, or what services depend on it.

Monitoring gives you the "**is**", you need the "**should**" to understand business impact.



The Service Layer challenge



Customers buy "10Gbps dedicated internet" or "MPLS VPN" – but we manage VLANs, interfaces, routing protocols, and ACLs

Key benefits: Service lifecycle automation, proactive capacity planning, automated billing ...

Current Options

Rigid CMDBs

Force your unique services into predefined boxes that don't match reality

Homemade Solutions

Expensive to build, harder to maintain

Multiple Systems

ServiceNow + IPAM + Excel + Visio = Data silos and synchronization nightmares

Result: Lot of time fighting tools, hard to reach the full potential

Infrahub

Flexible Platform

Adapts to your organization's unique needs and processes rather than forcing you to change

Version Control & Branching

Apply software development practices to infrastructure management

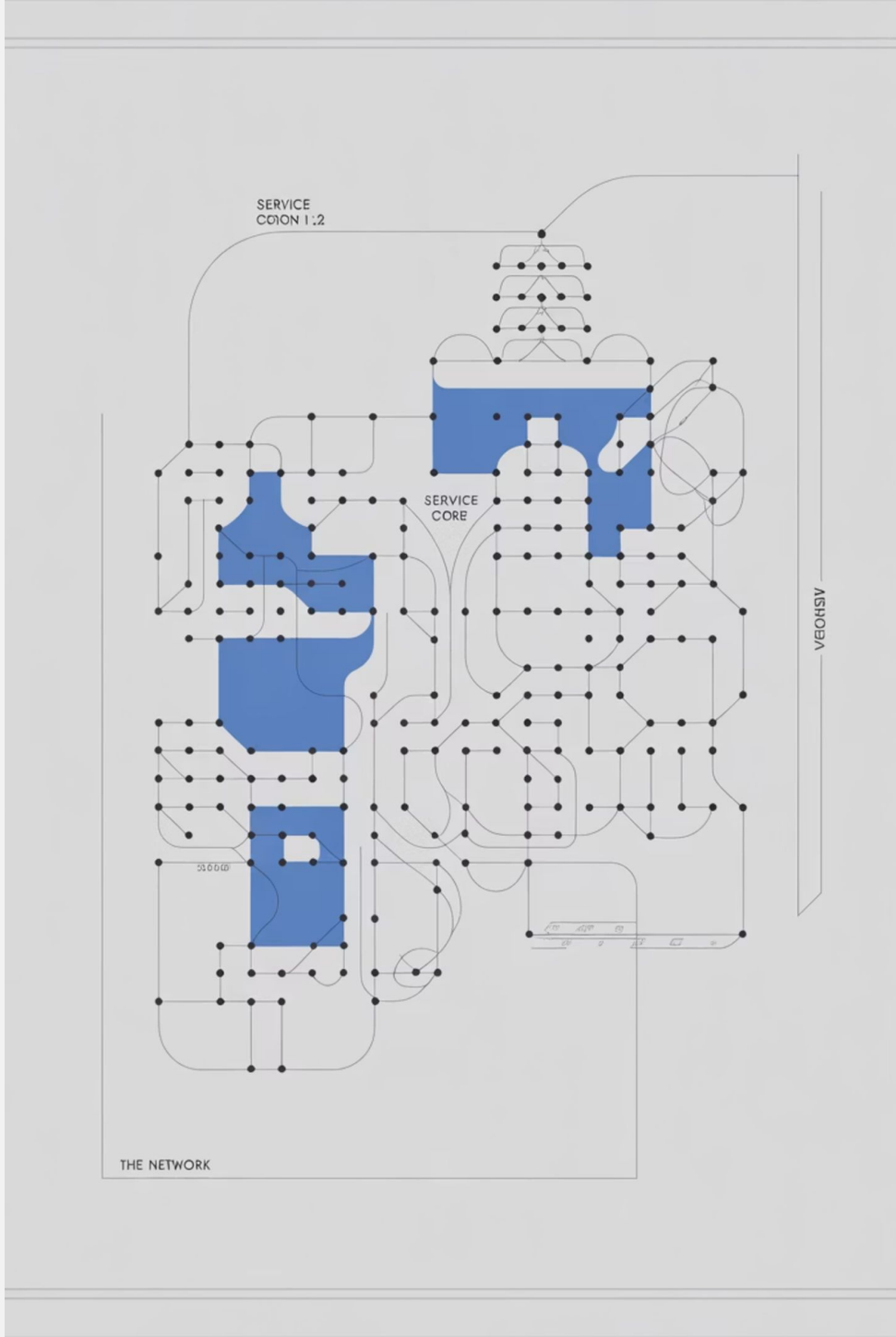
Unified Storage

Single platform for data, configurations, templates, and automation scripts

Step 1

Define Your Service Schema

Model your services exactly as your business needs them – no compromises



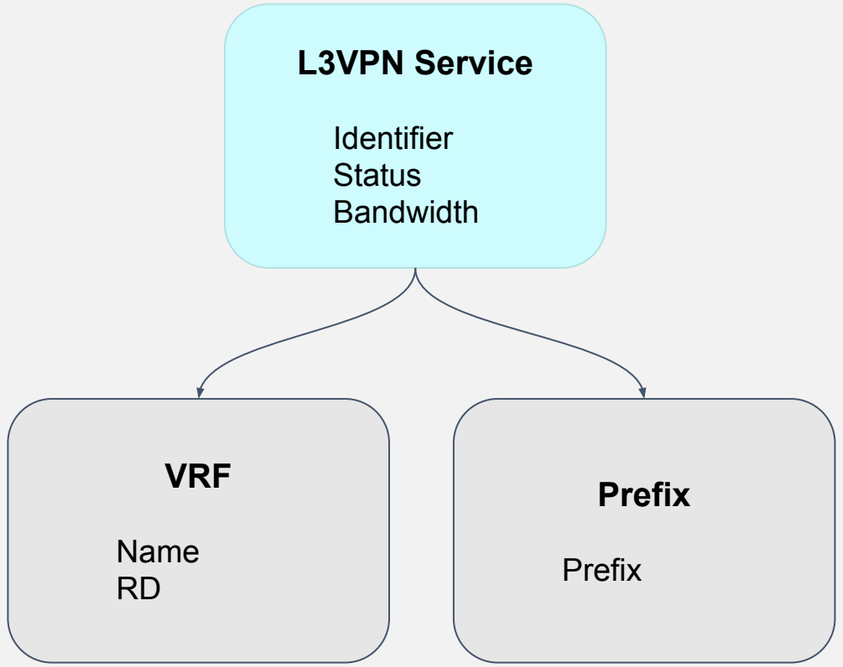
Schema Definition

A schema defines the **structure, format, and constraints** of data, specifying **how data is organized** and interpreted in databases or data models.

It is important because it **ensures data consistency, integrity, and facilitates communication between different systems** by providing a shared understanding of the data's structure.

Build a robust data model that can be leveraged across your entire pipeline

Example: L3VPN Service Definition



No Assumptions, Full Control

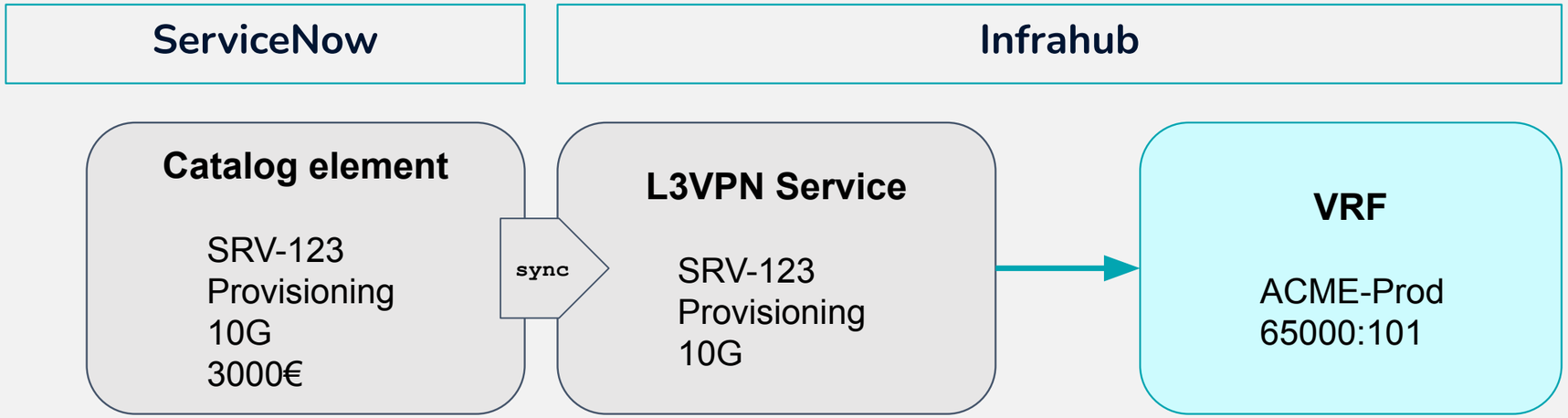
With flexible schemas, **Infrahub lets you define custom types** that match how you design and implement services. No more forcing your services into predefined templates.

example.yml

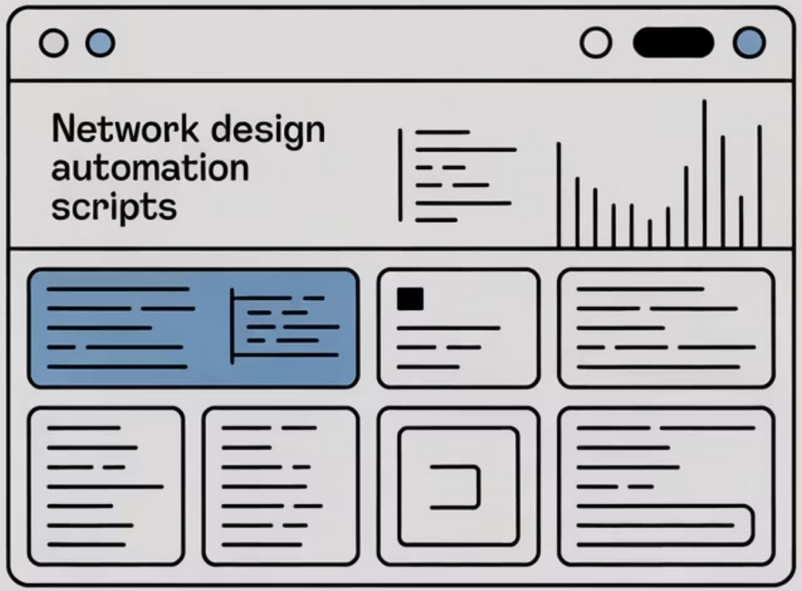
```yaml
nodes:
  - name: L3VPN
    namespace: Service
    icon: mdi:ethernet
    attributes:
      - name: identifier
        kind: Text
        unique: true
        optional: false
      - name: status
        kind: Dropdown
        optional: false
        default_value: draft
        choices:
          - name: draft
          - name: in-delivery
          - name: active
          - name: in-decomissioning
          - name: decomissioned
    relationships:
      - name: vrf
        peer: NetworkVRF
        cardinality: one
        optional: true
```

Integrate various data sources

Customer data synced from ServiceNow automatically populates the service model



Create strong integration and a comprehensive overview of your network implementation.



Step 2

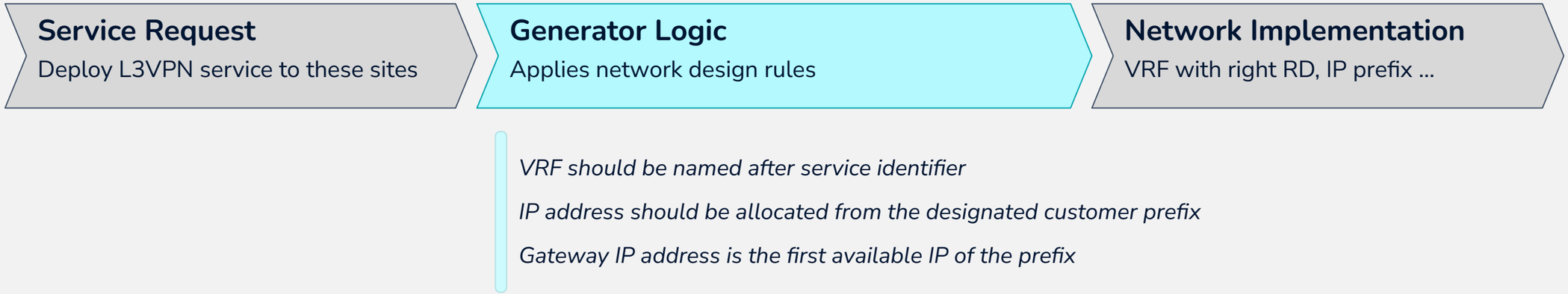
Codify Your Network Design

Transform tribal knowledge into structured, repeatable code

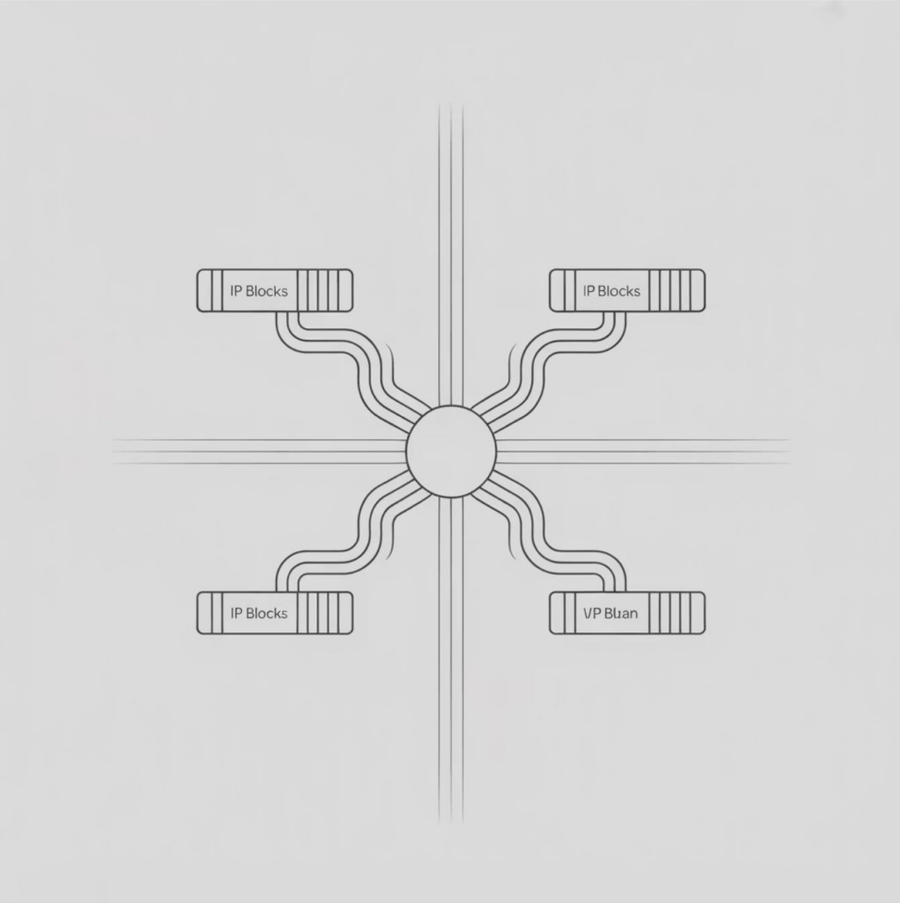
From Tribal Knowledge to Code

> “Network design lives in Visio diagrams and senior engineers' heads”

Infrahub Generators are python scripts that **encode your design patterns** and automatically **translate service requests into implementation.**



Intelligent Resource Management



Define Resource Pools

IP prefixes, VLAN ranges, AS numbers, and other network resources

Automatic Allocation

Works with generators, no more spreadsheet tracking or manual assignment

Guaranteed Uniqueness

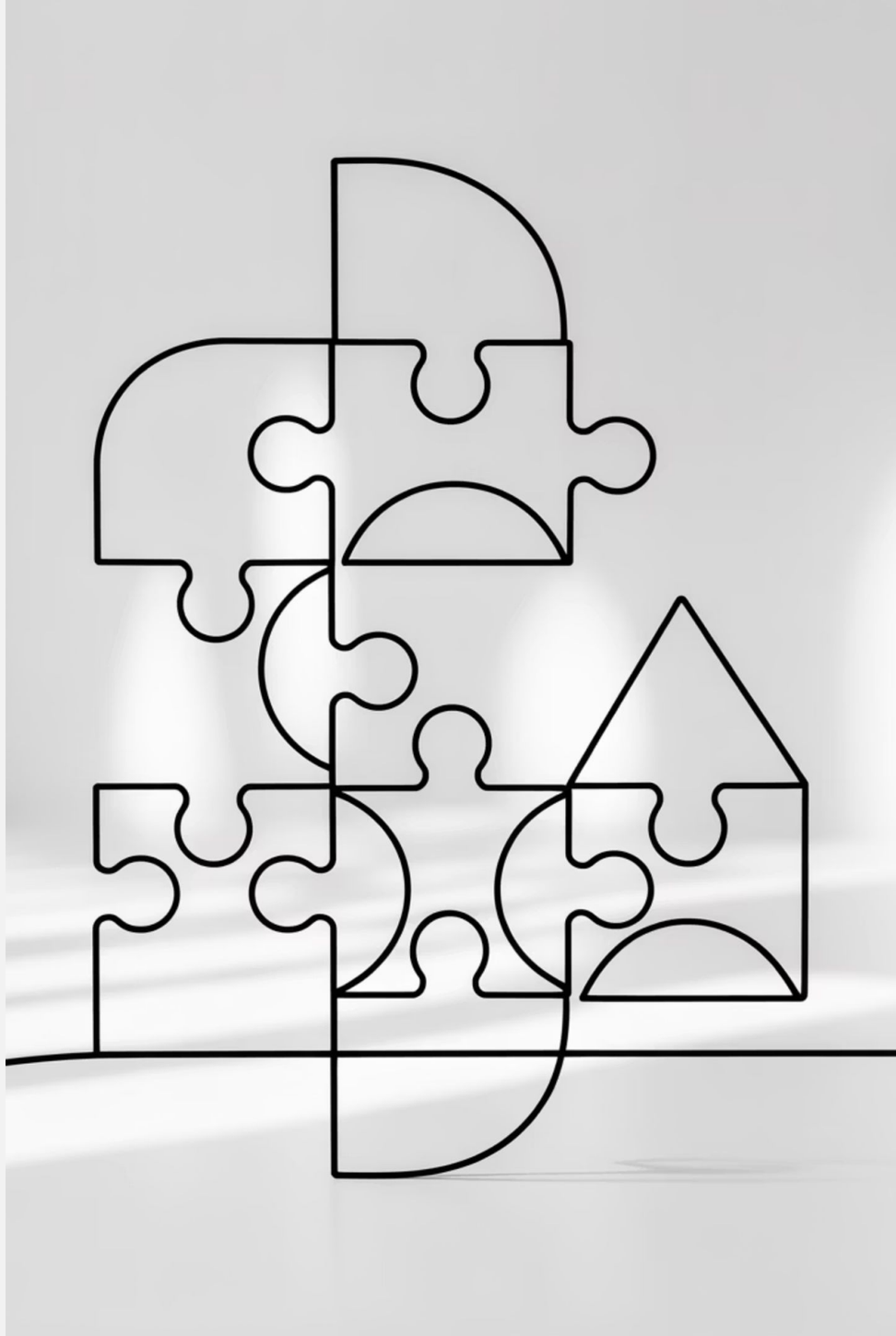
Prevent conflicts before they happen with built-in validation

Result: Ship consistent, predictable services with engineering confidence

Step 3

Implement Services

Transform your service model into vendor-specific implementations



From Data Model to Device Reality

Transforms let you **define vendor-specific logic** that converts a generic service into a deployable configuration for the field.

This approach maintains a **clear separation between your service definition and the technical implementation** details

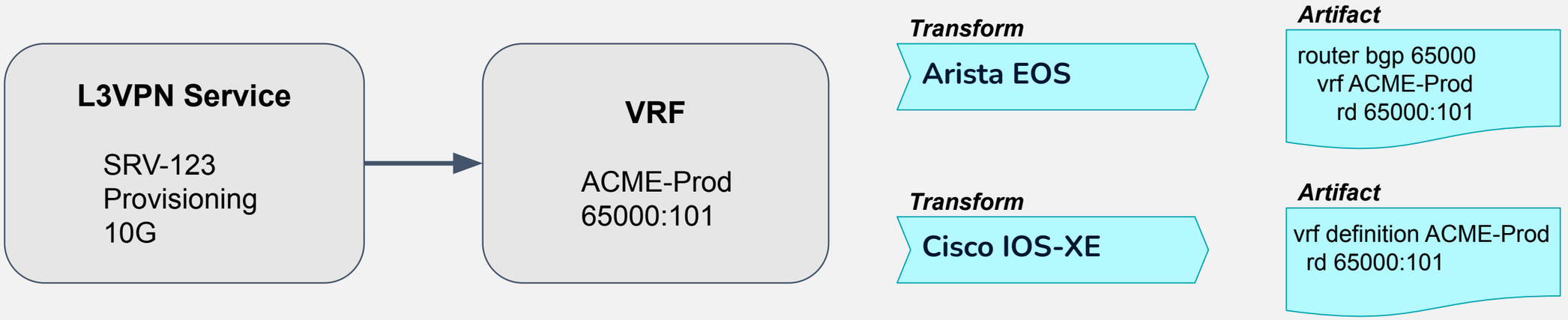
Vendor Abstraction in Practice

Mixed Technology Infrastructure

Same service spanning Cisco on Site A and Arista on Site B

Cross-Platform Services

Cisco routing with Palo Alto firewalls



Vendor complexity is contained in transforms – your service logic remains clean and portable

The Vendor Migration Scenario

> “We're replacing Cisco with Arista in Region 2”

Traditional Approach

Months of configuration rewrites

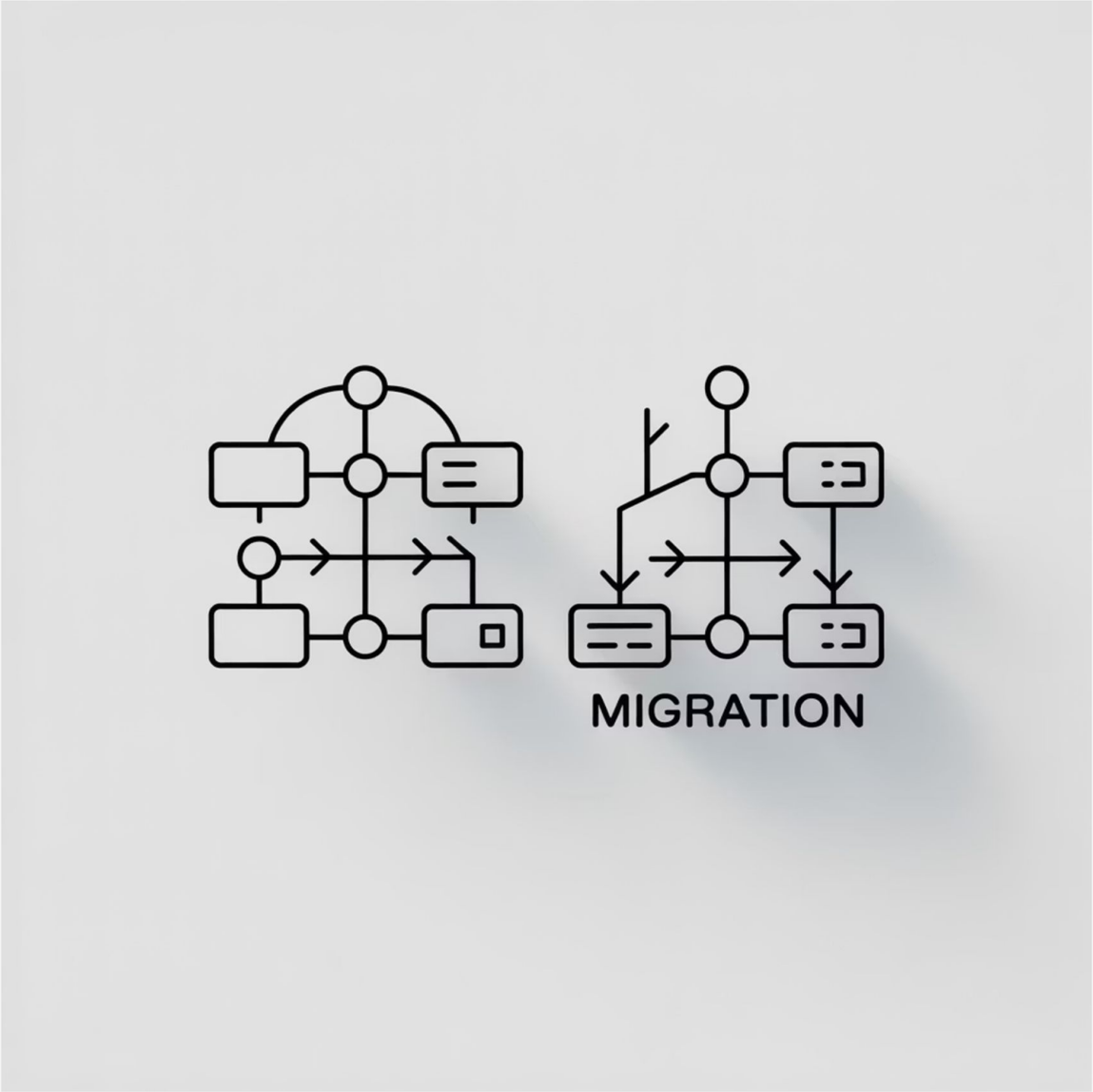
Human errors, hard to validate, impacts the operations ...

With Infrahub

Apply changes to the vendor specific bits (transforms)

Iterate, test and validate in a branch

Change only the vendor specific part - develop, test and iterate with agility and confidence



Deployment Integration

Generated artifacts feed into your existing tools. Infrahub works with your current deployment methods, not against them.

Ansible

Push configurations via playbooks

Cisco NSO

Leverage NSO to deploy configurations

Terraform

Infrastructure as code workflows

APIs & Orchestrators

Direct integration with API

Infrahub focuses on generating the right configurations while leveraging your existing deployment pipeline

Conclusion

Everything Starts with Data

- Flexible Source of Truth with branching and version control
- Your services, your way - not forced into rigid templates
- Model exactly what matters to your business

Robust Service Lifecycle Automation

- Beyond scripts and spreadsheets
- Predictable, tested, repeatable processes
- From service request to implementation

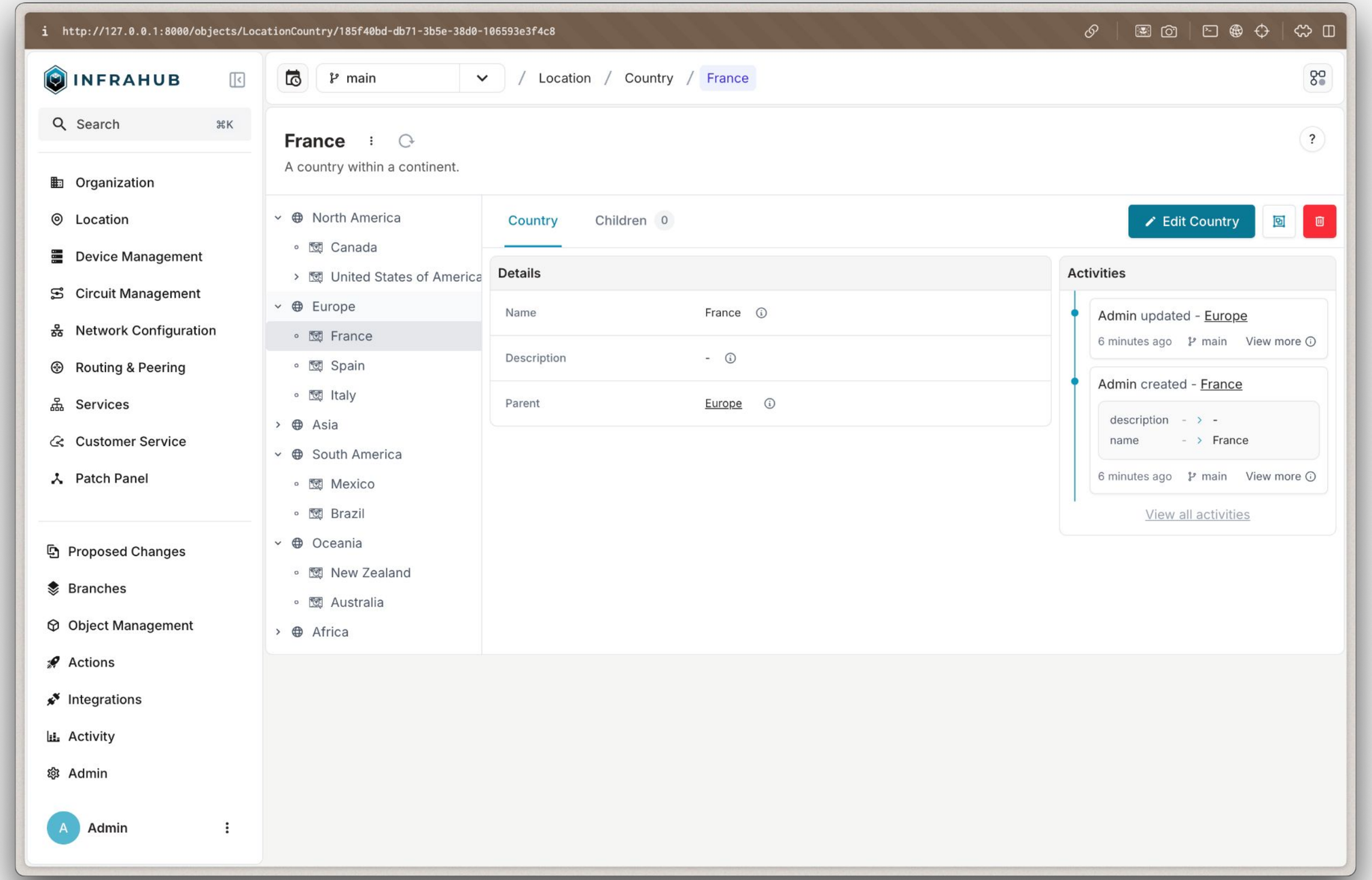
Multi-Vendor Environments

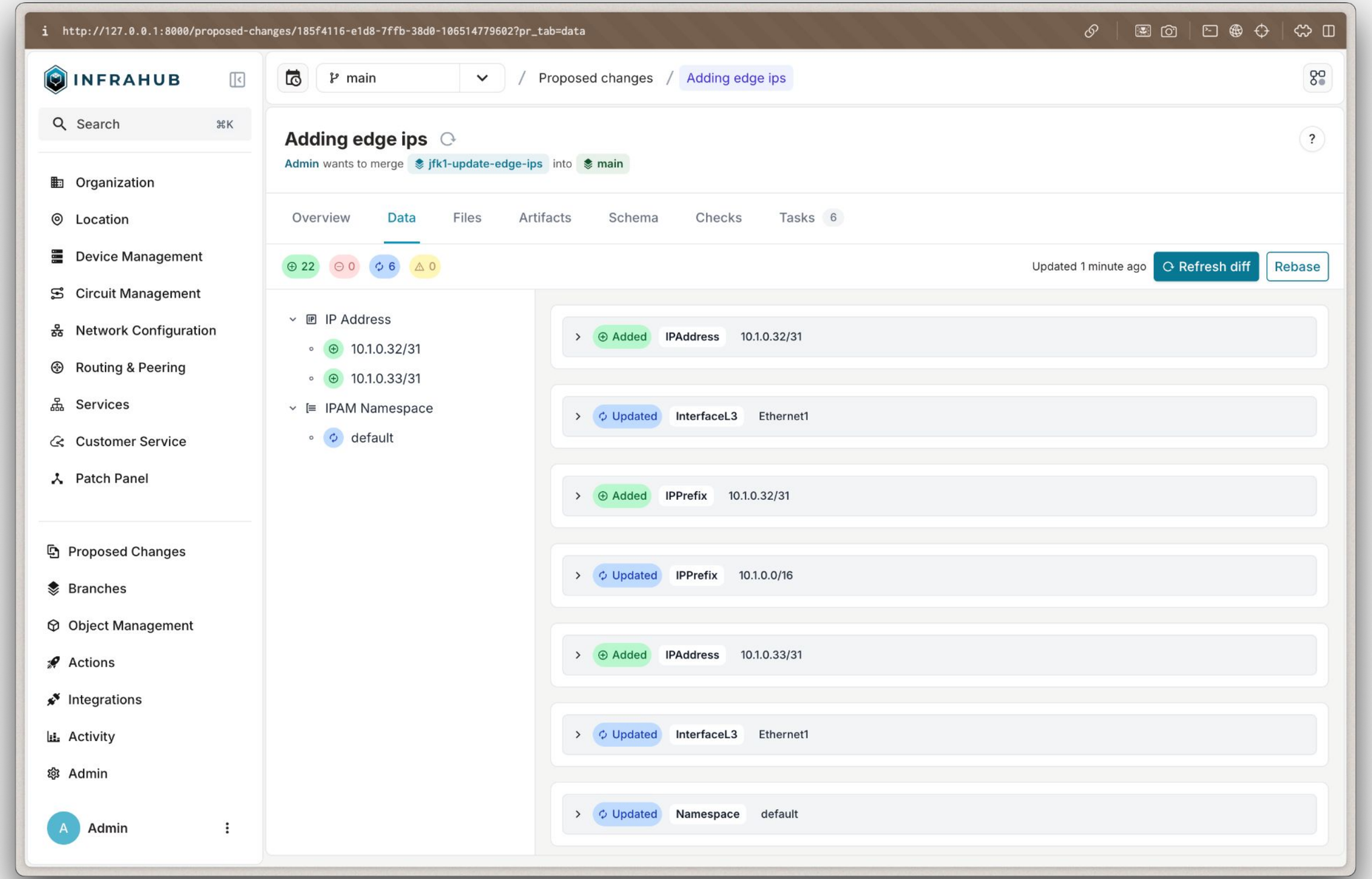
- Abstract vendor complexity
- Support complex multi-vendor products
- Simplify migrations and technology changes

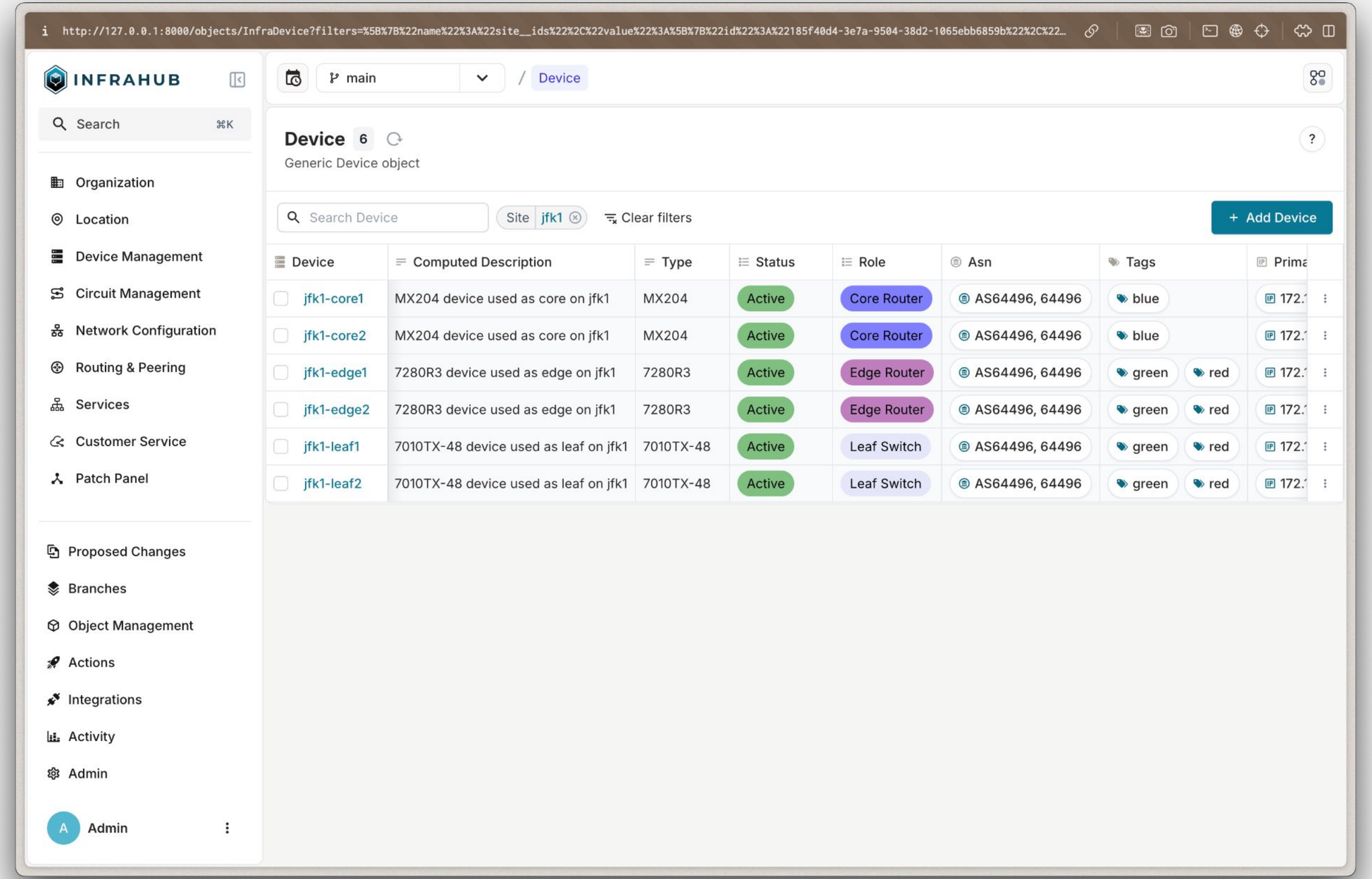
OpsMill

opsmill.com

opsmill/infrahub


http://127.0.0.1:8000/objects/LocationCountry/185f40bd-db71-3b5e-38d0-106593e3f4c8
INFRAHUB
Search ⌘K
Organization
Location
Device Management
Circuit Management
Network Configuration
Routing & Peering
Services
Customer Service
Patch Panel
Proposed Changes
Branches
Object Management
Actions
Integrations
Activity
Admin
A Admin
main
/ Location / Country / France
France
A country within a continent.
North America
Canada
United States of America
Europe
France
Spain
Italy
Asia
South America
Mexico
Brazil
Oceania
New Zealand
Australia
Africa
Country
Children 0
Edit Country
Details
Name France
Description -
Parent Europe
Activities
Admin updated - Europe
6 minutes ago main View more
Admin created - France
description - > -
name - > France
6 minutes ago main View more
View all activities


http://127.0.0.1:8000/proposed-changes/185f4116-e1d8-7ffb-38d0-106514779602?pr_tab=data
INFRAHUB
Search ⌘K
Organization
Location
Device Management
Circuit Management
Network Configuration
Routing & Peering
Services
Customer Service
Patch Panel
Proposed Changes
Branches
Object Management
Actions
Integrations
Activity
Admin
A Admin
main
/ Proposed changes / Adding edge ips
Adding edge ips
Admin wants to merge jfk1-update-edge-ips into main
Overview
Data
Files
Artifacts
Schema
Checks
Tasks 6
22 0 6 0
Updated 1 minute ago
Refresh diff
Rebase
IP Address
10.1.0.32/31
10.1.0.33/31
IPAM Namespace
default
Added IPAddress 10.1.0.32/31
Updated InterfaceL3 Ethernet1
Added IPPrefix 10.1.0.32/31
Updated IPPrefix 10.1.0.0/16
Added IPAddress 10.1.0.33/31
Updated InterfaceL3 Ethernet1
Updated Namespace default


http://127.0.0.1:8000/objects/InfraDevice?filters=%5B%7B%22name%22%3A%22site__ids%22%2C%22value%22%3A%5B%7B%22id%22%3A%22185f40d4-3e7a-9504-38d2-1065ebb6859b%22%2C%22...
INFRAHUB
Search ⌘K
Organization
Location
Device Management
Circuit Management
Network Configuration
Routing & Peering
Services
Customer Service
Patch Panel
Proposed Changes
Branches
Object Management
Actions
Integrations
Activity
Admin
A Admin
main
/ Device
Device 6
Generic Device object
Search Device
Site jfk1
Clear filters
+ Add Device
Device
Computed Description
Type
Status
Role
Asn
Tags
Prima
jfk1-core1 MX204 device used as core on jfk1 MX204 Active Core Router AS64496, 64496 blue 172.1
jfk1-core2 MX204 device used as core on jfk1 MX204 Active Core Router AS64496, 64496 blue 172.1
jfk1-edge1 7280R3 device used as edge on jfk1 7280R3 Active Edge Router AS64496, 64496 green red 172.1
jfk1-edge2 7280R3 device used as edge on jfk1 7280R3 Active Edge Router AS64496, 64496 green red 172.1
jfk1-leaf1 7010TX-48 device used as leaf on jfk1 7010TX-48 Active Leaf Switch AS64496, 64496 green red 172.1
jfk1-leaf2 7010TX-48 device used as leaf on jfk1 7010TX-48 Active Leaf Switch AS64496, 64496 green red 172.1

The Vendor Migration Scenario

Real case: "We're replacing Cisco with Arista in Region 2"

Traditional Approach

Months of manual configuration rewrites

Risk: Human error in thousands of config changes

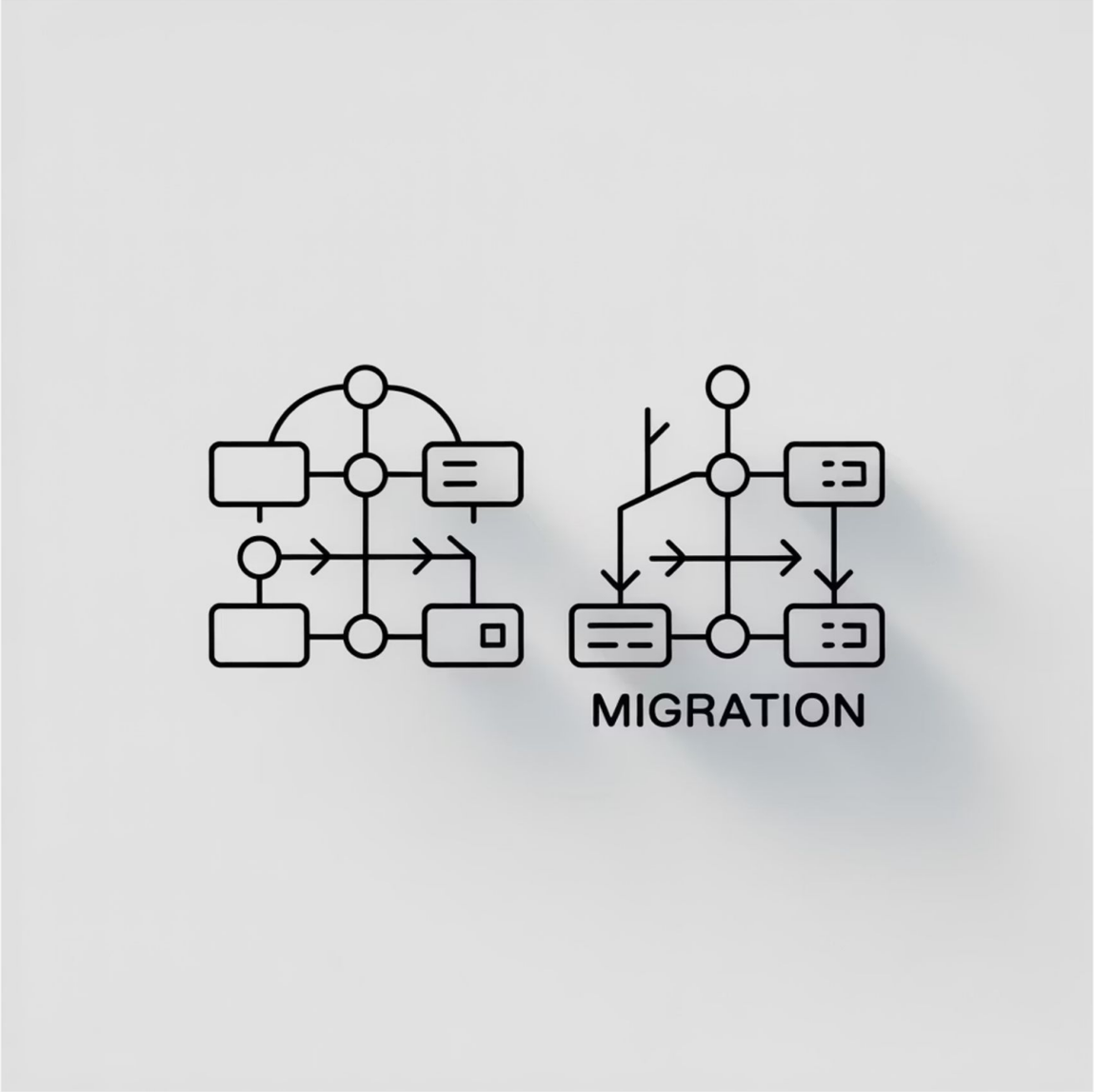
With Infrahub

Create a new branch

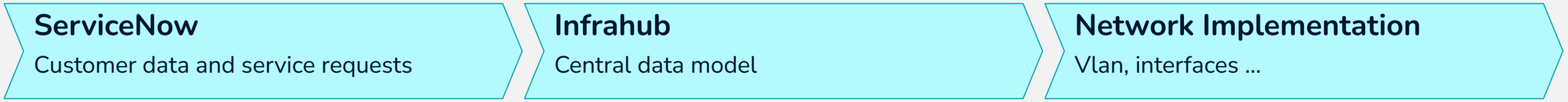
Develop Arista transform

One transform change vs. thousands of manual config rewrites

Same service model, different vendor implementation

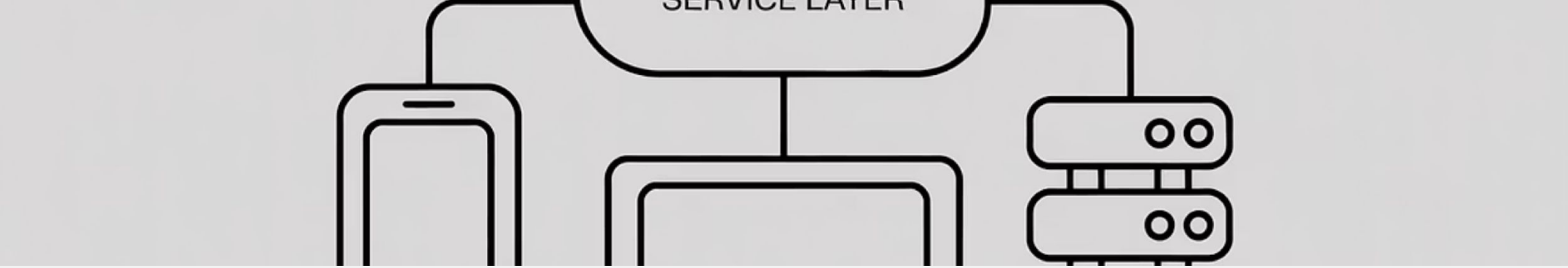


Various data sources



Real-world example: Customer data synced from ServiceNow automatically populates the service model

Key benefit: Create strong integration and a comprehensive overview of your network implementation.



Vendor Abstraction in Practice

Mixed Site Equipment

Same service spanning Cisco on Site A and Nokia on Site B

Cross-Platform Services

Cisco routing with Palo Alto firewalls

Key insight: Vendor complexity is contained in transforms – your service logic remains clean and portable

From Tribal Knowledge to Code

> “Network design lives in Visio diagrams and senior engineers' heads”

Infrahub Generators:

Python scripts that encode your design patterns and automatically translate service requests into implementation

Service Request

"Deploy L3VPN service to these sites"

Generator Logic

Applies network design rules

Implementation Plan

"Provision VRF, apply QoS profile..."

Generators also handle service changes like bandwidth upgrades or modifications to service definitions, automatically determining the required implementation changes

From Data Model to Device Reality

Service Data

Structured service definitions

Transforms

Vendor-specific templates

Devices

Physical implementation

Artifacts

Generated configurations

Transforms: Templates that turn your data model into vendor-specific configurations

Artifacts: The actual configurations stored and versioned in your system

This approach maintains a clear separation between your business service definition and the technical implementation details