Workshop Integrating OSS and BSS

Peter Boers

5th of June, 2023

Agenda

- 9:00 – 9:30 Introduction
- 9:30 – 10:00 Bootstrapping of the development environment
- 10:00 – 10:30 Details of the Orchestrator core models
- 10:30 – 11:00 Coffee 
- 11:00 – 12:30 Development of your first Orchestrator workflow
- 12:30 – 14:00 Lunch (Rogner Hotel) 
- **14:00 – 15:30 Integration of OSS and BSS to your workflow**
- 15:30 – 16:00 Coffee 
- 16:00 – 17:30 Tailoring the Orchestrator to your needs (Discussion)

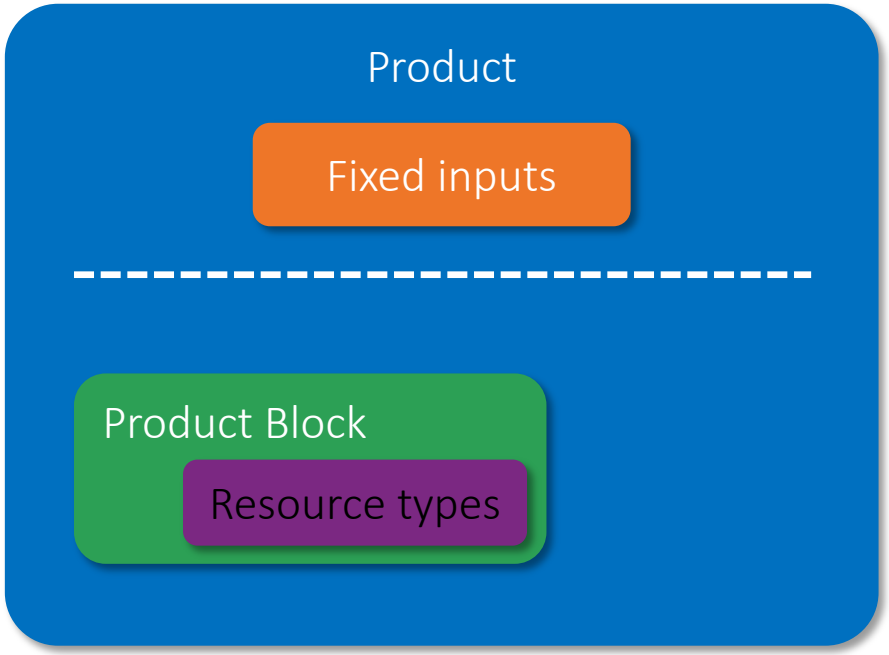
SURF

A quick recap

- Products describe the logical set of resources that together form a service
- Products contain product blocks and resource types that are populated during workflows, they are different per subscription
- Fixed inputs are like resource types, but are attributes that cannot change during the lifecycle of a subscription
- Workflows are run to create, update, delete and validate subscriptions.
- Subscriptions are a logical set of resources that together describe a valid service for a specific customer
- Domain models describe what a subscription to a product must contain for each lifecycle state and how it can be validated.

SURF

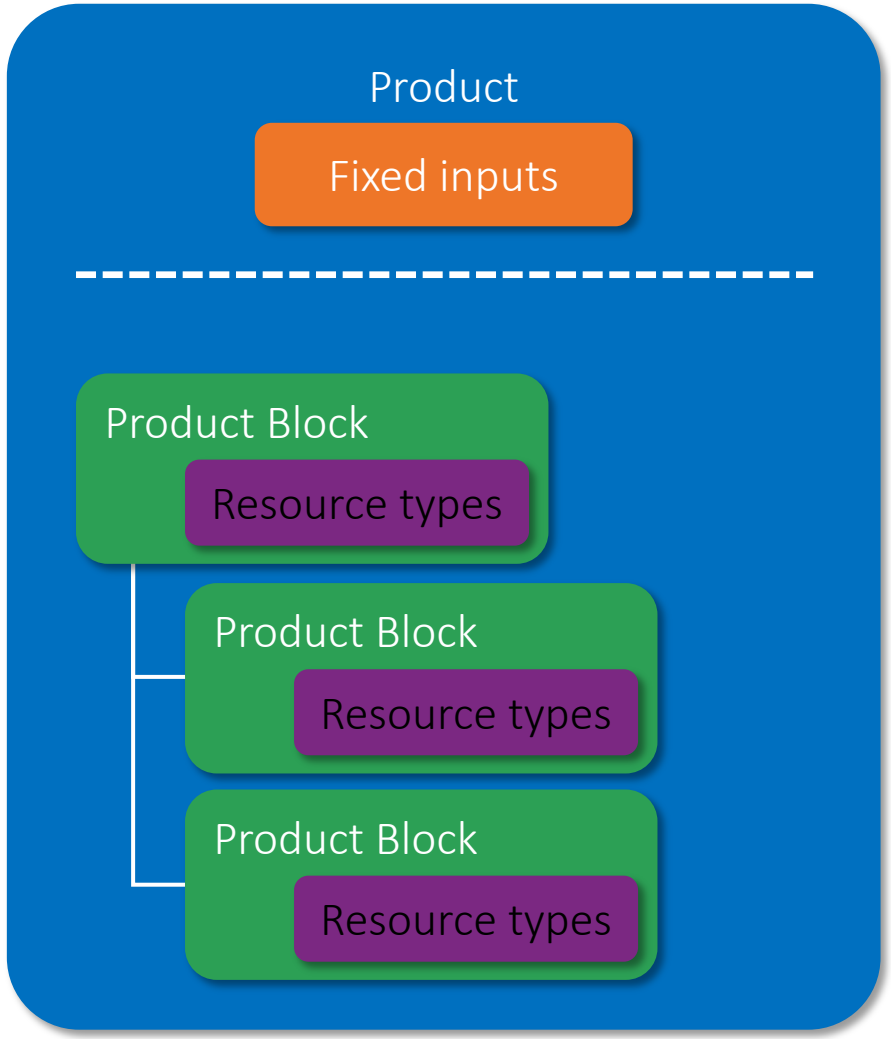
Product Model



Fixed inputs describes immutable parameters of the product

Product blocks are containers for resource types (key/value pairs) and (optional) other product blocks

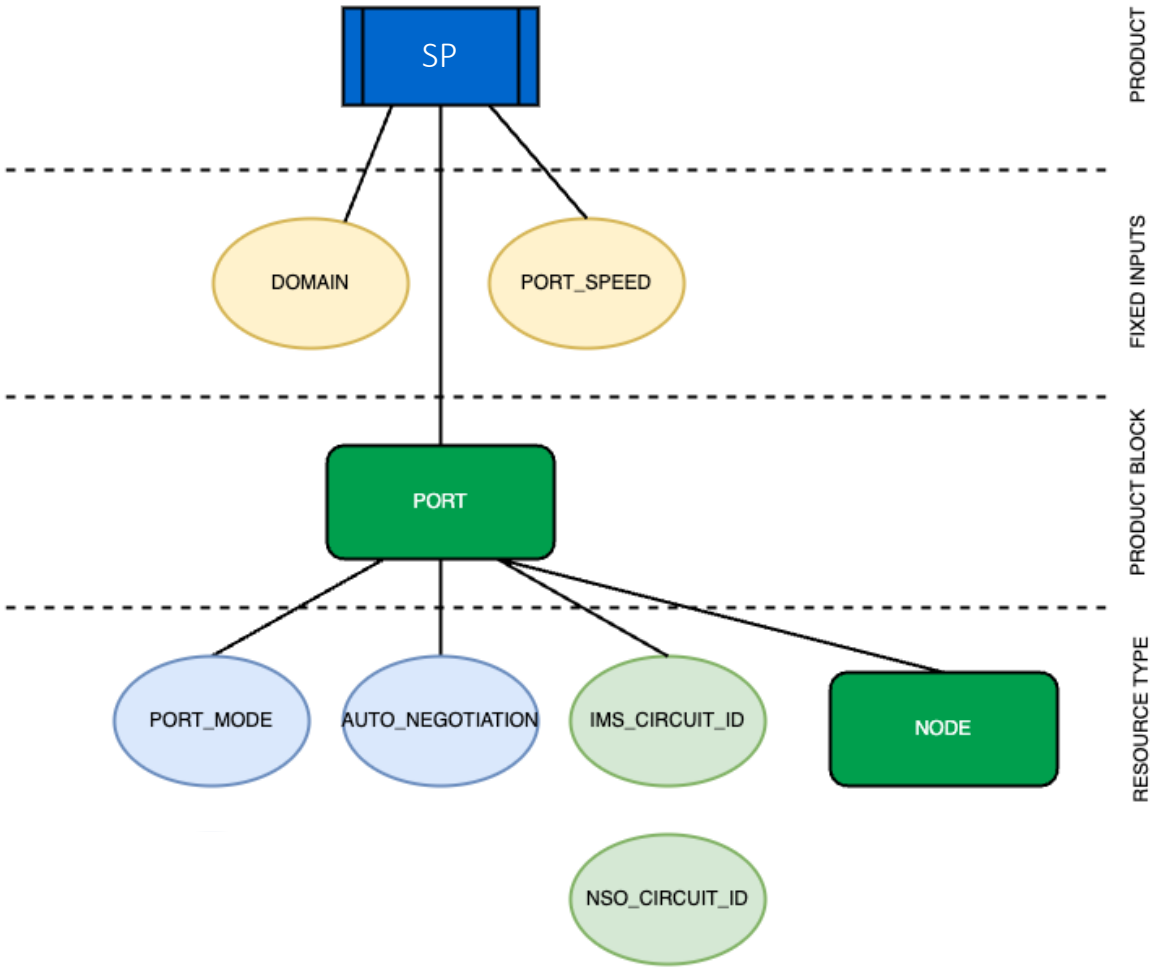
Product Model – little bit more complex

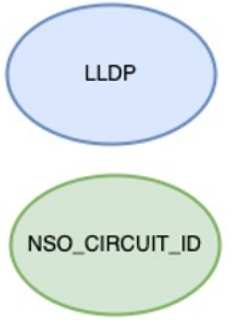


Fixed inputs describes immutable parameters of the product

Product blocks forms building block of resource types (key value pairs)

Simple Port product:

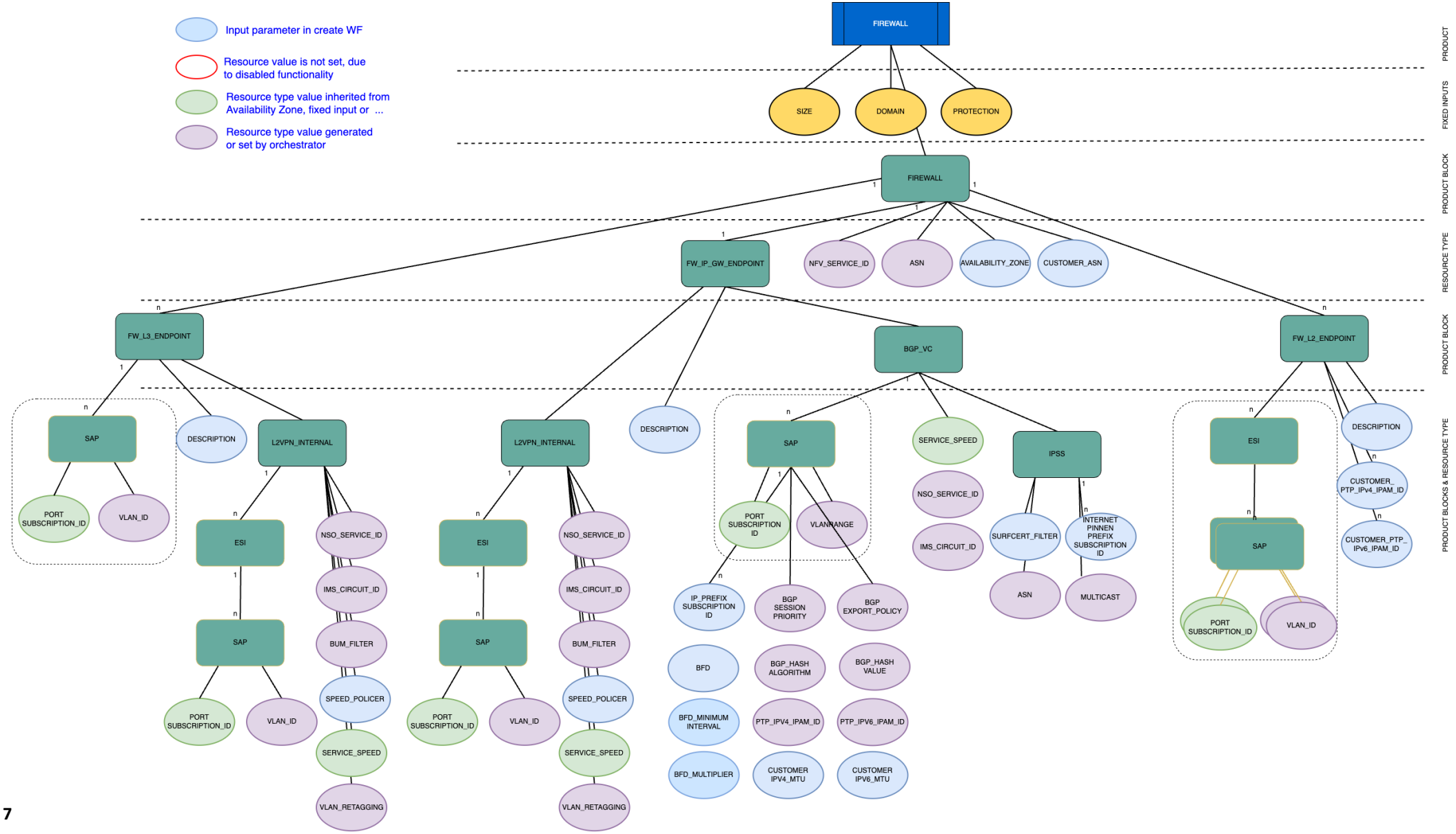




Customer facing parameter - to be filled in by USER

Pointer to OSS/BSS, autogenerated by orchestrator or OSS/BSS

Complex FW product model:



Questions so far?

Integration of OSS and BSS to your workflow

- Exercise in creating an automated service delivery
- Methods of integrating OSS and BSS
- How do you keep your data sane?
- Walk through of the example code
- Write some code:
- Implement the OSS/BSS integration in the create workflow
- Write the modify and terminate workflow
- Quickstarting your workflow writing experience

SURF

What methods can you use to integrate sources of truth to achieve automated service delivery?

Are there NRENS in the room who are willing to share some experiences?

- What method(s) have you chosen?
- What success have you had?
- What challenges did you have?
- Where did you define sources of truth?

Two methods: centralised and decentralised

Centralised

A single point of data entry. A user will enter data at a single point and that will be distributed across all other systems. Most or all validation logic is centralised

Pros: Very reliable, easy to trust and debug, single pane view

Cons: Rigid, time consuming to implement, paradigm shift in processes and data management

Decentralised

Multiple data entry points. A user will enter data in the CMDB, CRM and/or IPAM. Validation logic is handled very close or at the source of truth.

Pros: Flexible, less of a change in processes, easy to start, less data migration

Cons: More checks and balances, difficult to control, consistency not guaranteed

Exercise: Create a layer3 circuit for ACME corp. at NREN Wile E. Coyote

In a group of 3 – 4 people, do the following exercise:

- Design an automated process (workflow) through which you could deliver a layer3 circuit for customer ACME corp.
- Include as much of the process as you can: i.e how you can gather requirements for the service, what the automation flow looks like, what resources you need to configure the service (nodes, interfaces, VRF ids etc)?
- Define where your sources of truth live and how they interact to create working network configuration.
- Describe each step in as much detail as possible
- How do you know when the process is finished?
- When do humans interact with the automated process?

Take 30 minutes to discuss this and prepare a short pitch about what this would look like.

SURF

Centralised: The orchestrator first mentality

Integrating OSS and BSS

- The orchestrator is designed to be in control of all data
- The system works best when all data entry is done whilst running workflows, the benefits include:
- Defining your validations once
- It uses repeatable workflows to translate the data into valid OSS/BSS objects
- It eliminates data entry errors
- **This process only allows sane data to enter the system**

SURF

Mitigating the centralised architecture risk i,e closing the loop

The greatest risk to the centralised approach is someone going outside orchestration and making a change in an external system.

- Writing validation tasks that crosscheck all data
- Is the inventory in-sync with the network and vice versa
- Do all service parameters correlate with inventory and subscription intent
- Closing the loop makes sure you detect inconsistencies as soon as possible
- Closing the loop makes changes reliable as they enable you to block changes to services if they do not comply to business rules.

How can you enforce data consistency of a subscription

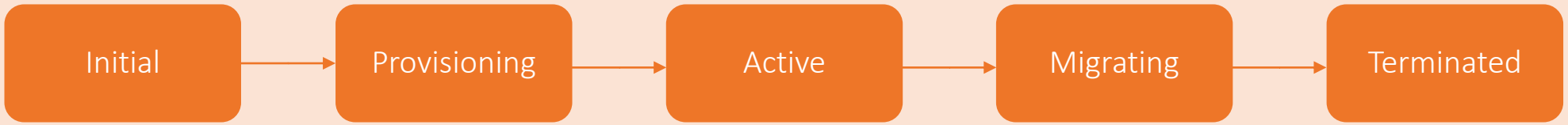
A subscription has two means of “locking” to guarantee “safety”

- Subscription Lifecycle
- The orchestrator gives you five states initially: Initial, Provisioning, Migrating, Active and Terminated
- An insync Boolean:
- This controls if the subscription is insync with all subsystems or not.

Subscription Lifecycle

Subscription lifecycle helps define what a subscription should look like a certain point in it's existence

- E.g At state *provisioning*, a subscription might not have access to all workflows and it may not be filled in completely. Some properties may not be known.
- At state *terminated* all external id's may be removed except for a reference to telemetry.
- When writing validation workflows, subscription lifecycle helps the system to decide what to check, and when you check it, what it should look like

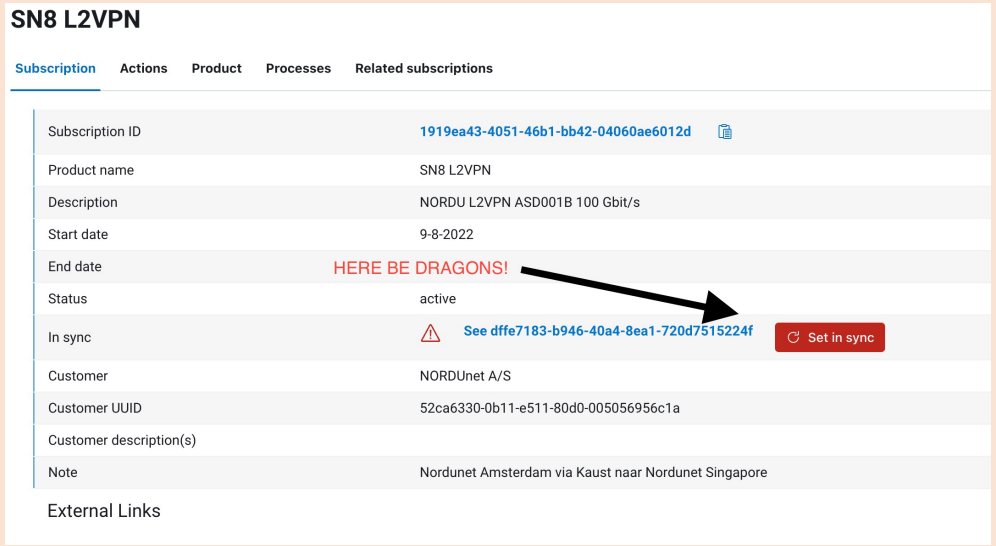


SURF

Insync boolean

Subscription insync helps the system understand if it is safe to run a workflow on a subscription or not.

- Only one workflow may run on a subscription at any given time.
- If a validation workflow is ”failed” the subscription remains out of sync.
- This can be overruled….



Mitigating the centralised architecture risk.

Check everything!

Gamify keeping data sane!

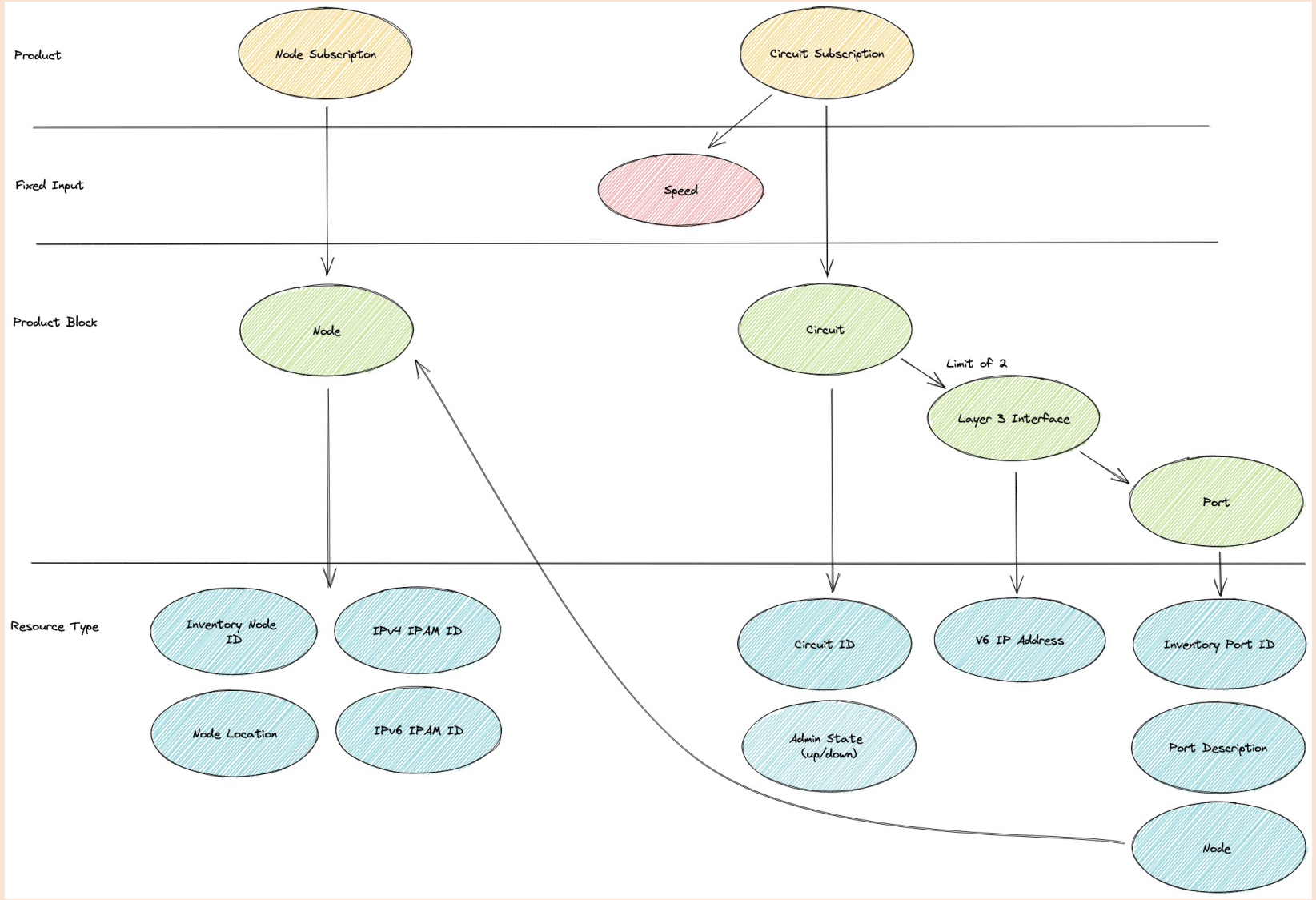
Do it from day one!

Questions so far?

Circuit product workflows (coding)

- Create a circuit between two nodes.
- Use only active nodes that are ready to accept traffic (according to the business rules)
- Use two available interfaces on each node and create the circuit
- Administer the circuit in Netbox (coding)
- Write the modify and or terminate workflows

Domain models



Architecture

- Input steps vs steps
- Workflow targets
- Testing

SURF