Infrastructure and Services as Code getting Netops and Devops together

Automation across IT technologies

Markus Harbeck - Principal Architect, CX EMEA Germany, @mhgrisu
Joerg Reinecke - Architect, CX EMEA Germany

TECOPS-2112

CCIE #8087
CCDE #20130015

CCIE #8300
License to reboot

cisco Live!

slido

Please download and install the Slido app on all computers you use

Join at slido.com #1136142

ⓘ Start presenting to display the joining instructions on this slide.

slido

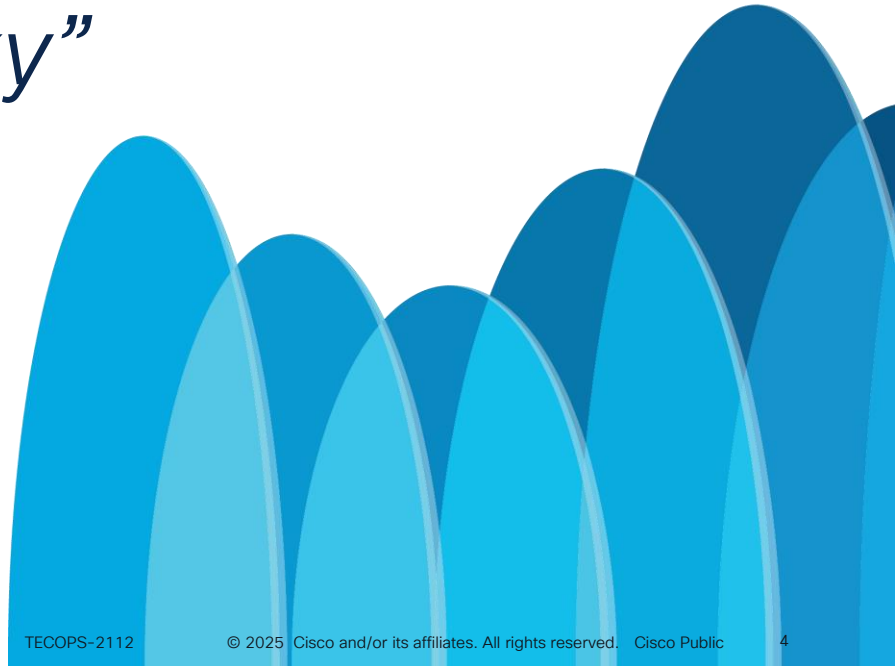
Please download and install the Slido app on all computers you use

Welcome to Cisco Live! Is this your first time to Cisco Live Amsterdam?

ⓘ Start presenting to display the poll results on this slide.

Short Hint: “My English might be bad but although sexy”

Source: Henning Bornemann – “Thank you for Deutsche Bahn”



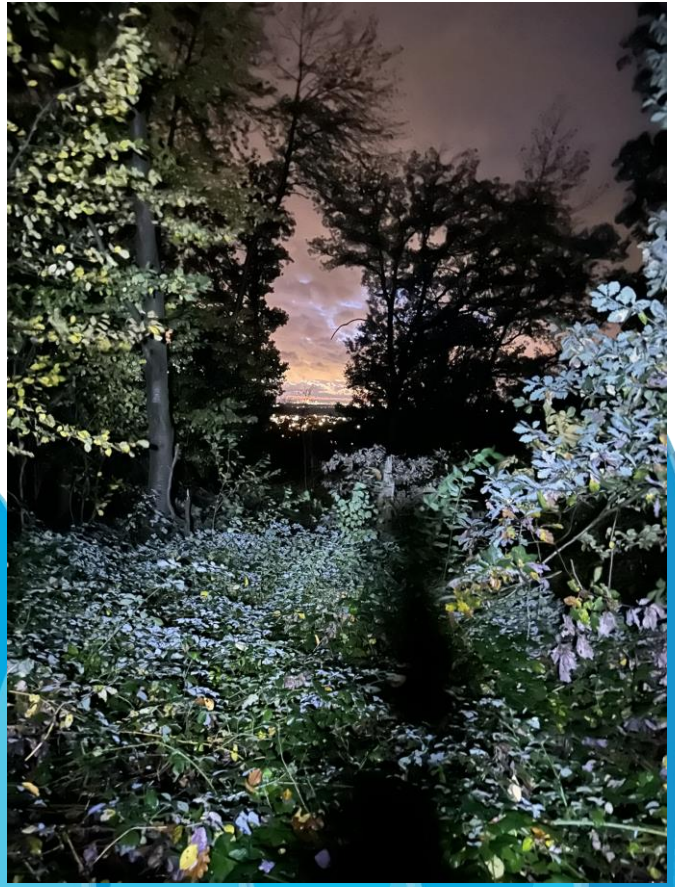
I am ...

...a network admin

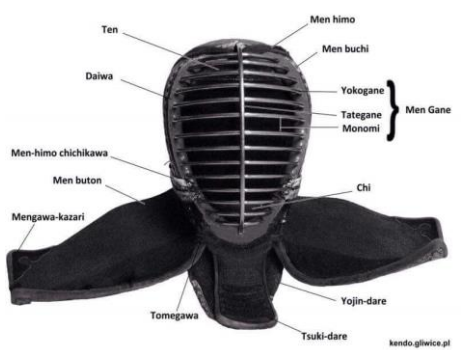
And you ?

“We lose most of our time by trying to save time”

“A Customer”



Who is Joerg?



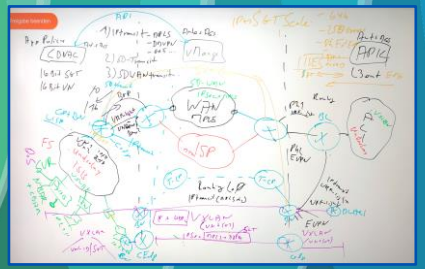
Personal

- Location: Eschborn, Germany (near Frankfurt) living in Mainz
- Interests: My family, my three kids, Kendo, reading books

My Background

- Craftsman (learned "Elektromechaniker" in 1987 to 1990
- Was 6 years in German Airforce, used to repair parts of Tornado electronic systems
- Started 1996 with SNA (System Network Architecture) from IBM
- CLI Junkie since 1997 for all Routing and Switching
- CCIE 8300 for routing and switching with license to reboot.
- Joined Cisco October 2007
- Learned since that about every 2 year a new technology
- Start learning 5 years ago about automation of network infrastructure

CCIE #8087 CCDE #20130015



Who is Markus Harbeck?

Personal

- Location: Eschborn, Germany (near Frankfurt) living in Bavaria
- Interests: My family, my two kids, my dog, horseback riding, motor cycling, hiking, flying drones for fawn rescuing

My Background

- CLI Junkie since 1996 for all Routing and Switching
- Joined Cisco October 2010
- Before: 12 years, operations, engineering, application engineering
- Orchestration and Cross Domain
- Analytics, assurance, automation and migration projects

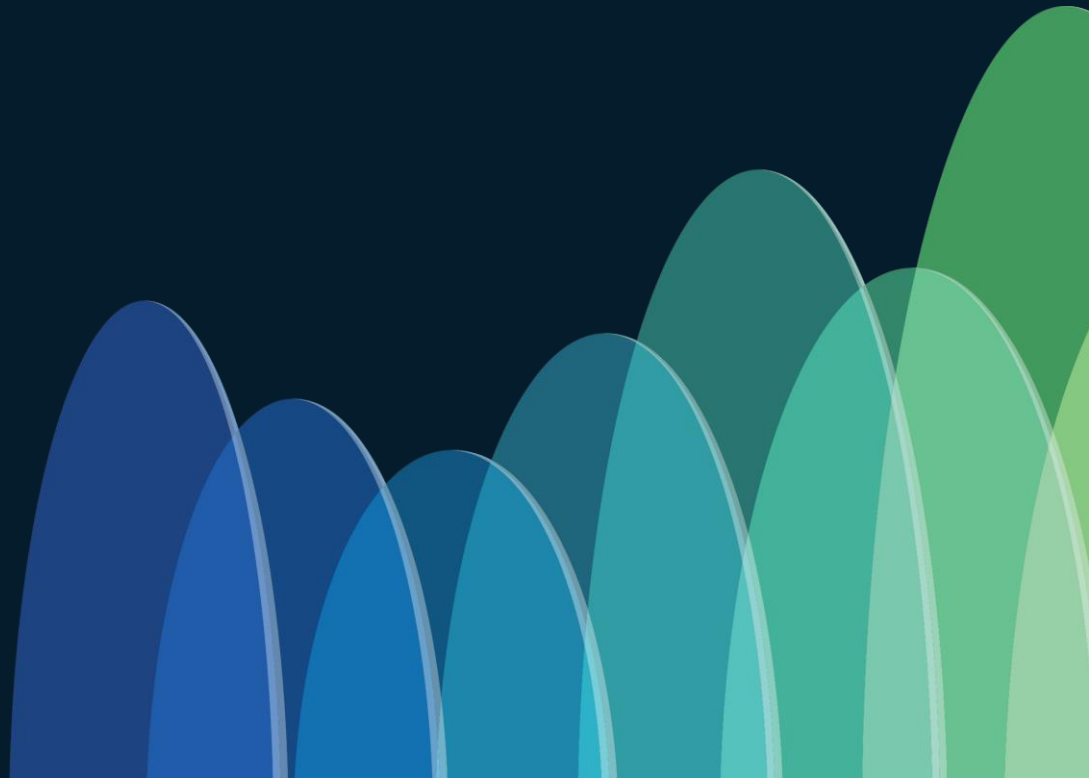
→ first guy who compared horses and (SDN) intent

Copyright by Saskia

My kids view on cross domain network design

Copyright by Hanna

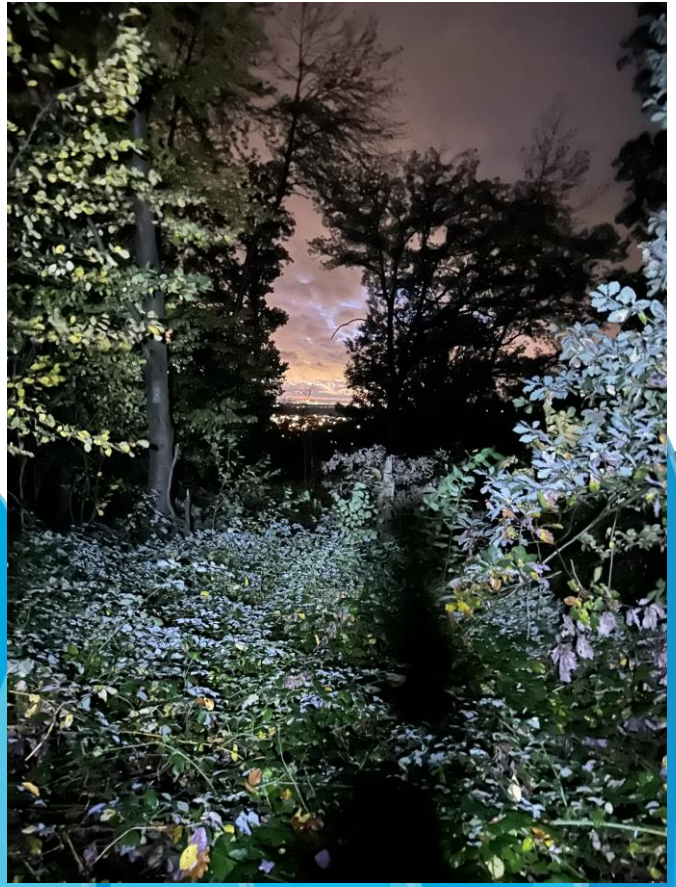
Introduction



Agenda

- Introduction
- Automation in action
- Services as Code & Automation
- Services as Code for
  - ACI
  - SDWAN
  - SDA
- Next steps in automation
- Summary and conclusion

”Increase Efficiency with Service as Code for scalable, repeatable infrastructure automation."



You can do everything yourself with infinite time and money

Unique design

Long build time

Questionable outcome

Extensive skillset

No warranty

Focus on build

No anticipated costs

Service as Code to let you focus

slido

Please download and install the Slido app on all computers you use

Are you using Infrastructure as Code already?

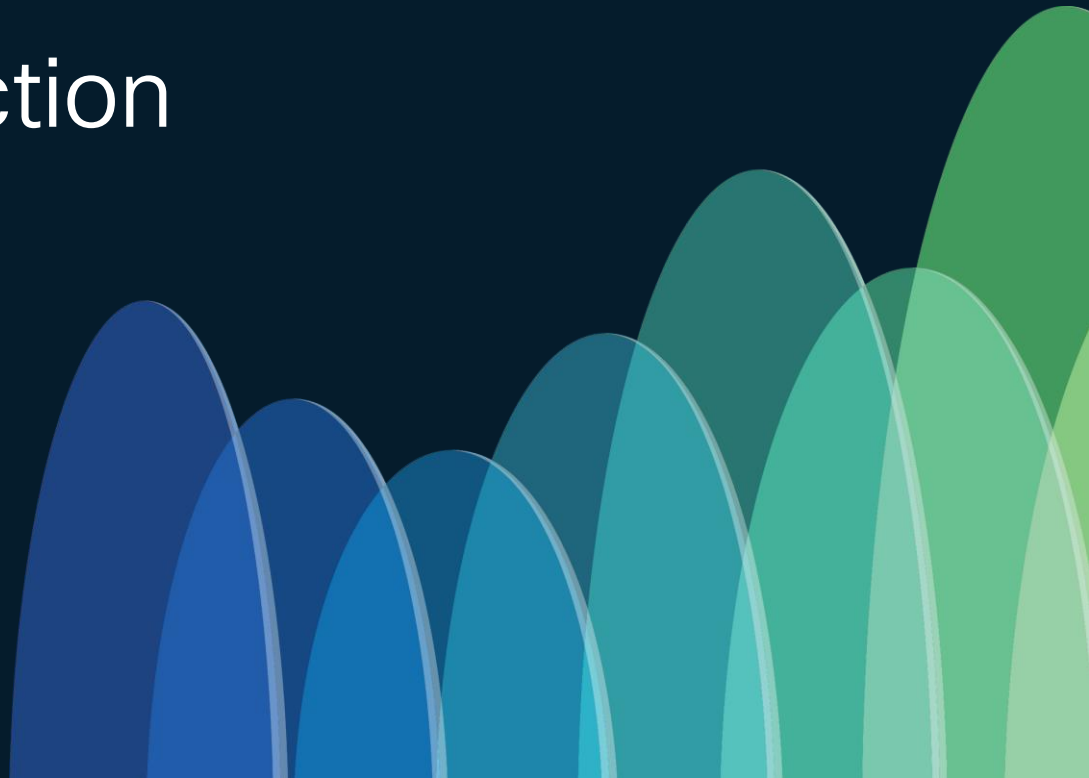
ⓘ Start presenting to display the poll results on this slide.

Session Flow

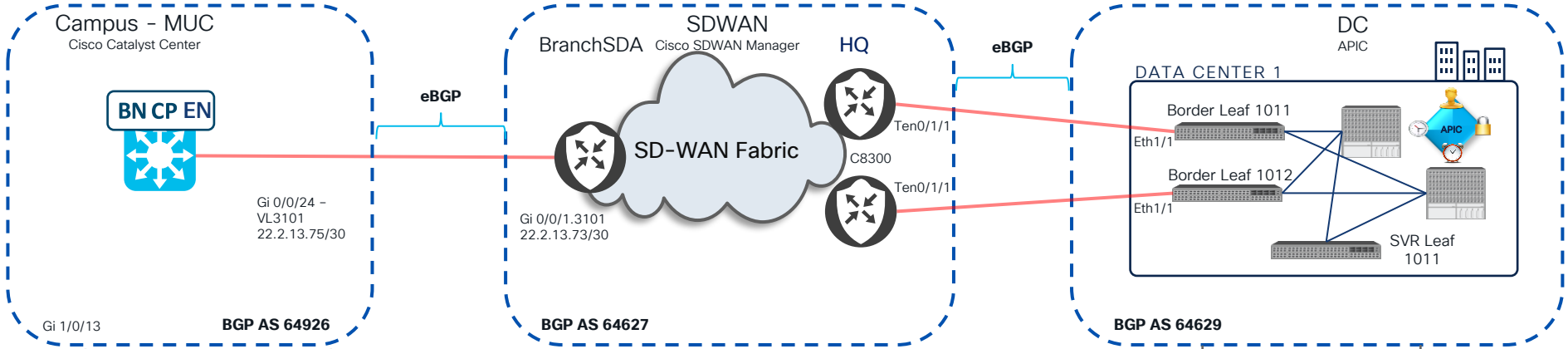
1:45 PM – 2:00 PM	Talk	**Intro**
2:00 PM – 2:30 PM	Apply and automat	**Automation in action – Demo – what happened**
2:30 PM – 3:30 PM	Destroy	**Services as Code and Automation**
3:30 PM – 4:00 PM	Break	Break
4:00 PM – 5:45 PM	Apply and automat	**Service as Code domain automation**
4:15 PM – 4:45 PM	Apply and automat	ACI as Code
4:45 PM – 5:15 PM	Apply and automat	SDWAN as Code
5:15 PM – 5:45 PM	Apply and automat	Catalyst Center (for SDA) as Code
5:45 PM – 6:05 PM	Talk	**Next steps in automation**
6:05 PM – 6:15 PM	Talk	**Summary with Q&A**

Apply and automat
Break
Destroy
Talk

Automation in action
– Demo –



LAB Time → Yes let's start with a demo



Physical Demo Topology

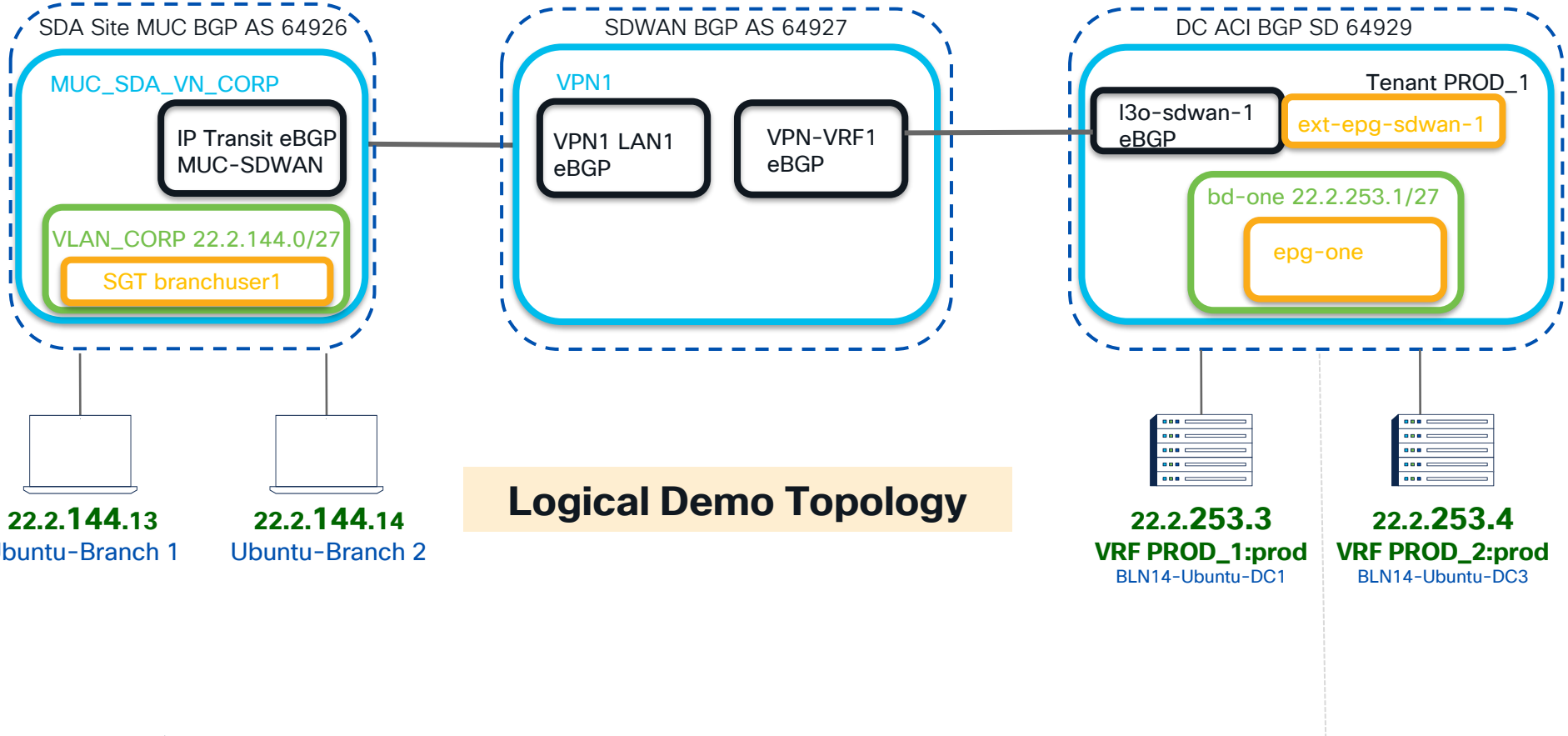
22.2.144.13
Ubuntu-Branch 1

22.2.144.14
Ubuntu-Branch 2

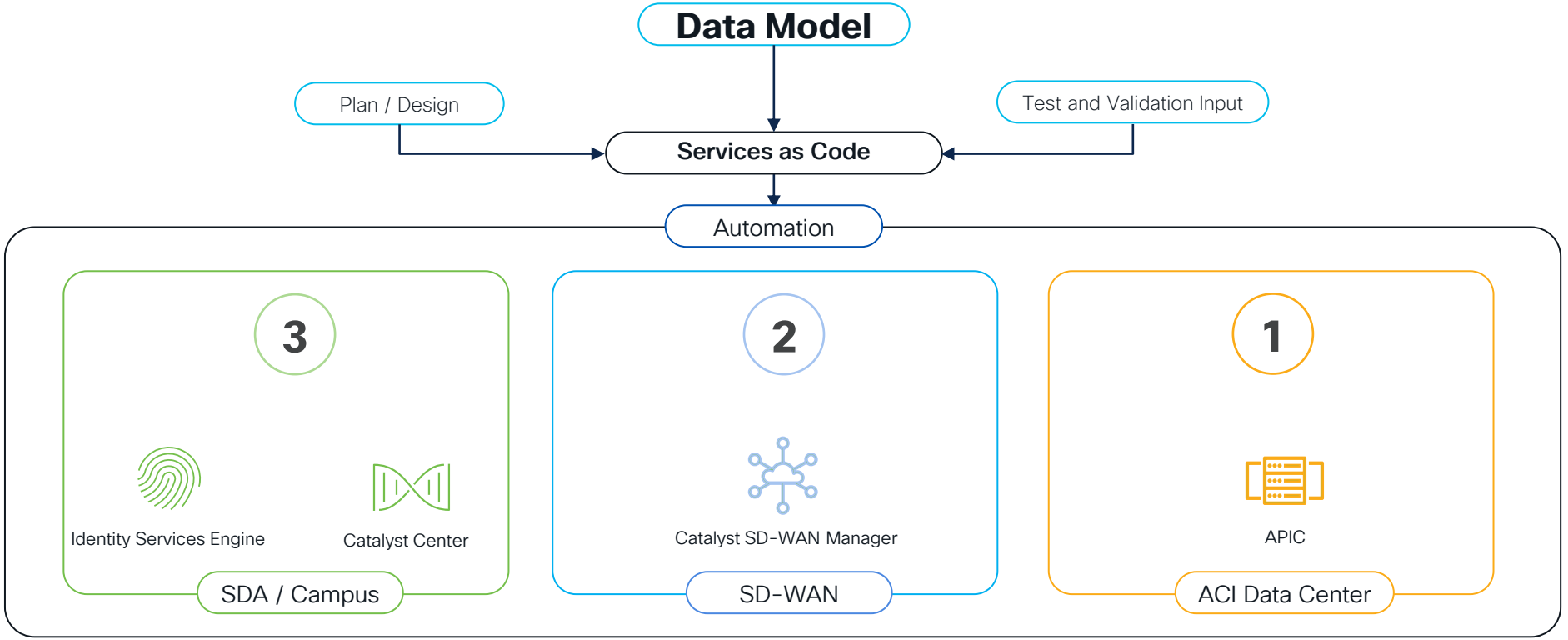
22.2.253.3
VRF PROD_1:prod
BLN14-Ubuntu-DC1

22.2.253.4
VRF PROD_2:prod
BLN14-Ubuntu-DC3

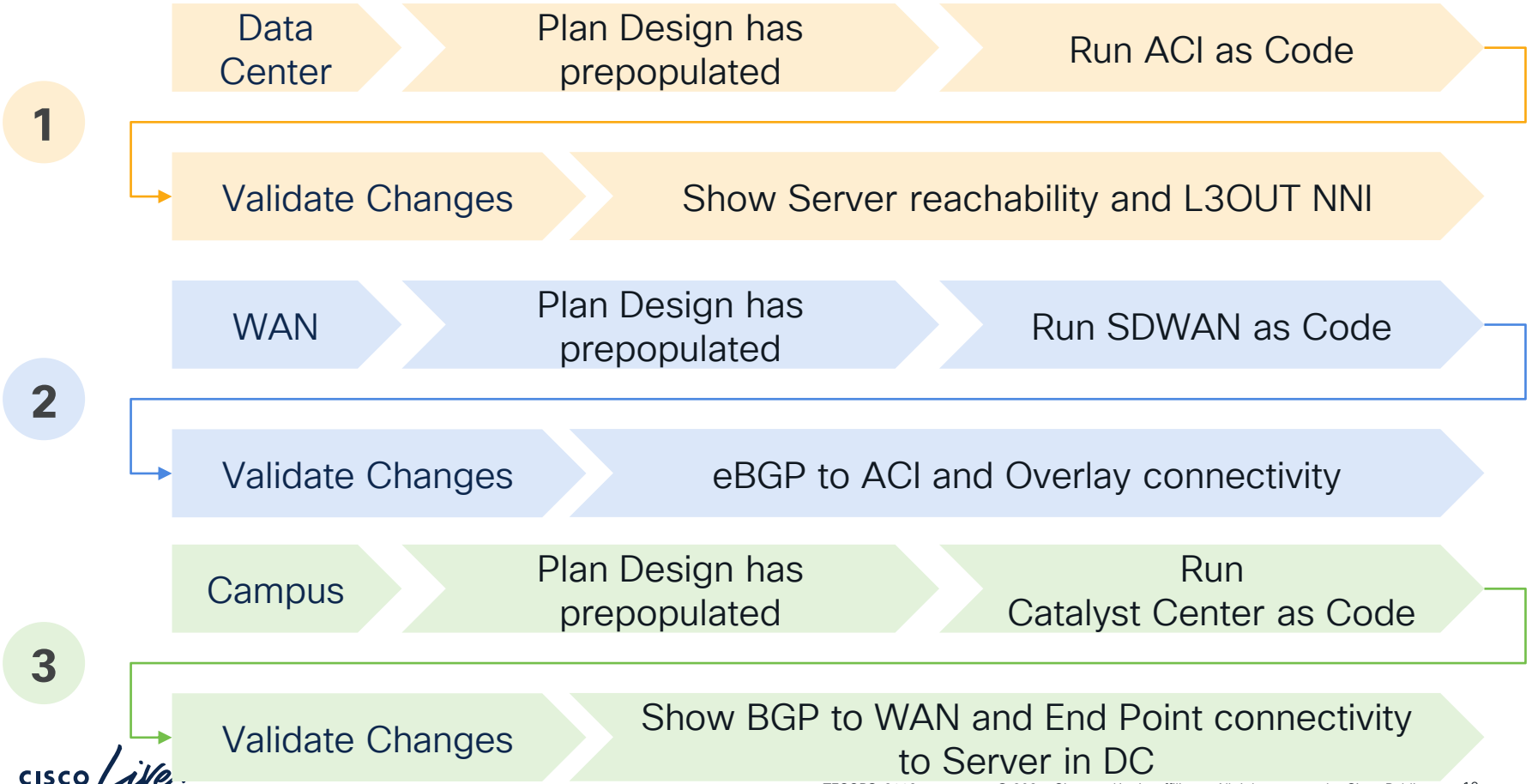
Demo Logical view



LAB / Demo Architecture view



Demo Introduction



1st run what to expect

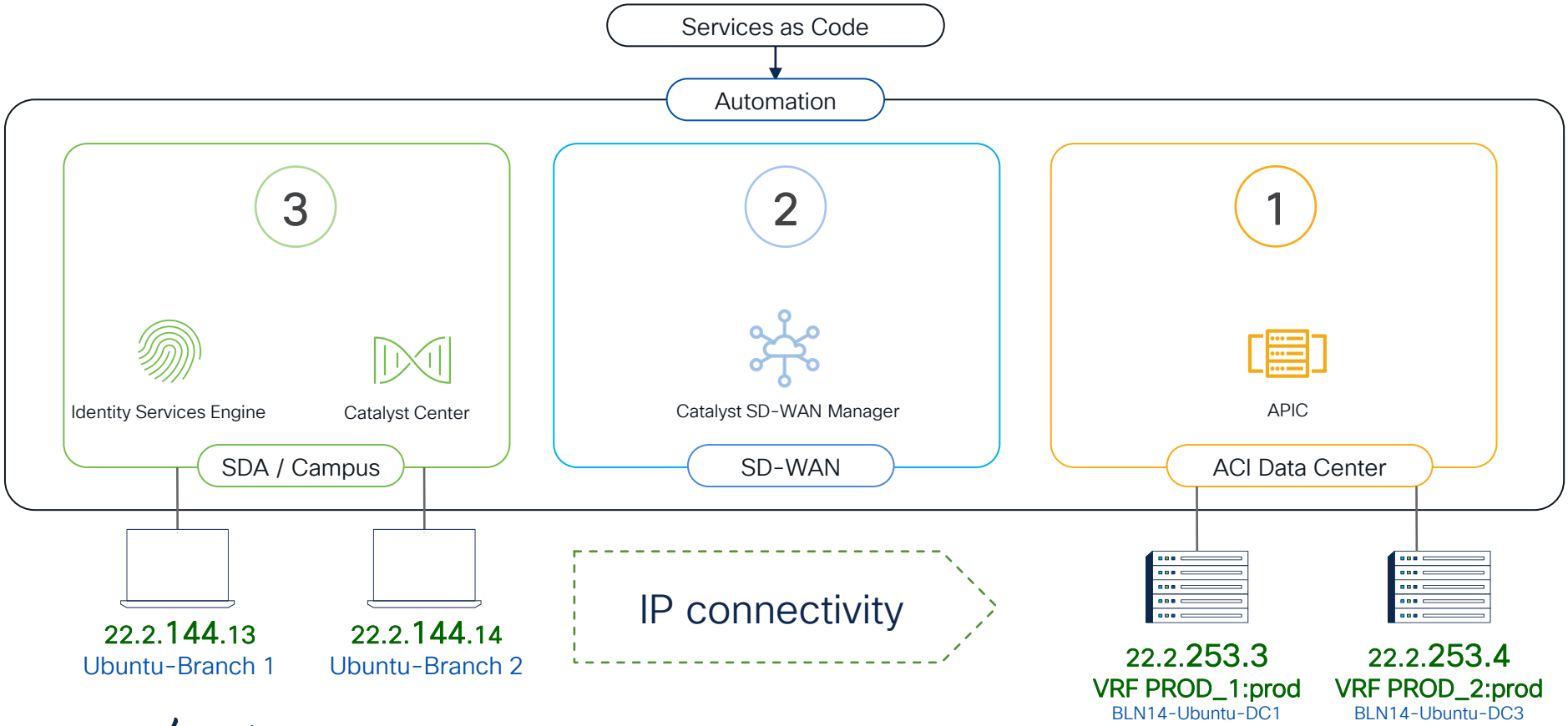
- Day 0 deployment of ACI, SDWAN and SDA with NNIs
  - Controllers and some initial setup has been done
- Create all required configurations to get User / Endpoint connectivity to DC Services

Note: End to end automation in this demo will show the outcome – the session will go through into every detail what happens under the covers

→ We will NOT configure via UI

Demo

Demo Time → expected Outcome



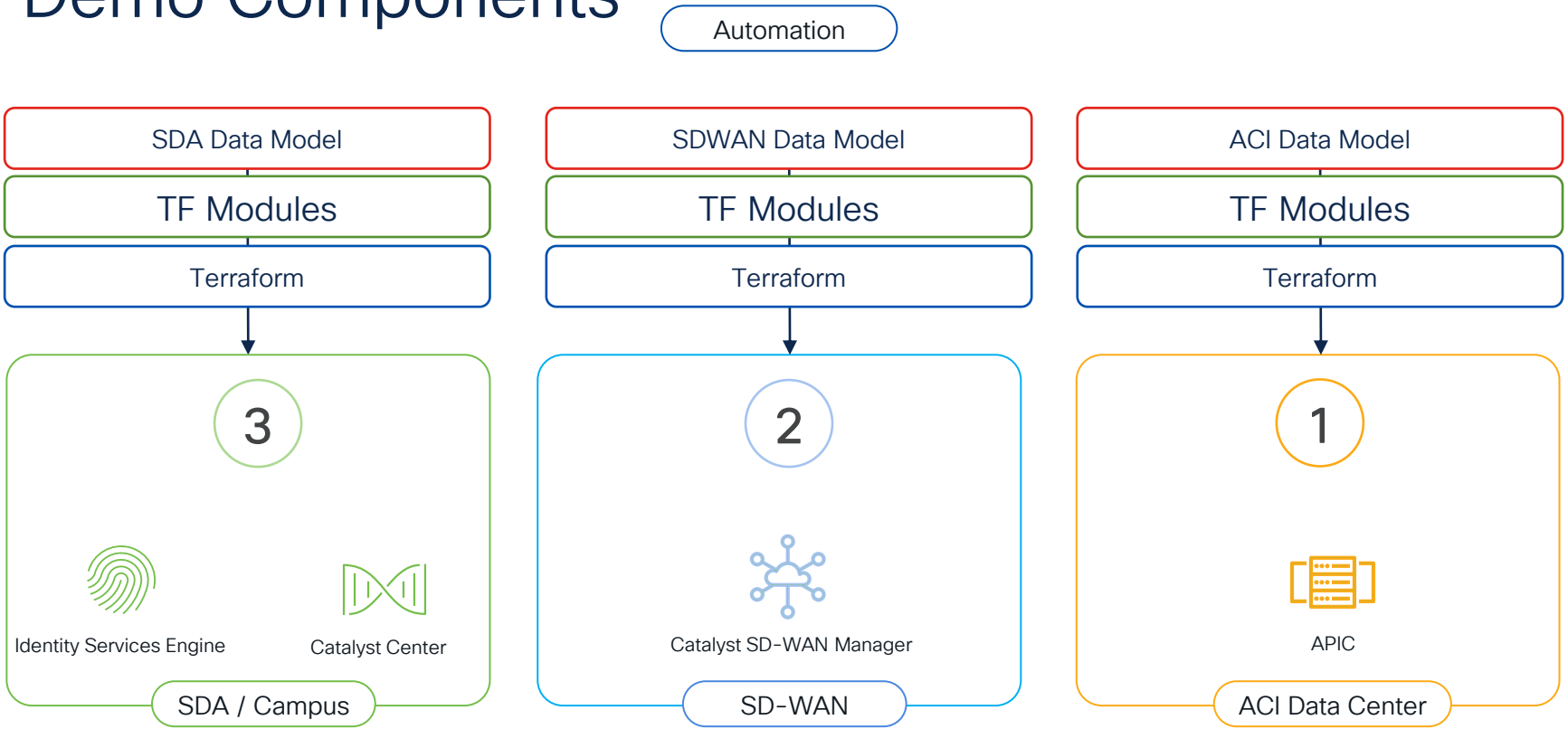
slido

Please download and install the Slido app on all computers you use

First run of Services as Code - whats your impression

ⓘ Start presenting to display the poll results on this slide.

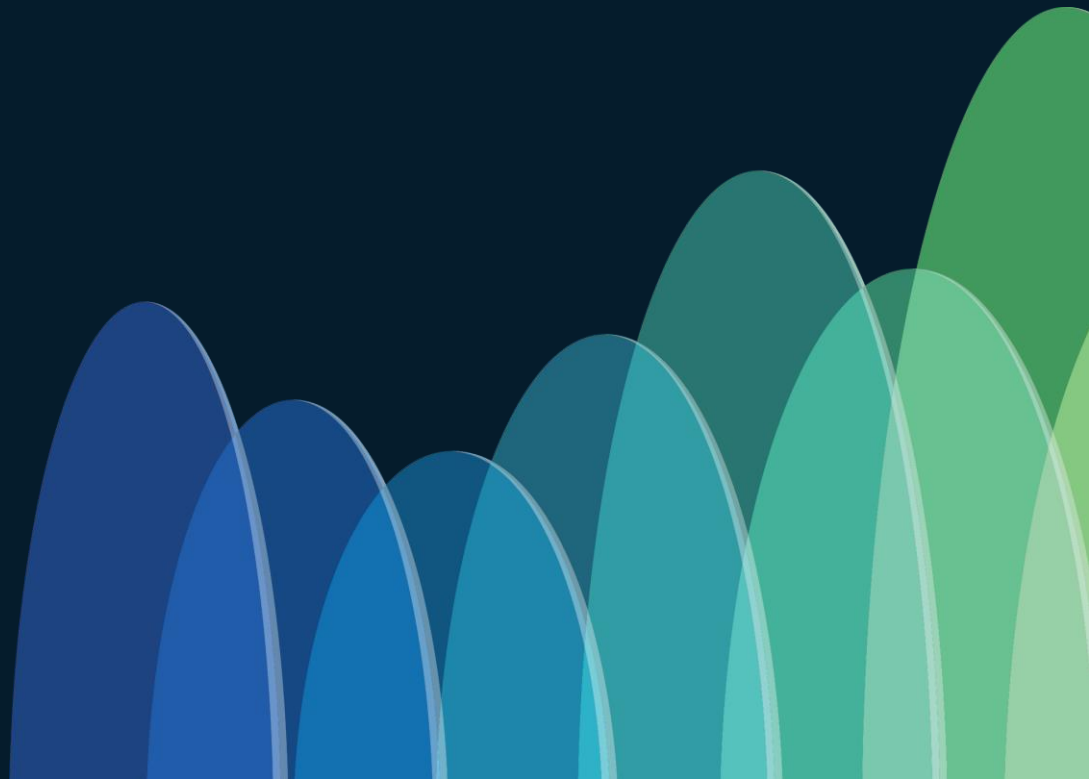
Demo Components



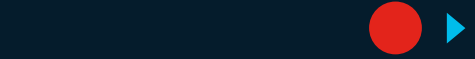
”Increase Efficiency with Service as Code for scalable, repeatable infrastructure automation."

Any idea ? Stay tuned – but its very Manual

What happened in the last minutes – technical intro



cisco Live!



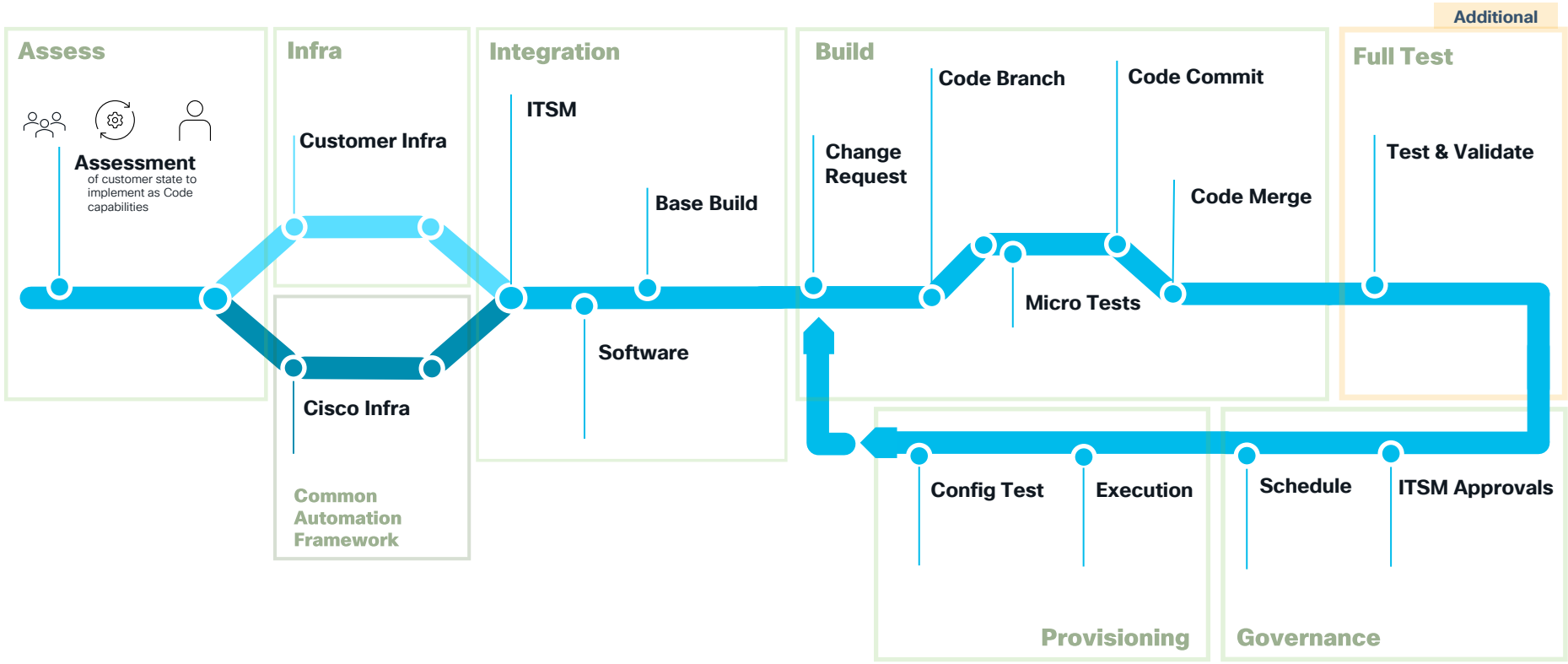
Services as Code Intro

IT is in a perfect storm

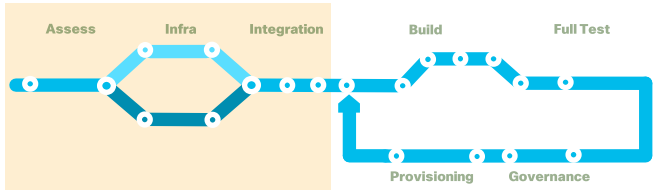
1. Networks are more complex than ever
2. Expectations from the network are higher than ever: performance, reliability, customer experience, provisioning time, scalability
3. Technology lifespans are shortening, creating increasing pressure to reduce OPEX to ensure ROI.
4. IT resources are more limited than ever. Most companies indicate that they'd prefer to reduce tech spending*.
5. Get AI ready !

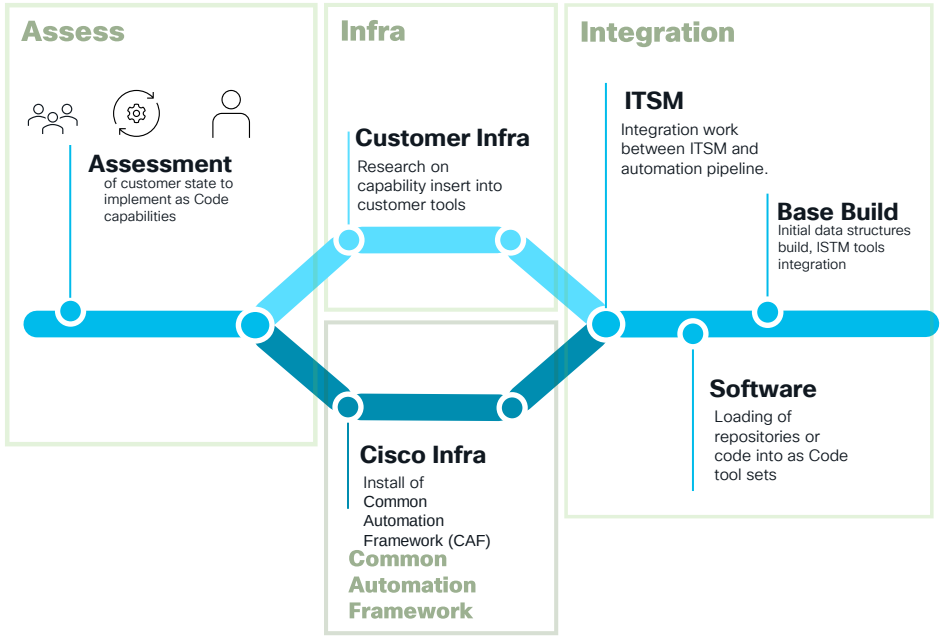
We need digitization & automation at unprecedented scale to navigate this perfect storm

Our service journey



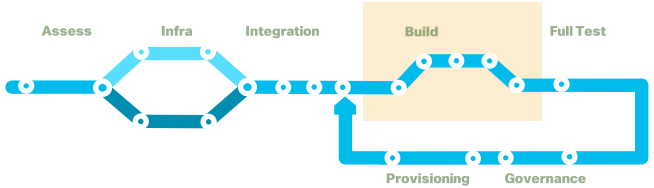
Initial Steps

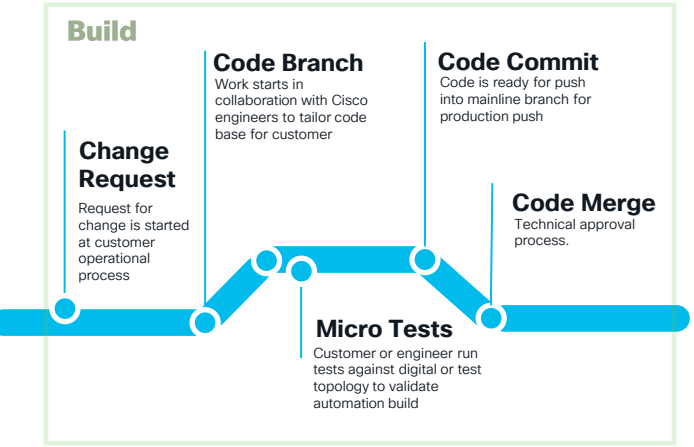




- Prepare the project
- Add automation engines as defined in first phase
- Integrate into Customer environment

The build process



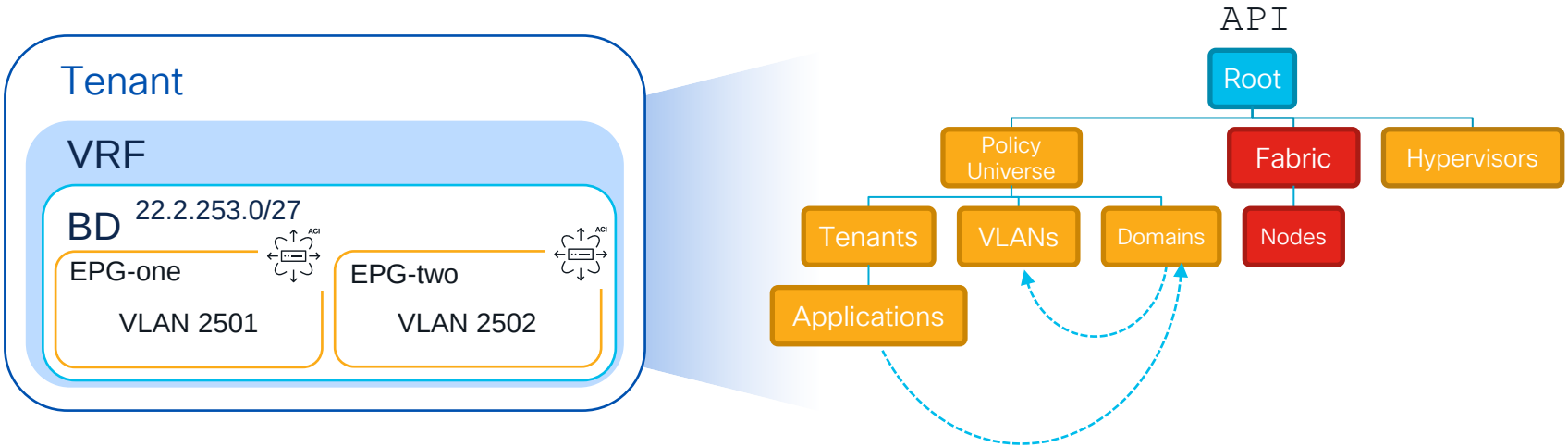


- Prepare change requests
- Have a Code Branch – Test
- Test the code model
- Prepare the data in the data model
- Built repos with YAML files
- Build tests and validate the data
- Potentially merge to other branches or production

Example ACI

The key element is Data and abstraction

- Every Controller or every device configuration follows a data structure
- Data needs to be human and machine readable



Code vs Data – why it simplifies

Native Terraform

```
resource "aci_tenant" "tenant_CiscoLive" {
  name = "CiscoLive"
}

variable "vrfs" {
  default = {
    VRF1 = {
      name = "VRF1"
    },
    VRF2 = {
      name = "PROD"
    }
  }
}

resource "aci_vrf" "vrfs" {
  for_each  = var.vrfs
  tenant_dn = aci_tenant.tenant_CiscoLive.id
  name      = each.value.name
}
```

ACI / Nexus-as-Code

```
apic:
  tenants:
    - name: CiscoLive
      vrfs:
        - name: VRF1
        - name: PROD
```

CX Data Model – example tenant config

CX Data Model

Product Configuration Model
API - native


```
{
    "totalCount": "1",
    "imdata": [
        {
            "fvTenant": {
                "attributes": {
                    "annotation": "",
                    "descr": "",
                    "dn": "uni/tn-CX",
                    "name": "CX",
                    "nameAlias": "",
                    "ownerKey": "",
                    "ownerTag": "",
                    "userdom": ":all:"
                },
                "children": [
                    {
                        "vnsSvcCont": {
                            "attributes": {
                                "annotation": "",
                                "userdom": ":all:"
                            }
                        }
                    },
                    {
                        "fvRsTenantMonPol": {
                            "attributes": {
                                "annotation": "",
                                "tnMonEPGPolName": "",
                                "userdom": ":all:"
                            }
                        }
                    },
                    {
                        "fvEpTags": {
                            "attributes": {
                                "annotation": "",
                                "userdom": ":all:"
                            }
                        }
                    },
                    {
                        "fvCtx": {
                            "attributes": {
                                "annotation": "",
                                "bdEnforcedEnable": "no",
                                "descr": "",
                                "ipDataPlaneLearning": "enabled",
                                "knwMcastAct": "permit",
                                "name": "VRF2",
                                "nameAlias": "",
...
```


+200 Lines of Configuration

TF-Provider

Infrastructure as Code Model
TF provider - native

```
resource "aci_tenant" "tenant_CX" {
  name = "CAB"
}

variable "vrfs" {
  default = {
    VRF1 = {
      name = "VRF1"
    },
    VRF2 = {
      name = "VRF2"
    }
  }
}

resource "aci_vrf" "vrfs" {
  for_each  = var.vrfs
  tenant_dn = aci_tenant.tenant_CX.id
  name      = each.value.name
}
```

20 Lines of Configuration

TF-Module

```
apic:
  tenants:
    - name: CX
      vrfs:
        - name: VRF1
        - name: VRF2
```

6 Lines of Data

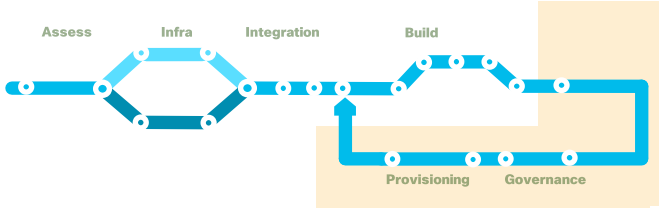
- Drives all pit stops as single “source of truth”
- Same look & feel across architectures
- Highly simplified and abstracted
- Best Practices baked in
- Possibility to expand to 3rd parties
- Usable via NAC-API in ITSM “no Code”
- Toolset independent “Open Source”
- passive part of Digital Twin

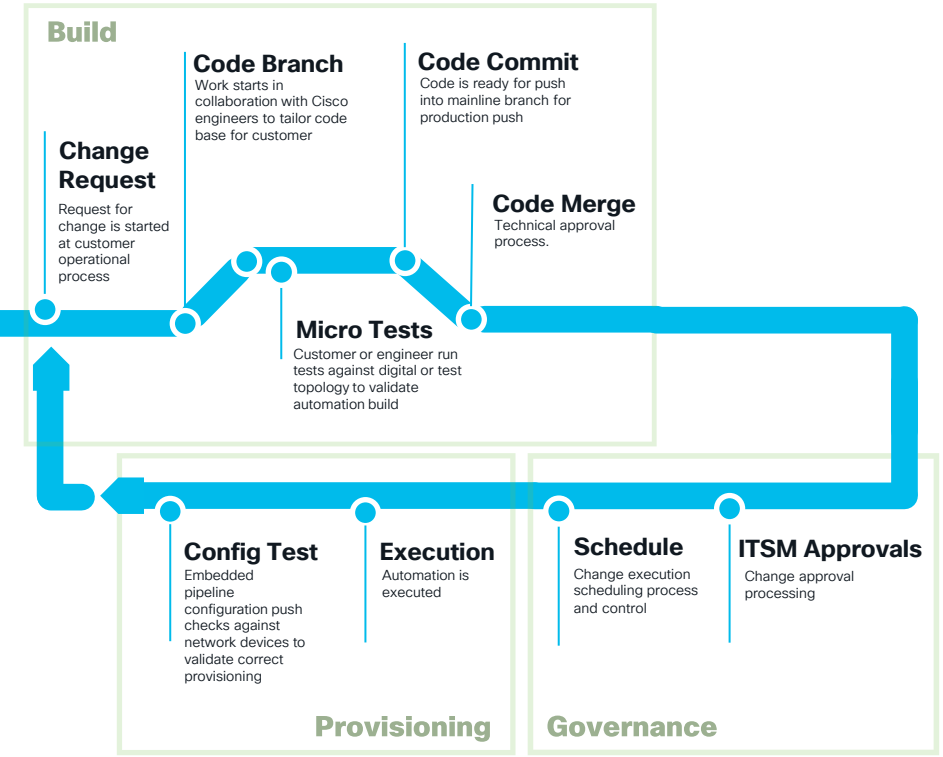
Include important settings by default's

```
---
defaults:
  apic:
    fabric_policies:
      ep_loop_protection:
        admin_state: disabled
        detection_intervall: 60
        detection_multiplier: 4
        action: bd-learn-disable
    tenants:
      vrfs:
        name_suffix: '_VRF'
        data_plane_learning: enabled
        enforcement_direction: ingress
        enforcement_preference: enforced
```

- Lessons learned
- CVD and best practice
- Customer requirements
- Eliminate
- Standardize
- Simplify cases

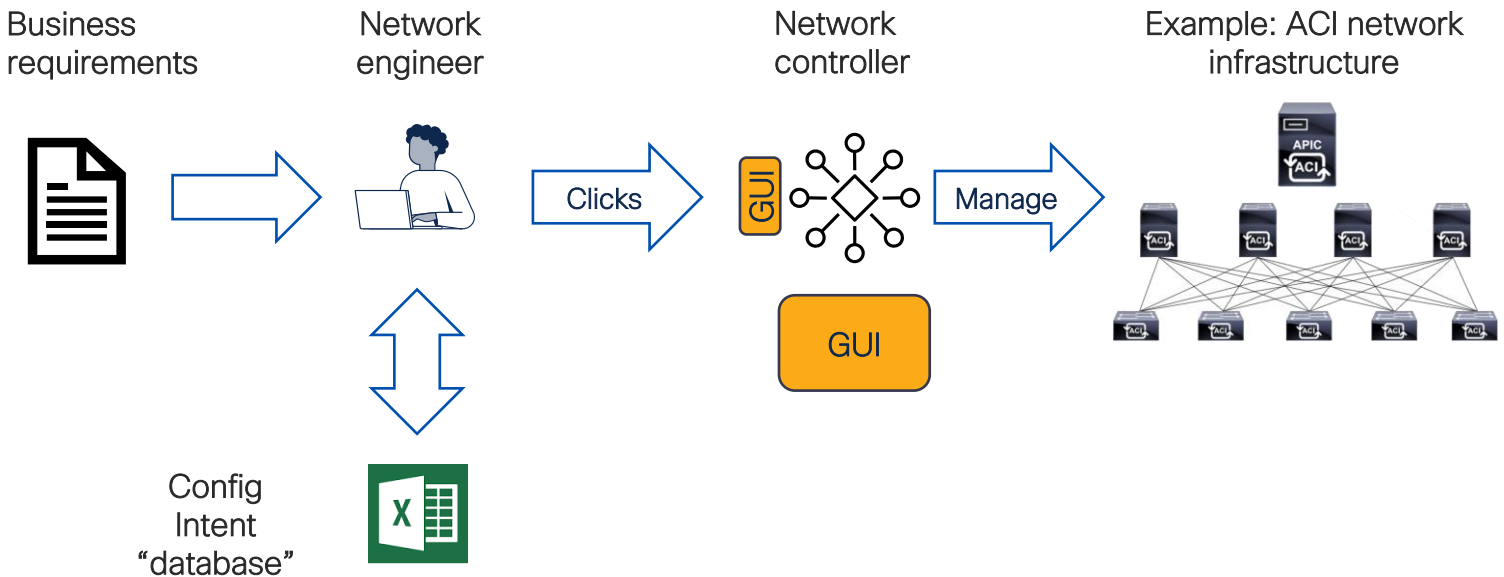
The build process – production



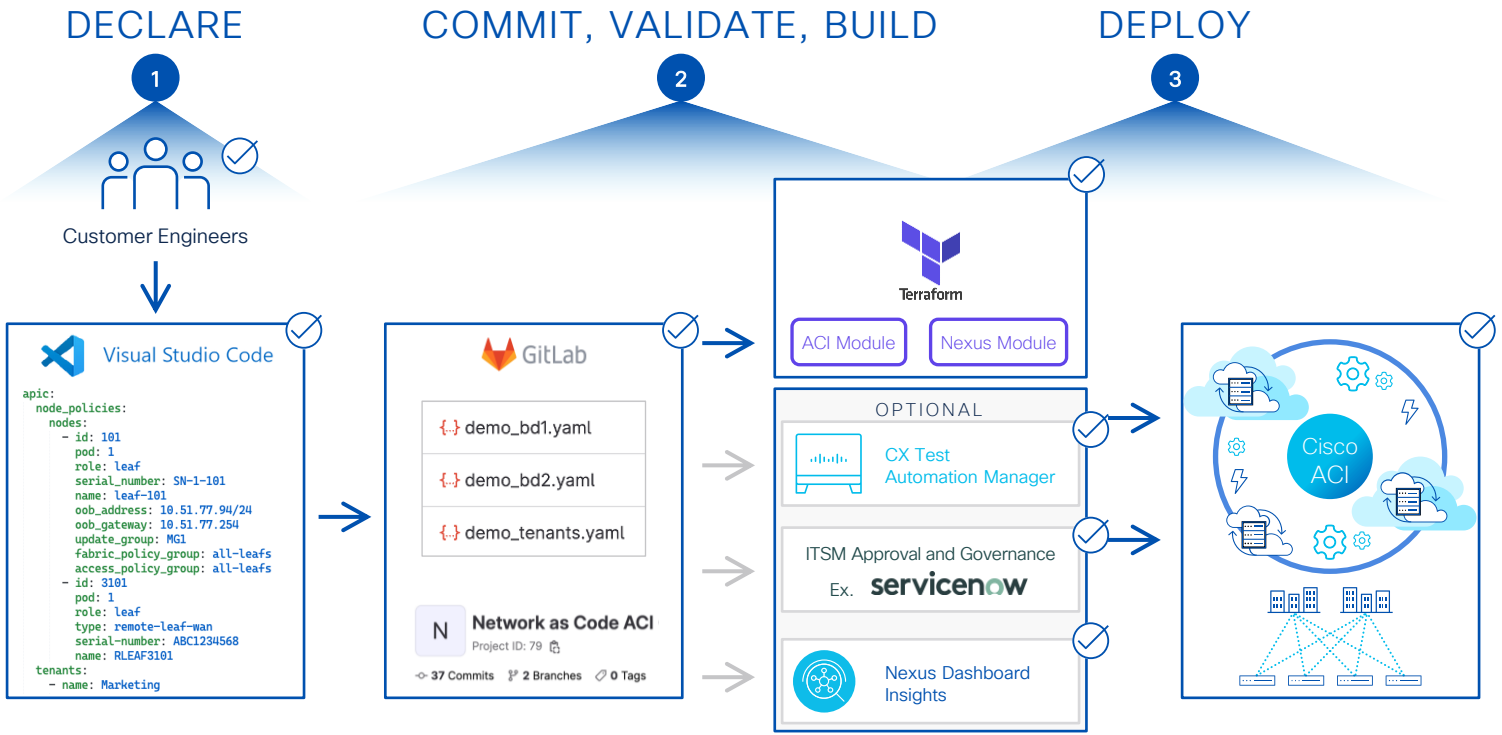


- Merge the data to production
- Execute the change
- Validate the result
- Continue

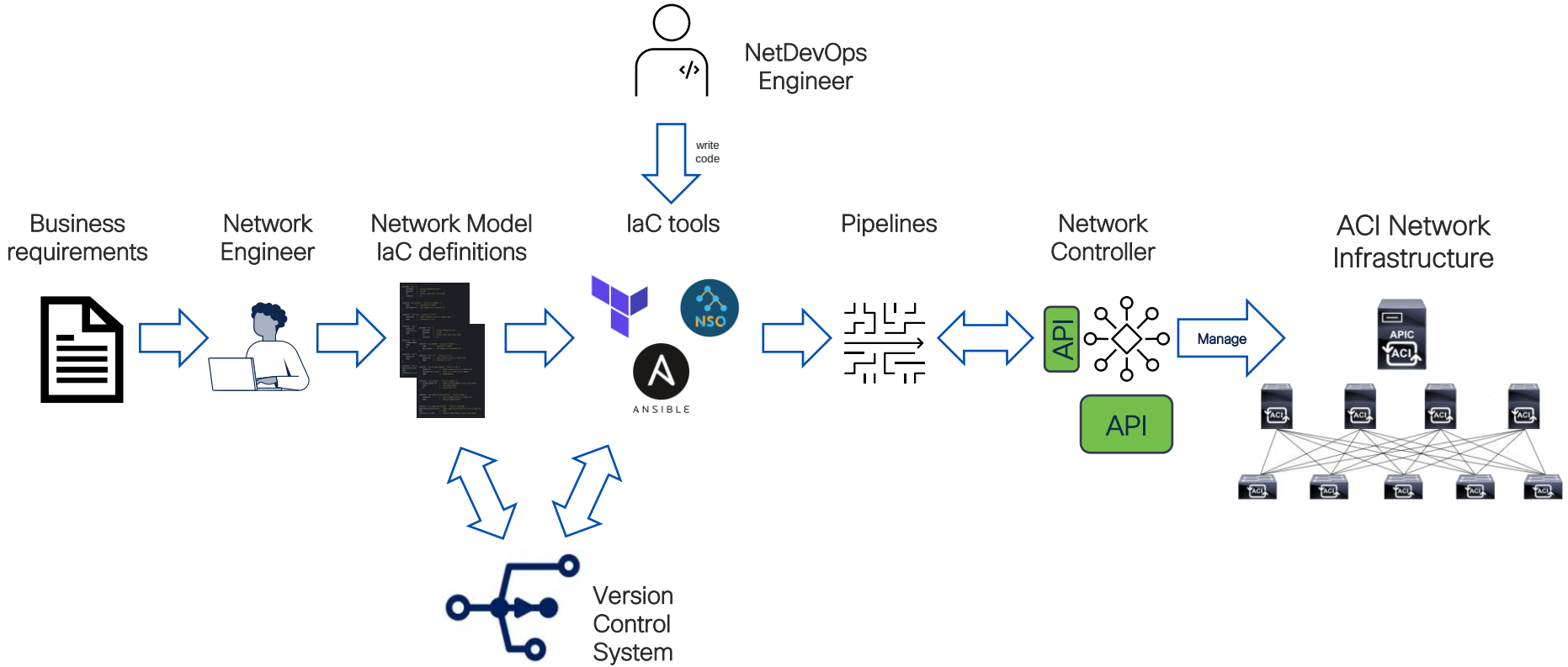
I am the network engineer / operator / admin…



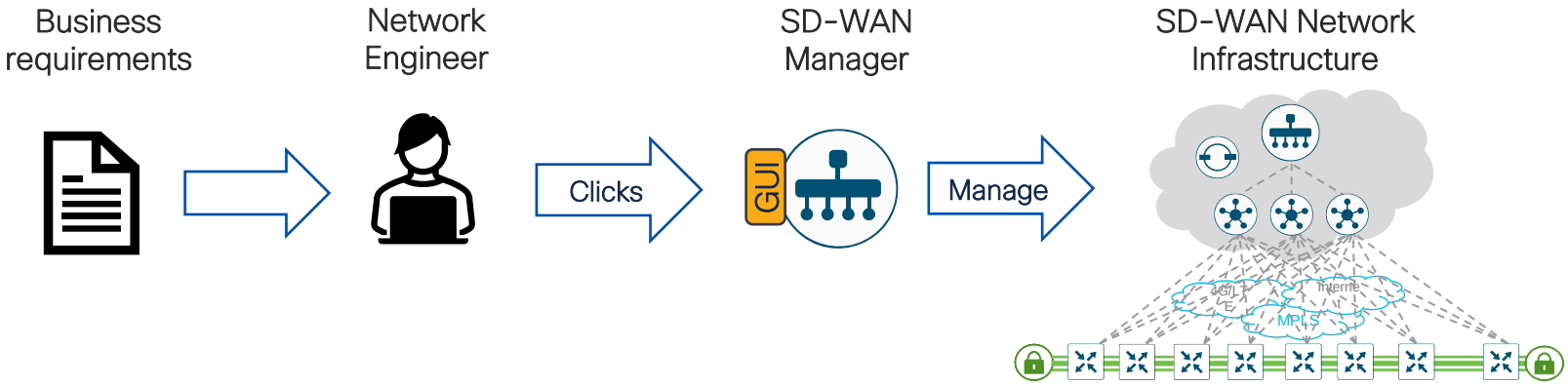
An Example: Services as Code for ACI flow option



Model Driven – Infrastructure as Code



“SDWAN Click” Operations – is it the same?



Why buy Cisco Service as Code?

Faster, cheaper Time-to-Value

Lower the time and cost to deliver operational automation for your ACI, SDWAN, FW or ISE deployments by using Cisco's SaC pre-built, tested and validated automation examples saving at least 6 months development costs

Built using best practise

Built with DevOps methodology such as integration with (CI/CD) pipelines improving reliability and quality of delivery

Evergreen supported lifecycle

Cisco provide support of the solution and maintain SaC framework against new versions of software and hardware saving customers time and effort and reducing risk

Compliant out of the box

Automation files are version controlled, allowing change tracking, roll-back and lifecycle management.

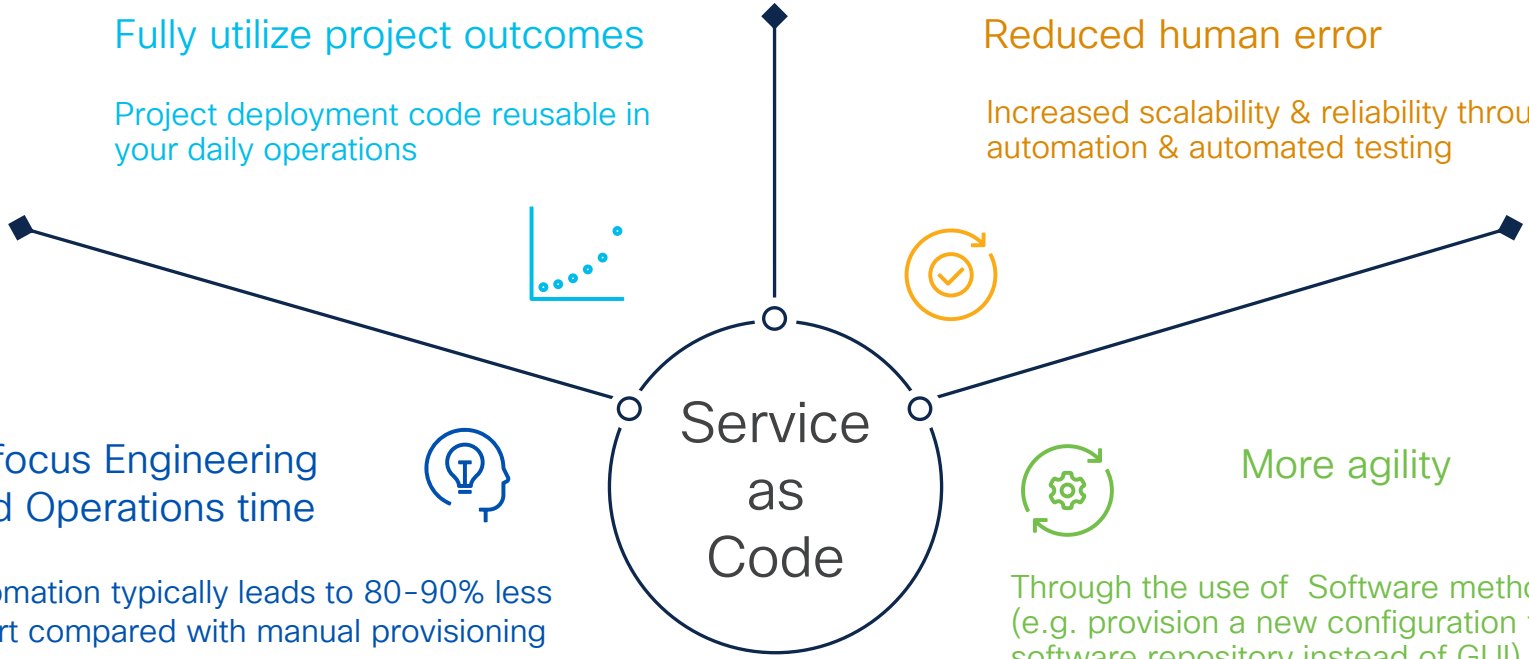
Scalable and proven

Solution scales to very large infrastructure bases. Referenceable delivery at scale. No subscription pricing means costs are contained as solution grows.

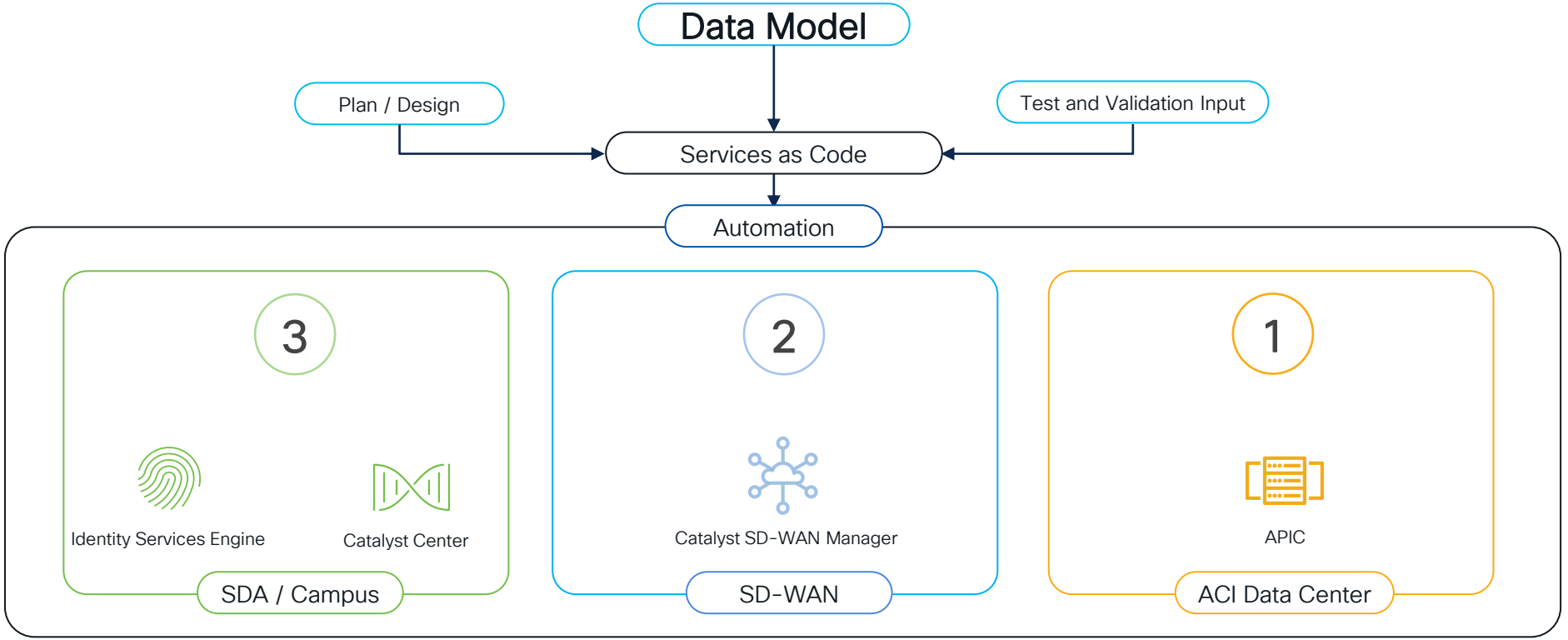
Open and interoperable

Built on industry standard tools to ensure interoperability with customer's existing tool sets and processes reducing need for complex integration

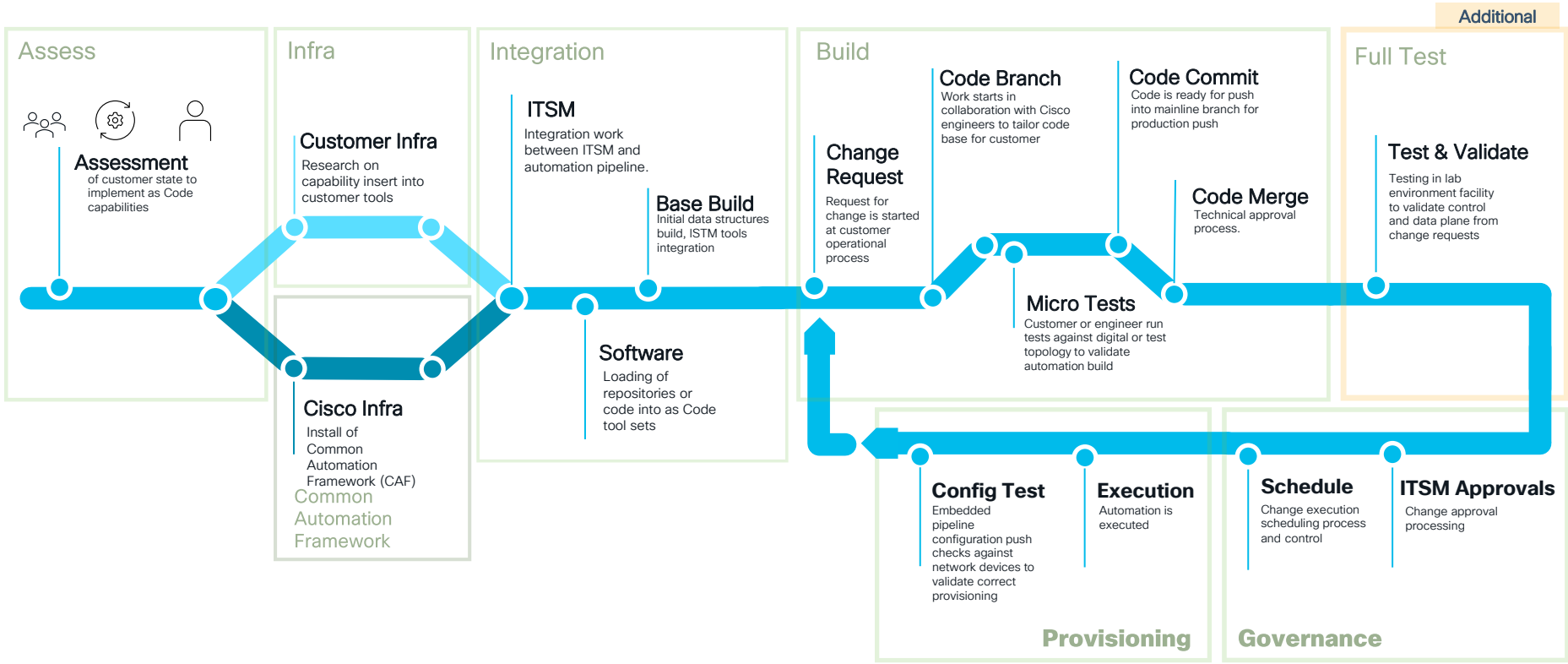
Value with Services as Code



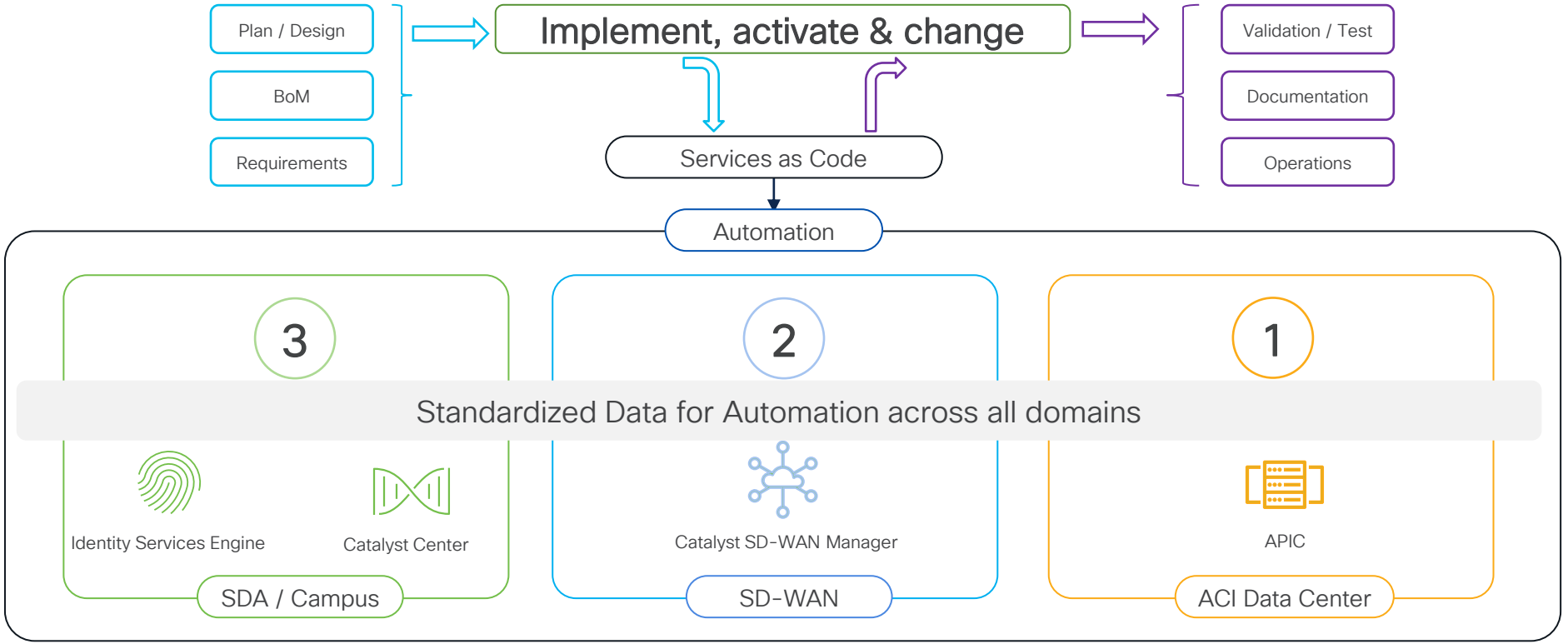
Back to the concept



High level service architecture



LAB / Demo Architecture view



Traditional work of a network admin

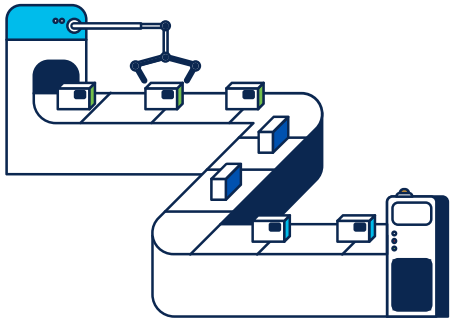
My tasks

- Design the network
- Ensure the network is working
- Plan, process and document changes
- Change VLANs, ports, upgrades etc.
- Be the SPOC for all issues
- Know CLI very well and all the features
- Today – understand UI workflows and how to use the UI, does not matter
- ...

Network operator
Network admin
CLI junkie

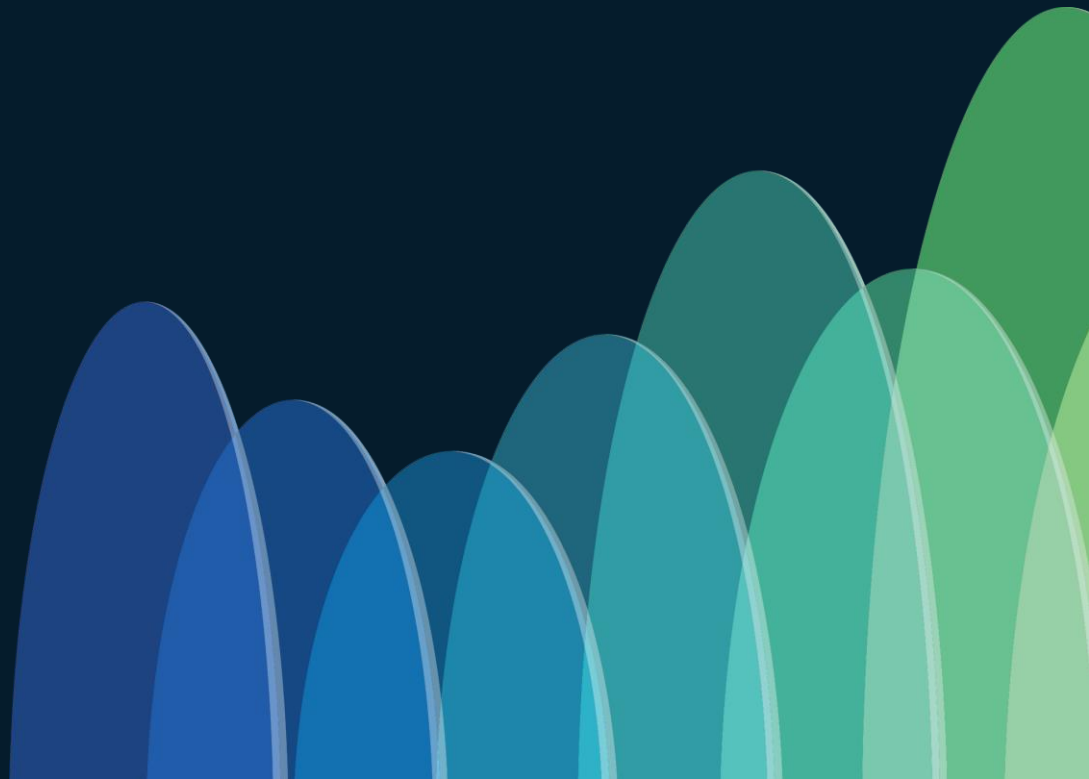
Do everything!

The tasks evolved



- Ensure 100% up time
- Enable automation
- No errors before during and after changes
- CICD
  (Continuous Integration / Continuous Development)
- Plan seamless Roll back
- Ah and 100% uptime

Automation



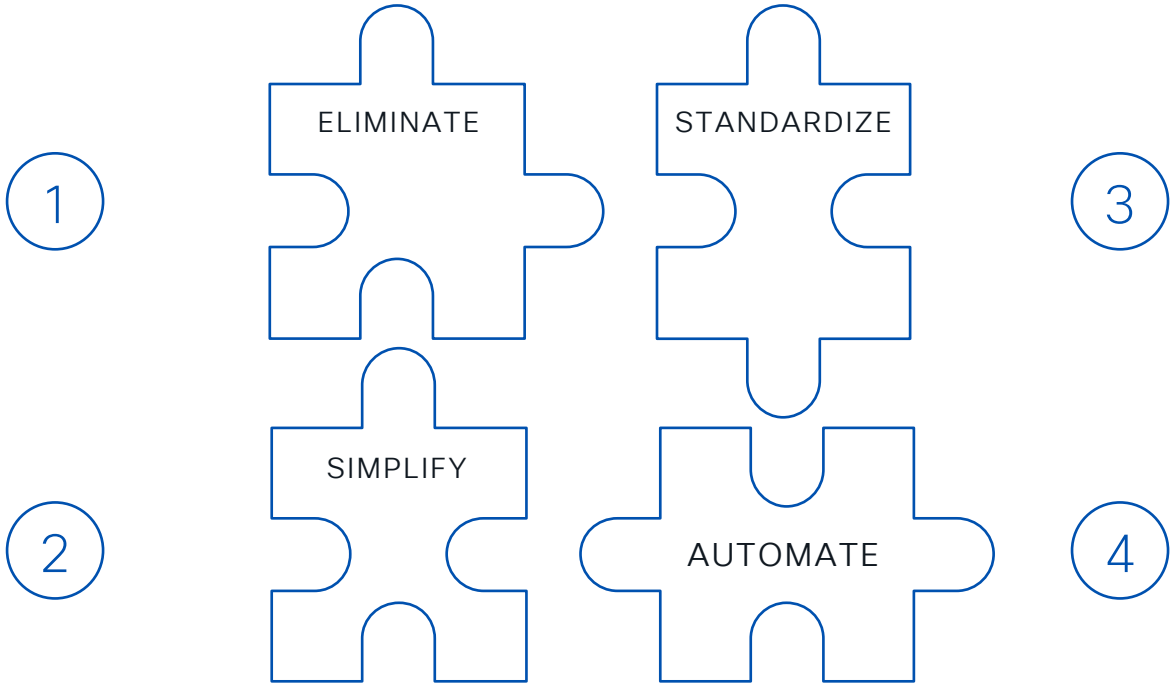
slido

Please download and install the Slido app on all computers you use

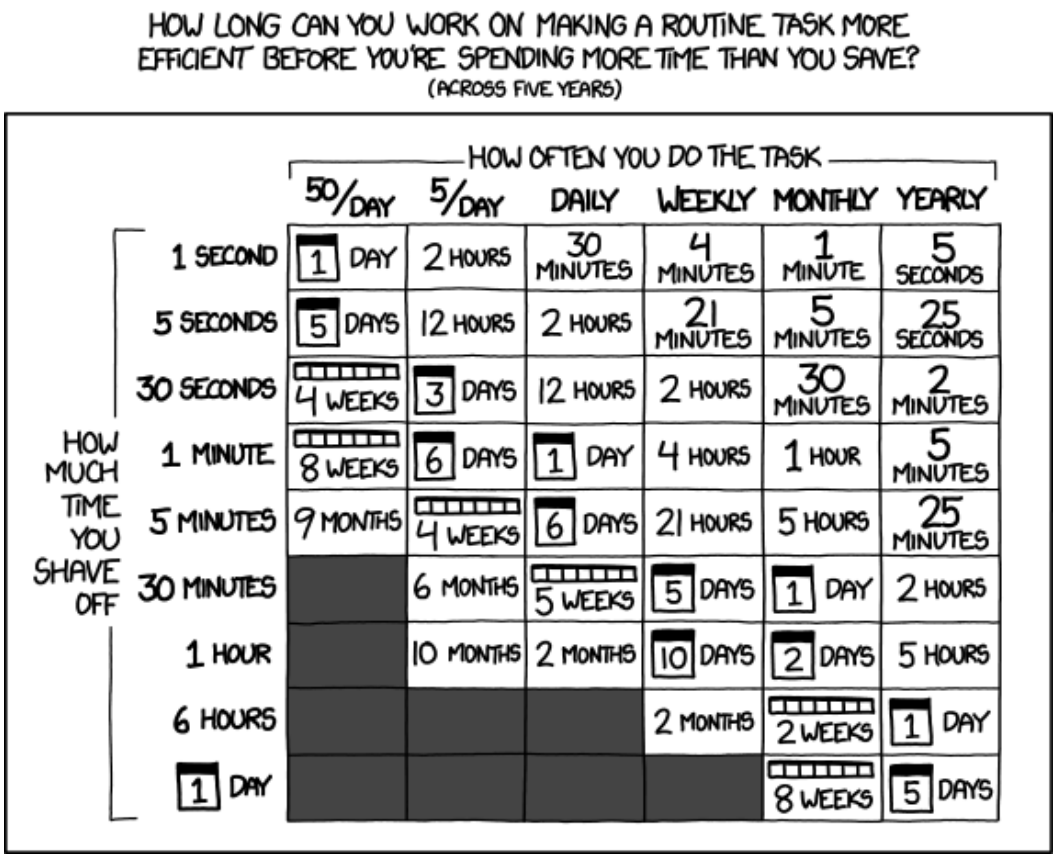
Do you remember this?

ⓘ Start presenting to display the poll results on this slide.

Services as Code streamlines operation and enhanced business efficiency.



Automation and why?



Source: https://xkcd.com/1205/

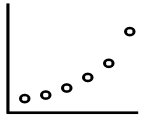
Why Automation ?

- Number of device increase is higher than growth of staff
- Increase of solutions capabilities result in increase of complexity
- Question of math, more manual changes result in more failures
- Waste of time by doing boring tasks

Main challenges automation can solve

- Number of device increase is higher than growth of staff
  - Automation will become mandatory to grow in the future
- Increase of solutions capabilities result in increase of complexity
- Question of statistics, more manual changes result in more failures
  - Human failures are about 80% the root cause of problems in changes
- Waste of time by doing boring tasks
  - Helps to focus on interesting and complicate tasks.

Considerations for automation projects

Learning Curve 

>50% 

Considerations for automation projects

- Automation require a learning curve
  - Benefit will not come full at day 1
- Automate 100%? It may never happened
  - But 50% will be already great improvement
  - Increase it from there by time
  - There will be always the "special case", but limit those!

How to start with "blue-prints"

Usual Suspects

Data -> YAML

New CLI

Service Use Cases

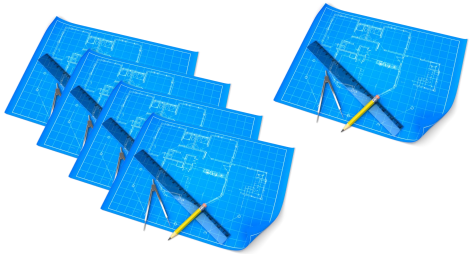
How to start with "blue-prints"

- Collect your "usual setups"
- Translate this into data for automation
  - Describe the capabilities from “network” and “service” view
  - Define “service use cases”
- Transform this data in YAML
  - This will be your new "CLI"

What if new requests didn't fit the blue-prints?

“Play it again Sam...”?

Sequence



Same for all domains

What if new requests didn't fit the blue-prints?

- Could service request come again?
- Sequence for automation:
  - Test and optimize manual via GUI or CLI
  - Design should be done with automation in mind
- Create new blue-print
  - Simplify / generalize the data
  - Describe blue-print from network and service point of view!
- You have increased your portfolio!
- Same method for ACI, SDA as well SD-WAN!

How to enforce your clients using automation?

Cost

Delivery Time

How to enforce your clients using automation?

- Different cost for services been automated versus "induvidual manual handcrafted and signed"
  - You think didn't work, all cloud and service provider are doing this
  - Might not be possible inside your company
- Different delivery time for services been automated versus "individual manual handcrafted and signed"
  - Time is usually key in projects... if handcrafted services needs 10 times longer, maybe the internal clients will think twice.

What should be automated?

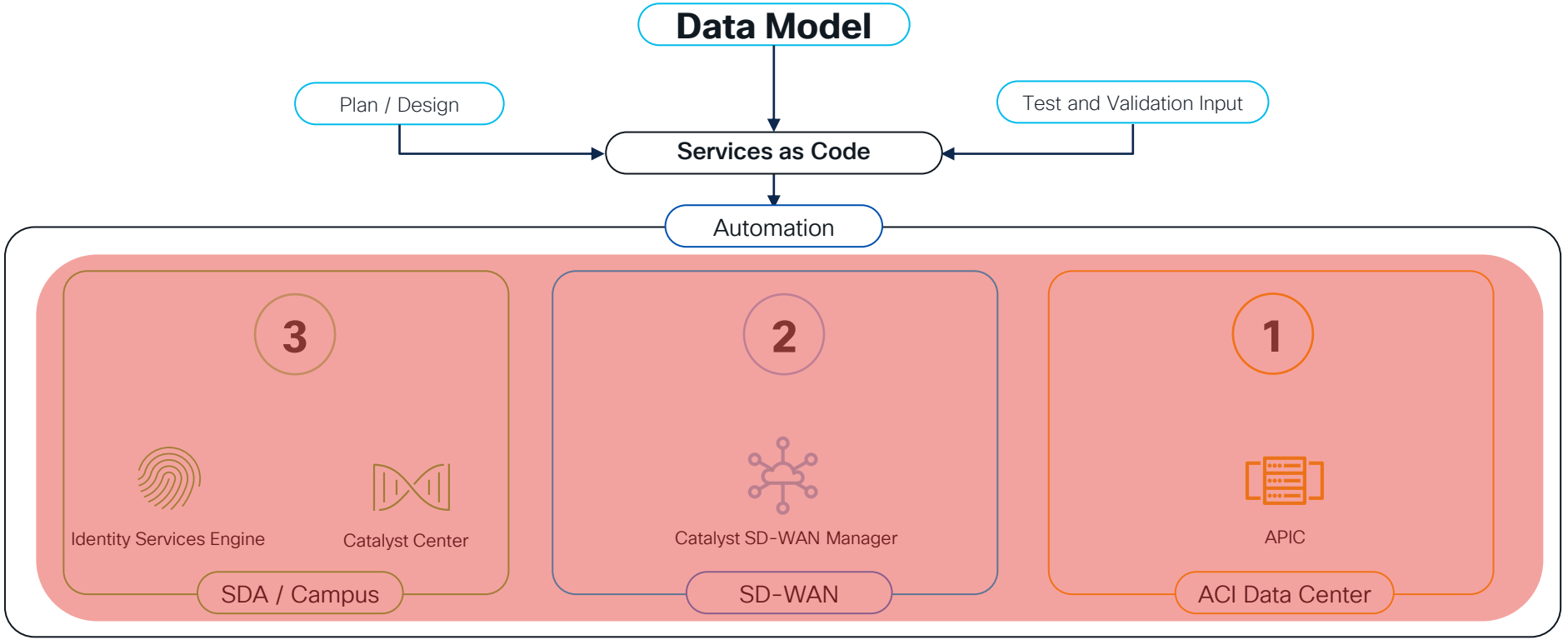
Start Easy

Rule-Of-Thumb

What should be automated?

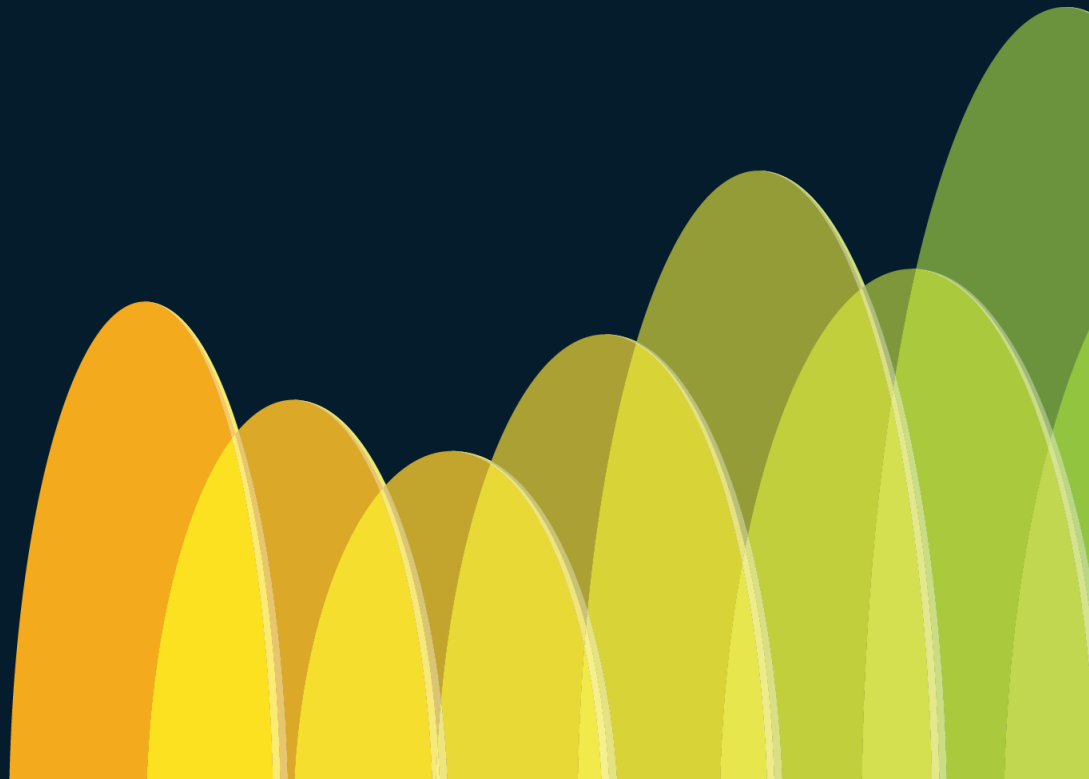
- “Give me the red easy button for all”
  - No good idea … 99% chance of disaster
- Rule of thumb:
  - What you are doing often and has low level of variants is good to automate
  - If you are doing something few times a year, don't spend the time … one reason for our decision for the baseline of this technical seminar
  - Automate what is boring … more likely the automation will be used, and you will see an effect of the automation
- Start with easy example (low hanging fruit)
  - During first automation you will mostly fight with the automation framework and process, make your life easier if the technical level is low

Back to the concept and yes, we destroy now



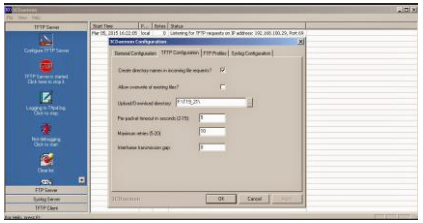
Demo - destroy

Challenges and optimization



NetOps Tools

WIRESHARK



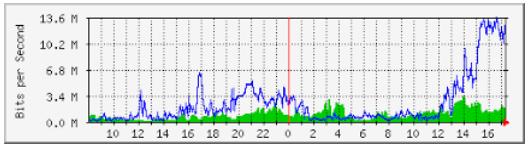
3com FTP Server

```
(base) maharbec@MAHARBEC-M-92HK .ssh % ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8): 56 data bytes
64 bytes from 8.8.8.8: icmp_seq=0 ttl=117 time=65.238 ms
64 bytes from 8.8.8.8: icmp_seq=1 ttl=117 time=10.076 ms
^C
--- 8.8.8.8 ping statistics ---
2 packets transmitted, 2 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 10.076/37.657/65.238/27.581 ms
(base) maharbec@MAHARBEC-M-92HK .ssh %
```

Ping

```
(base) maharbec@MAHARBEC-M-92HK .ssh % traceroute 8.8.8.8
traceroute to 8.8.8.8 (8.8.8.8), 64 hops max, 52 byte packets
 1  fritz.box (192.168.155.1)  71.241 ms  0.592 ms  0.301 ms
 2  p3e9bf3d7.dip0.t-ipconnect.de (62.155.243.215)  26.740 ms  4.764 ms  4.460 ms
 3  m-ef2-i.m.de.net.dtag.de (217.0.194.110)  31.444 ms  11.650 ms  11.049 ms
 4  m-ef2-i.m.de.net.dtag.de (217.0.194.110)  16.886 ms  10.678 ms  10.597 ms
 5  80.157.205.162 (80.157.205.162)  49.431 ms  11.656 ms  11.223 ms
^C
```

Traceroute



MRTG

...Simple ?

DevOps tools periodic table

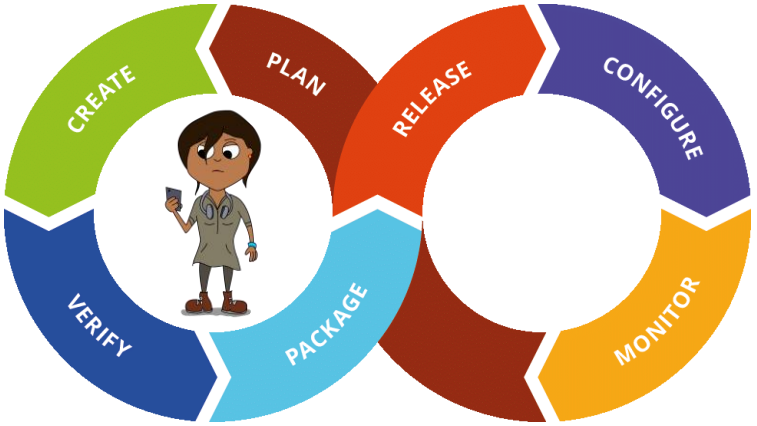
...Simpler ?

Source: https://xebialabs.com/periodic-table-of-devops-tools/
https://digital.ai/learn/devops-periodic-table/

Different mindsets

Change management mindset

Avoid failure, change is risky and complex, empowered accountability, limited feedback systems, manual



DevOps Mindset

Embrace failure, change is good, active collaboration, empowered accountability, feedback systems, automation

What are your top challenges?

My network as designed doesn't match reality and all networks look different

Network changes consume too many resources with too many errors

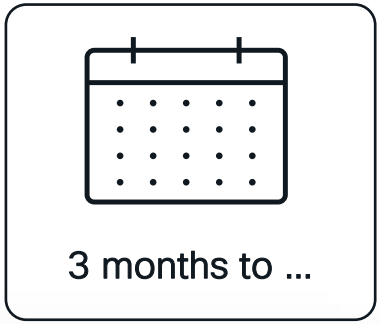
I spend more of my time reacting to network needs

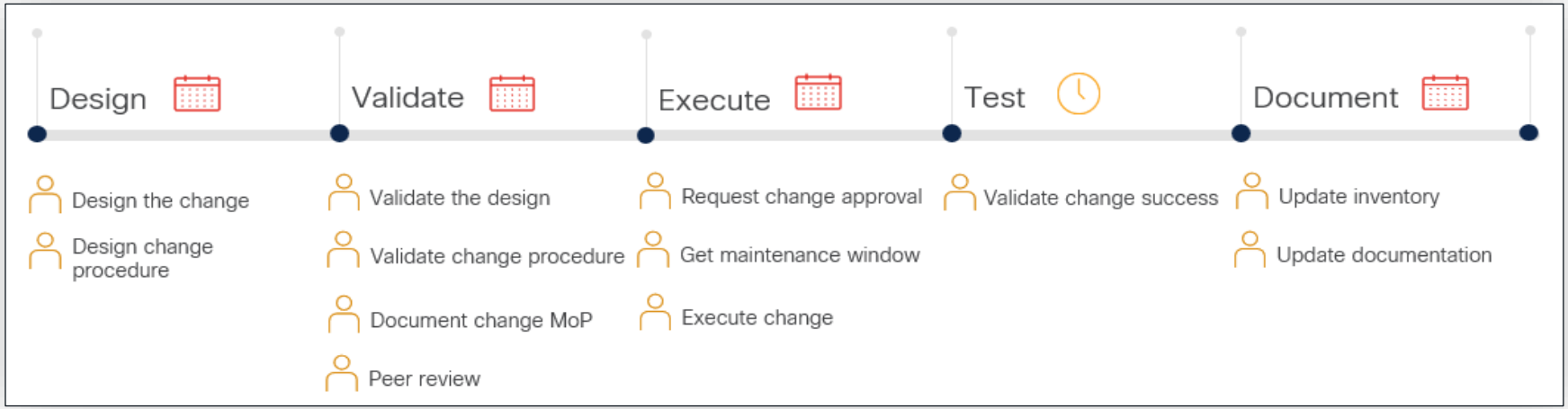
I’m questioning network and security compliance

I’m challenged to apply automation

Skill shortages and low innovation

Manual Implementation & Operation

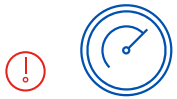




Network operations challenges

Lack of version control

- Difficult to track changes
- Potential configuration drifts
- Inconsistency
- Hard to audit and roll back
- Difficult troubleshooting



Lack of automation

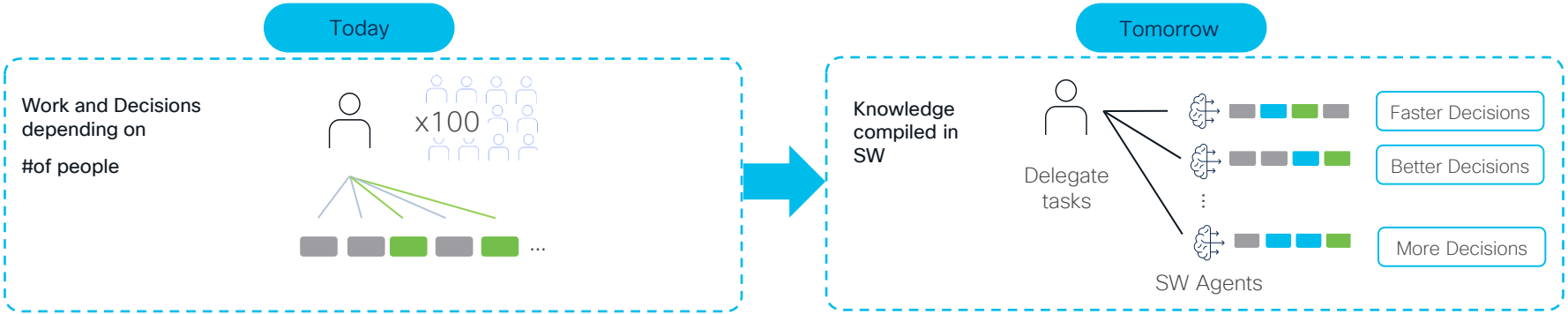
- Each deployment or provisioning becomes a unique manual process
- Automation is complex
- Each deployment or provisioning becomes a unique manual process

Lack of pre-production testing and validation

- changes made to the infrastructure and services are not thoroughly tested before being deployed in a production
- High risk off issues
- Unexpected downtimes

Manual execution does not meet market needs

The increasing complexity of IT infrastructures demands making many more decisions… faster and better



It’s essential to hand tasks structurally over to SW to execute, it is preferable way to meet the needs for growth and complexity of IT infrastructure!

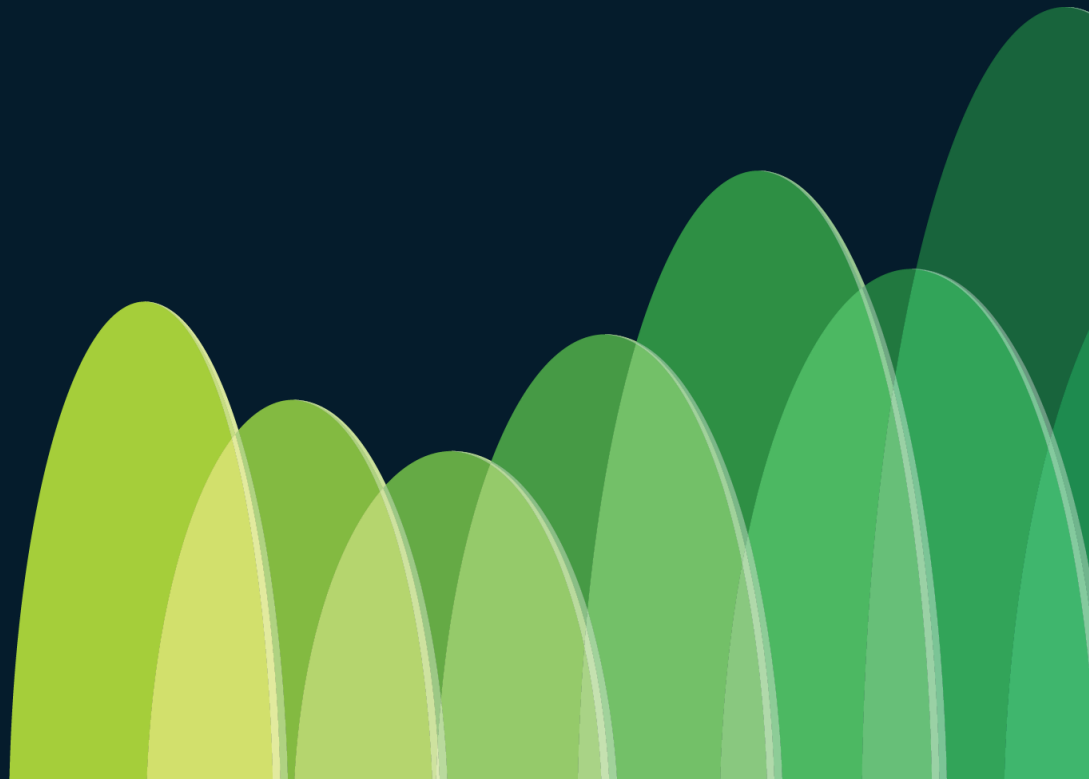
slido

Please download and install the Slido app on all computers you use

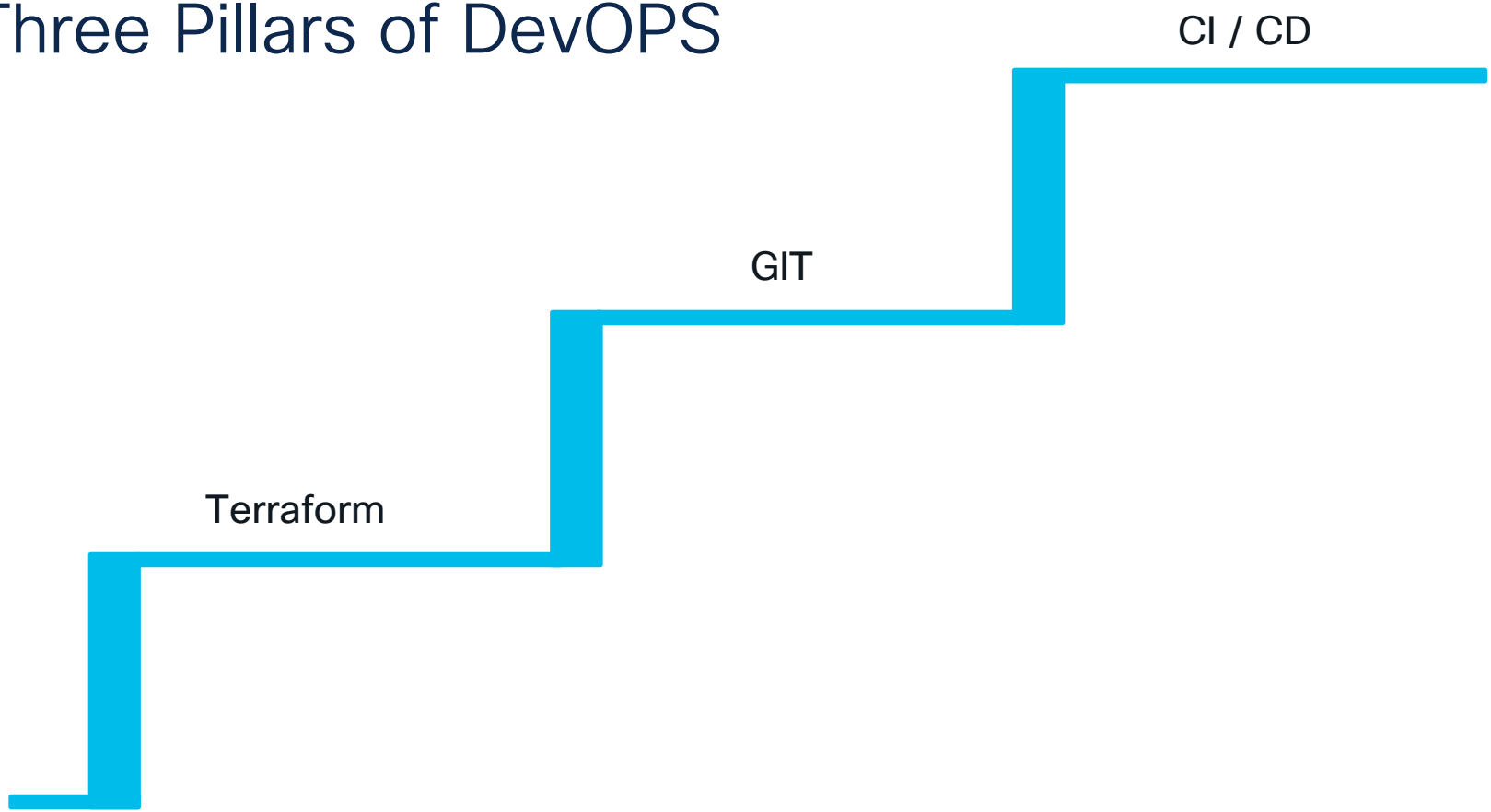
Its a network or?

ⓘ Start presenting to display the poll results on this slide.

Some DevOPS



Three Pillars of DevOPS



Declarative vs Imperative

- Define ***what*** the eventual target configuration should be
  - e.g., 1 Tenant with 2 BDs and 2 EPGs
- Define the desired state
- Automation is responsible for the desired state to be reflected in the infrastructure

- Define ***how*** the infrastructure should be changed
  - e.g., Add BD X and EPG Y
- Automate a common use case (e.g. add a network segment)
- Automation defines steps (workflow) to end with the desired conclusion

What is Ansible?

- Open-source Configuration Management Tool
- Commercial support from RedHat
- Focus is Imperative
- Declarative (when possible) and idempotent
- Can manage a wide range of systems:
  - VMs, network devices, cloud instances, etc.
- Agentless
- Python server-side dependencies

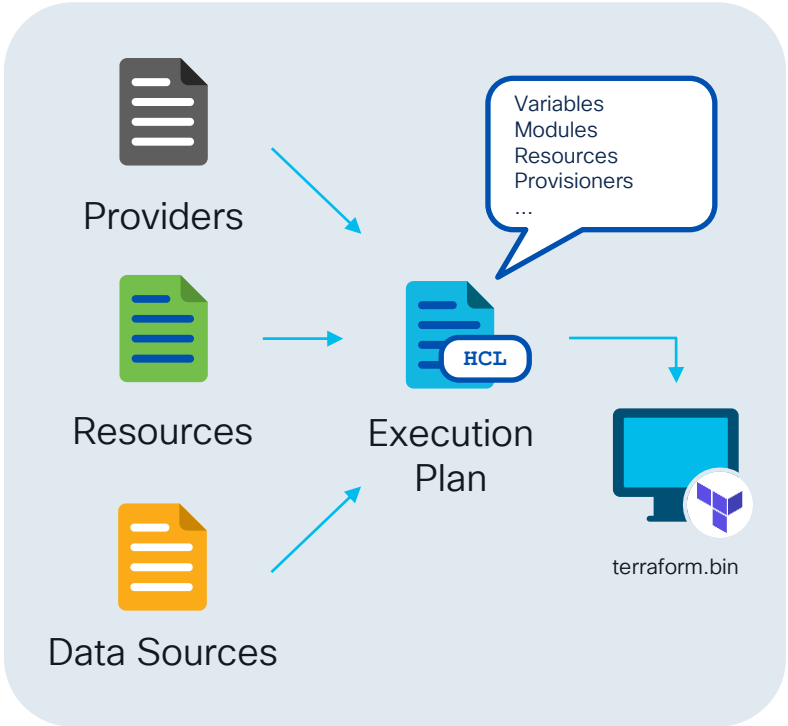
What is Terraform?

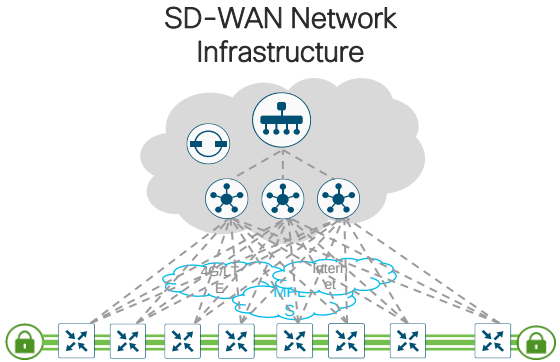
- Open-source Infrastructure Provisioning Tool
- Commercial support from HashiCorp
- Declarative and idempotent
- Immutable infrastructure concept
- Can manage a wide range of systems:
  VMs, network devices, cloud instances, etc.
- Agentless, single binary file
- Zero server-side dependencies

Ansible or Terraform?

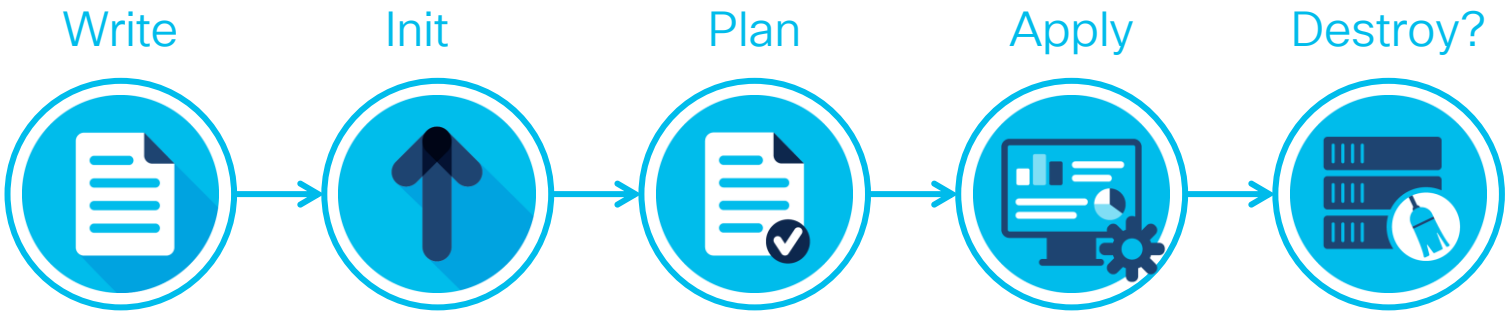
- Both Ansible and Terraform can coexist
  - It's not an either/or story
  - Terraform can call Ansible for ad-hoc tasks after deploying a VM
- Terraform keeps state locally
  - It knows what is configured vs desired end-state
  - Can automatically destroy / recreate resources
- Ansible mutate the infrastructure
  - Need to re-run everything
  - Might need to create advanced controls to avoid long running scripts
- For Service-as-Code Terraform is the best match

Terraform Building Blocks





Terraform Workflow Stages



Write your code

Download and initialize environment
(providers, modules, backend, ...)

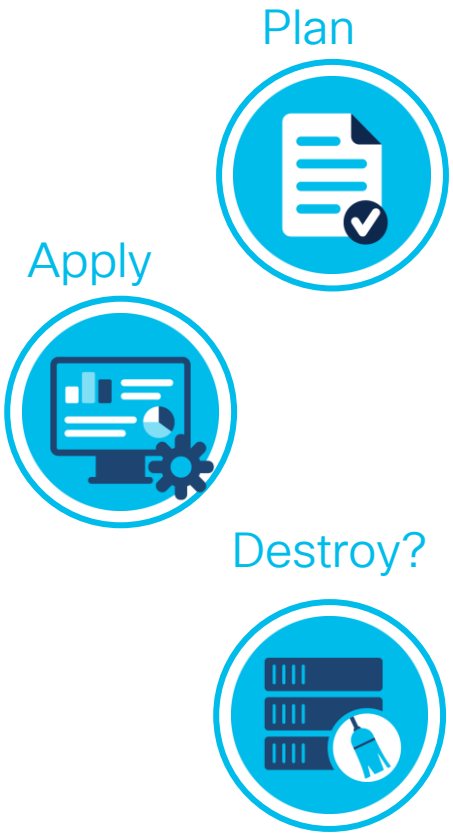
Generate "terraform-statefile"
by calculate required changes

Apply changes into
real infrastructure

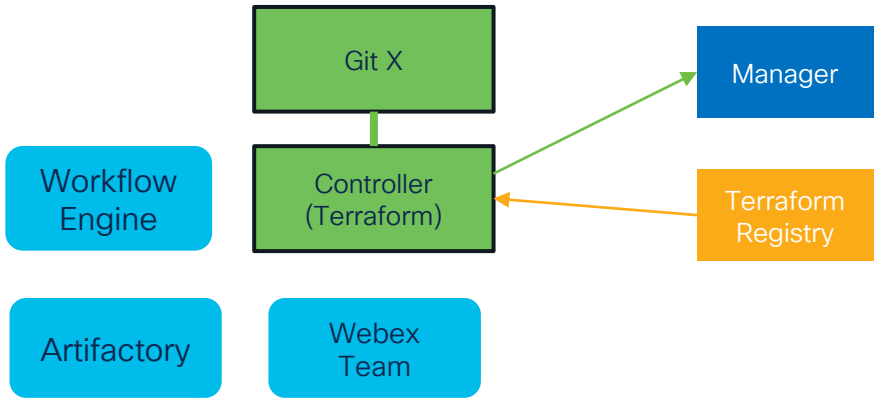
Remove all resources

Why Terrafrom Statefile is Important

- Stage "Plan" generate the "terraform state" file
- Calculate changes based on:
  - YAML file changes
  - Device changes
- Benefit: Allow dry run of changes
- Stage "Apply" is roll out "terraform state" objects
- Stage "Destroy" is delete "terraform state" objects



Terraform AAC – Execution Components



Controller
Host/agent running the Terraform script

Git X
Version Control System like GitHub, GitLab, BitBucket

Workflow Engine
GitLab CI, Drone, Jenkins
Note that Terraform script can also be played without Workflow Engine

Terraform Registry
Storage for Terraform Provider and Modules

Artifactory
General Storage for test results, logs
Workflow engines have their own archive that can be used on this purpose

Webex Team (Optional)
For Notification of Execution Success/Failure

GIT basics

Source: https://xkcd.com/1205/

GIT basics

- GIT is not an acronym
- GIT is a "Version Control System"
- GIT can be used for source code, single files, or many YAML files

```
joreinec@JOREINEC-M-6P24 0_Git % ls -l
total 984
-rw-r--r--@ 1 joreinec  staff   70780 Nov 20 11:40 AAC commands.txt
-rw-r--r--@ 1 joreinec  staff   23759 Nov  1 16:57 ACI commands.txt
-rw-r--r--@ 1 joreinec  staff   10724 Jul 19  2023 GIT commands.txt
-rw-r--r--@ 1 joreinec  staff   43829 Oct 14 11:05 Linux and Netconf commands.txt
-rw-r--r--@ 1 joreinec  staff   18263 Oct  9 22:38 Praxis copy and paste.txt
-rw-r--r--@ 1 joreinec  staff  188283 Nov 12 19:05 SDA config template show commands.txt
-rw-r--r--@ 1 joreinec  staff   19328 Nov 19 18:05 acronyms.txt
-rw-r--r--@ 1 joreinec  staff   83132 Nov 11 13:09 copy and paste Master.txt
-rw-r--r--@ 1 joreinec  staff   27184 Oct 24 12:09 docker and k8s commands.txt
-rw-r--r--@ 1 joreinec  staff    3280 Nov 16  2023 pyATS commds.txt
joreinec@JOREINEC-M-6P24 0_Git % _
```

```
joreinec@JOREINEC-M-6P24 0_Git % git log
commit 11b7e89a54f2ec6afc741e79dda9363f6a6f17a1 (HEAD -> joreinec-master)
Author: joreinec <joreinec@cisco.com>
Date:   Wed Nov 20 15:24:53 2024 +0100

    Added to AAC comment about gitlab support for terraform

commit c6dbacbc412679436710157ce20e997e2d5441ad
Author: joreinec <joreinec@cisco.com>
Date:   Tue Nov 19 18:05:32 2024 +0100

    Added to acronyms meaning of GIT

commit e8b0f1fc854d5de66a40a4124413a2459f4ff2b7
```

GIT basics – GIT acronyms and few commands

Local repo

branch

main branch / master branch

git checkout

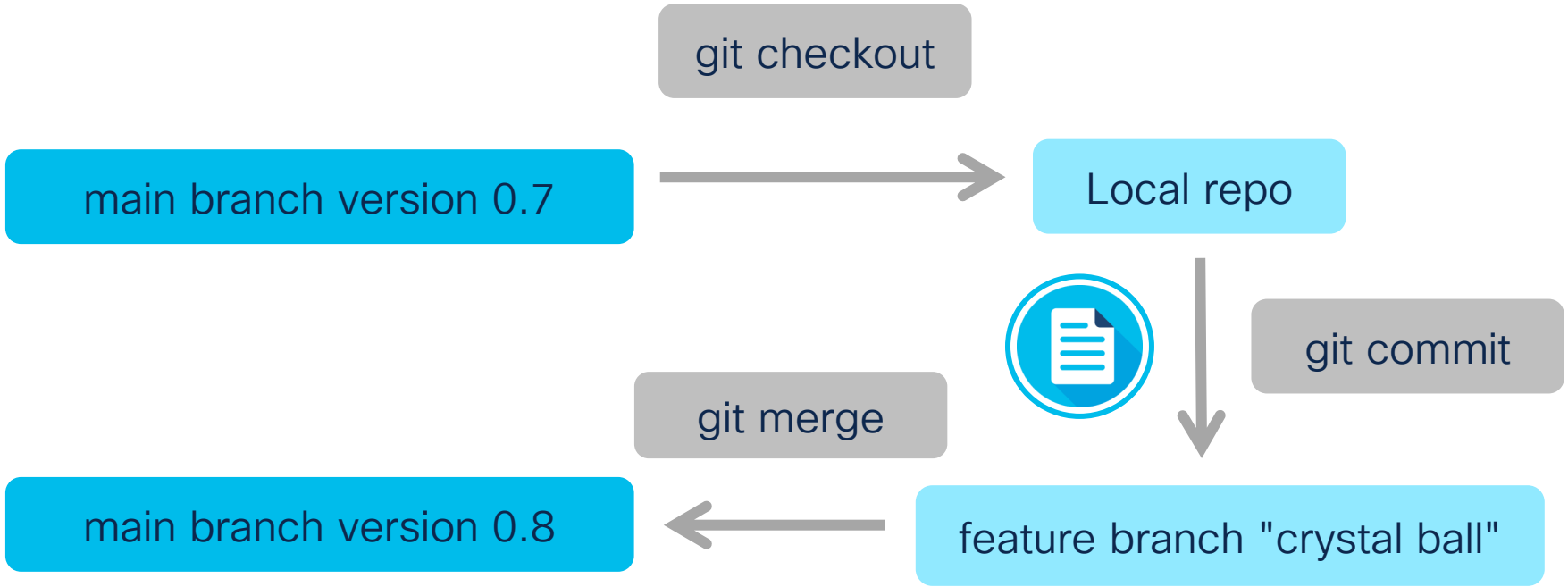
git commit

git merge

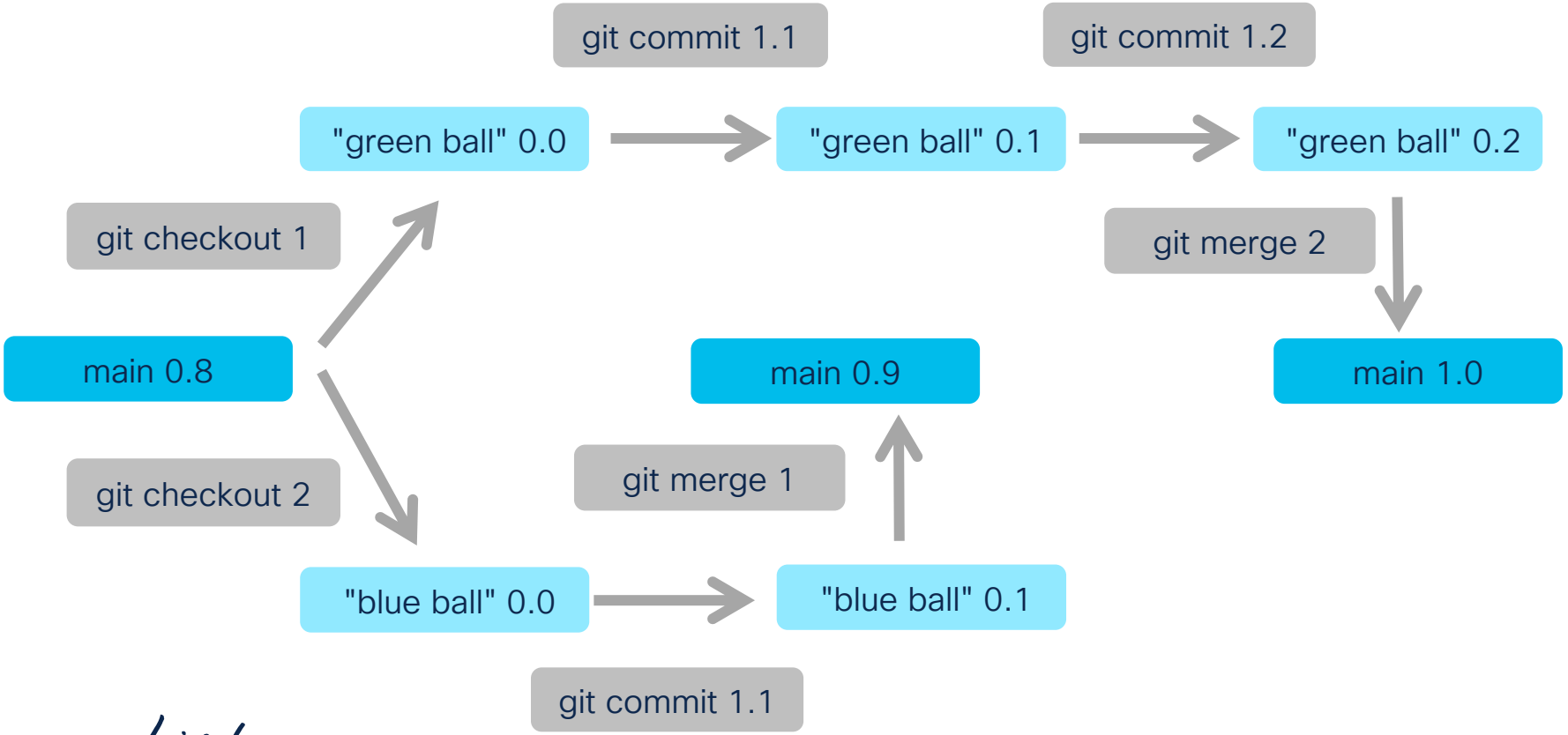
GIT basics

- GIT acronyms and few commands:
  - "local repo" – Local copy of the YAML files individual working on
  - "branch" – Separate line / path of development within a repository to add new features
  - "main branch" or "master" branch– The line of development which is in production
  - "git checkout" – Copy from a branch the content, and add this to a new branch
  - "git commit" – Approve changes done in one branch, and make them active in this branch
  - "git merge" – Merge two branches to one, e.g. activate a new feature from "feature-branch" in the "main branch".

GIT Branch Flow



GIT Branch Flow



GIT Branch Flow

From high level view, looks like new main branches are added with new feature in version, as well in real life with bug fixes for existing features

0.7 plus feature
"crystal ball"

0.8 plus feature
"blue ball"

0.9 plus feature
"green ball"

GIT basics

- Main GIT functions are:
  - Track changes and provide history
  - Multiple users can work independent in "own branches"
  - In parallel multiple versions of YAML files can exist and worked on
  - Tools to merge multiple "branches" into "main branch"

Questions you can answer with GIT

- Who approved and applied the changes 2 weeks ago?
- Hopefully, what was the intention for the change.

- Where and what was changed 2 weeks ago?

data/tenant_PROD_1.nac.yaml

```
305  337          external_endpoint_groups:
306                 - name: 'ext-epg'
     338          + - name: 'ext-epg-sdwan-2'
```

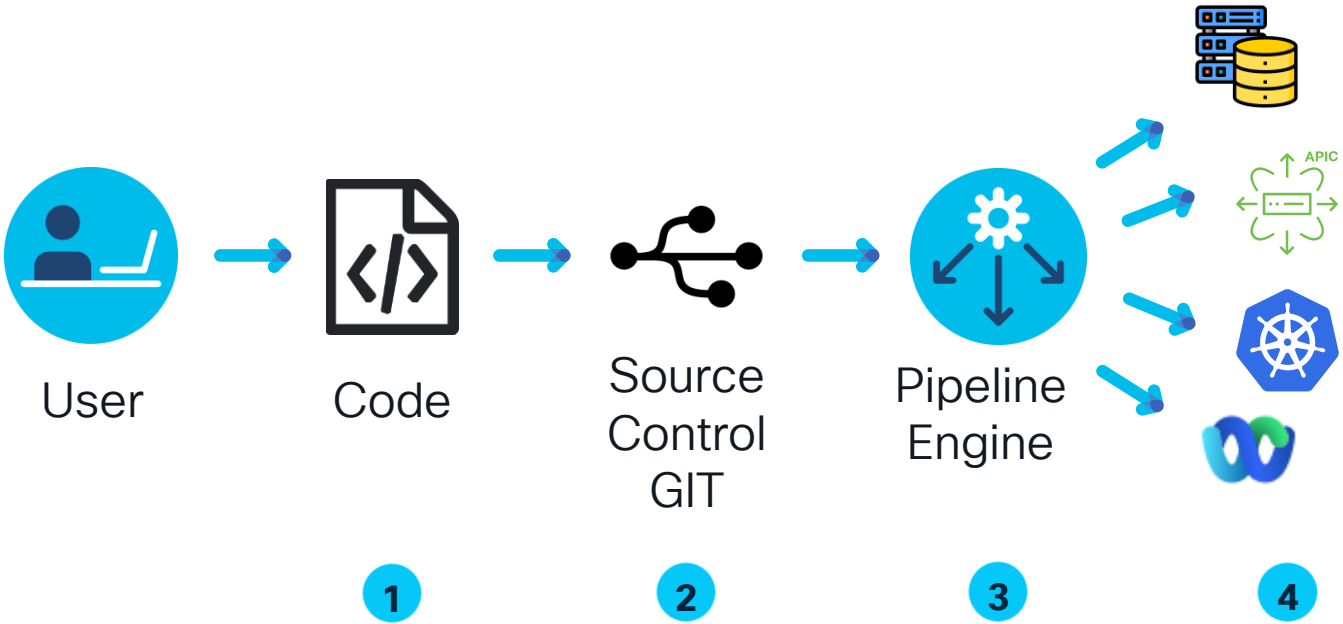
- One big difference to "just execute postman scripts"

Demo GIT

CI/CD Pipeline

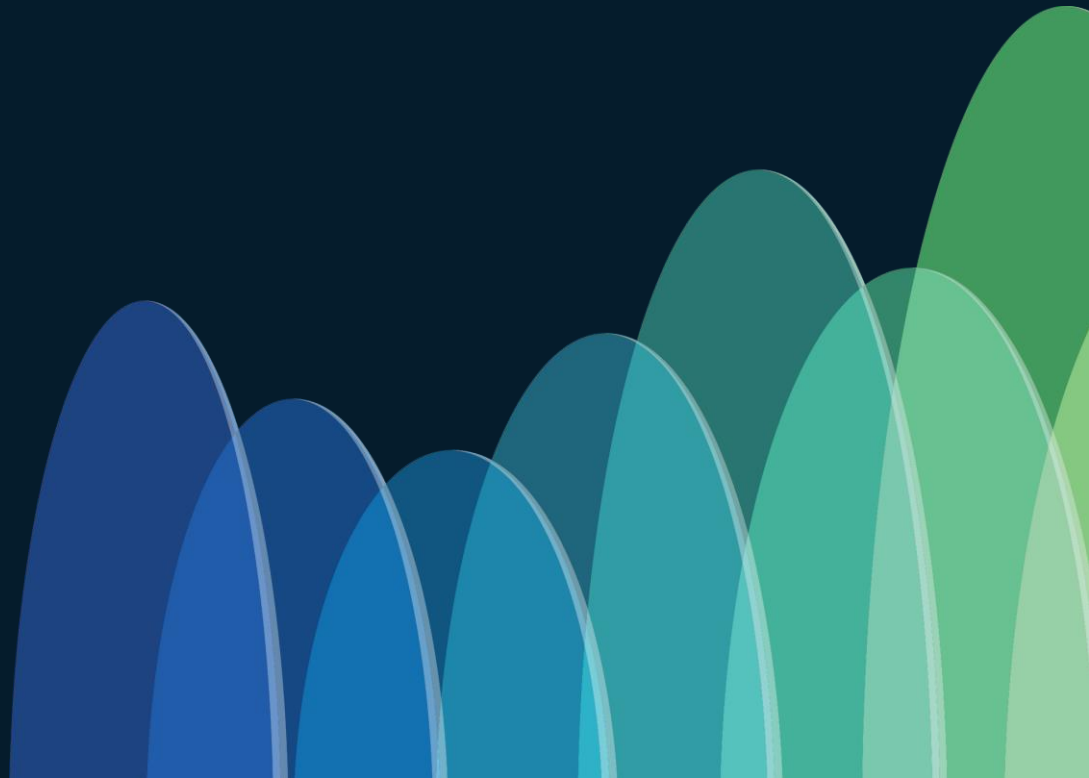
- Continuous Integration (CI)
  - Practice of merging all developer changes to a shared repo several times a day
  - It main include the creation and test of artifacts (executable, app, ...)
- Continuous Deployment (CD)
  - Approach to deliver new software functionalities frequently through automated deployments
  - Rely on Continuous Integration for tracking changes

What a CI/CD workflow looks like



Automation “apply” via Pipeline

ACI as Code



slido

Please download and install the Slido app on all computers you use

Are you using ... today?

ⓘ Start presenting to display the poll results on this slide.

AClasCode & Associated Cisco Services

Cisco Services team provided best practices & highly recommended add-ons

Ownership

Support

Cisco Services (CX) Customized

Ensure compliance with...
Nexus Dashboard Integration

Effectively communicate changes with...
Email & ChatOps Integrations

Avoid configuration errors with...
Semantic Pre-Change Validation

Integrate into YOUR CI/CD environment using the...
Platform Agnostic CI/CD Modules

Verify every change with...
Context-Driven, Automated Testing

Imports YOUR current configuration using the..
Brownfield Import Module

Cisco Services (CX) Best Practices

Cisco Professional Services (CX)

Cisco BU / CX Open Source

Nexus-as-Code Terraform Modules

Access Policies

Pod Policies

Interface Policies

Fabric Policies

Node Policies

Tenants

150+ Low-Level Terraform Modules

Cisco Professional Services (CX)

BU Open Source

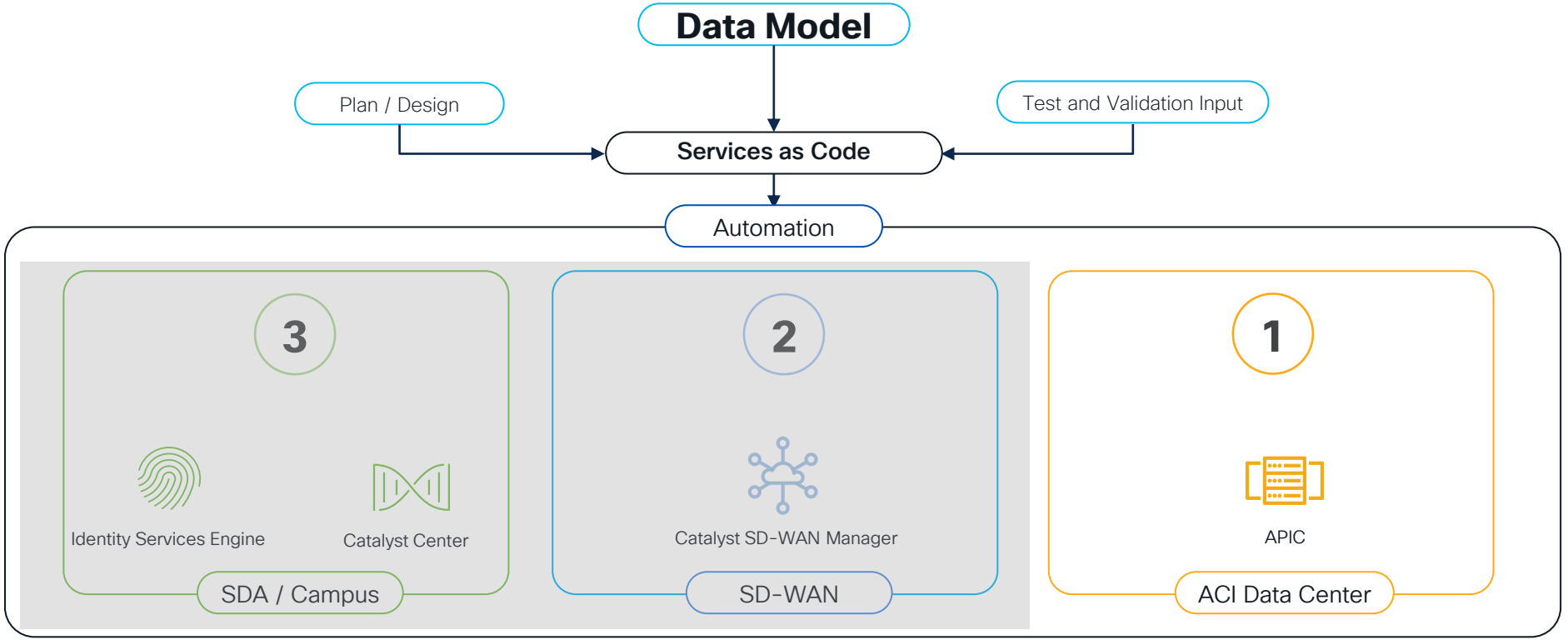
ACI Terraform Provider

TAC

Cisco Services team provides environment setup, direct support and knowledge transfers to reduce typical open-source ramp up time and installation efforts by 80%

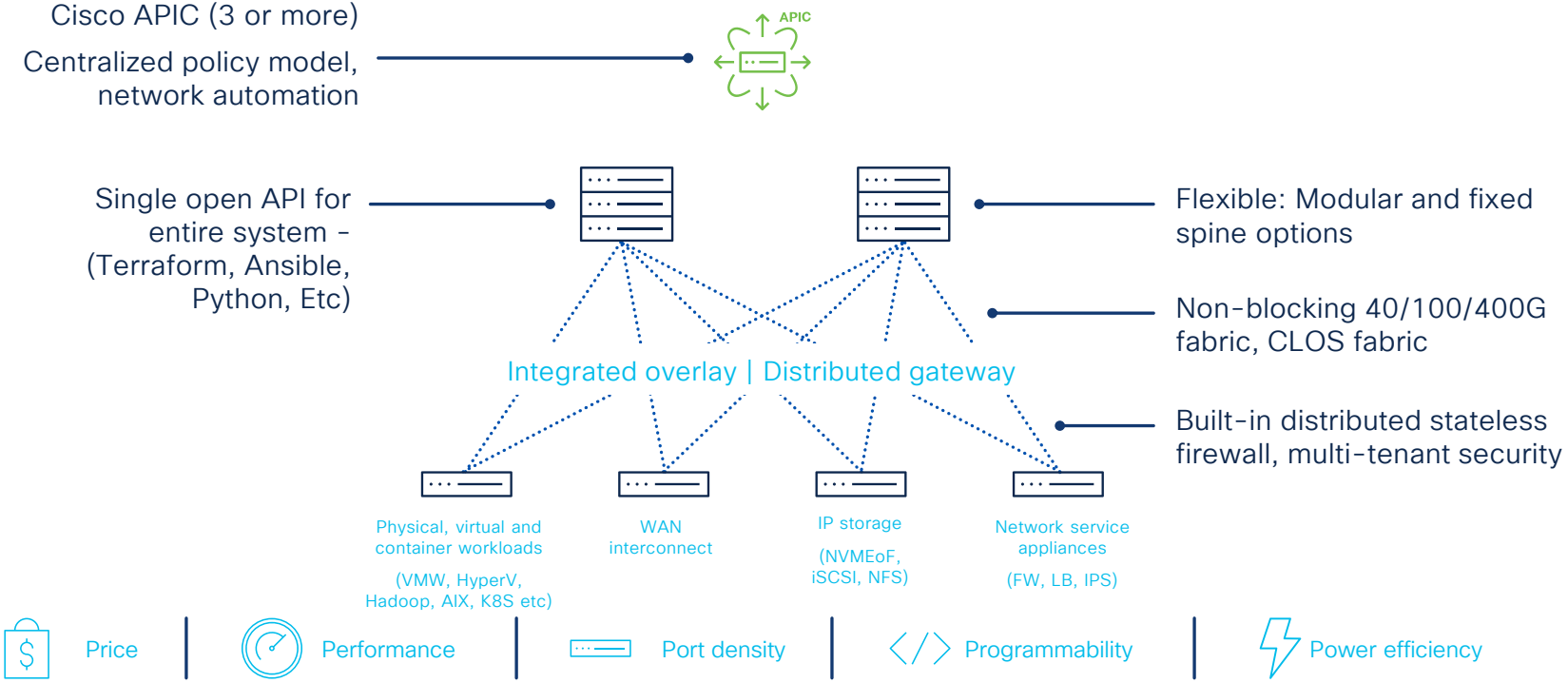
Provides direct support, which streamlines open-source feedback and simplifies updates as YOU don't need to manage/copy source-code

Back to the concept

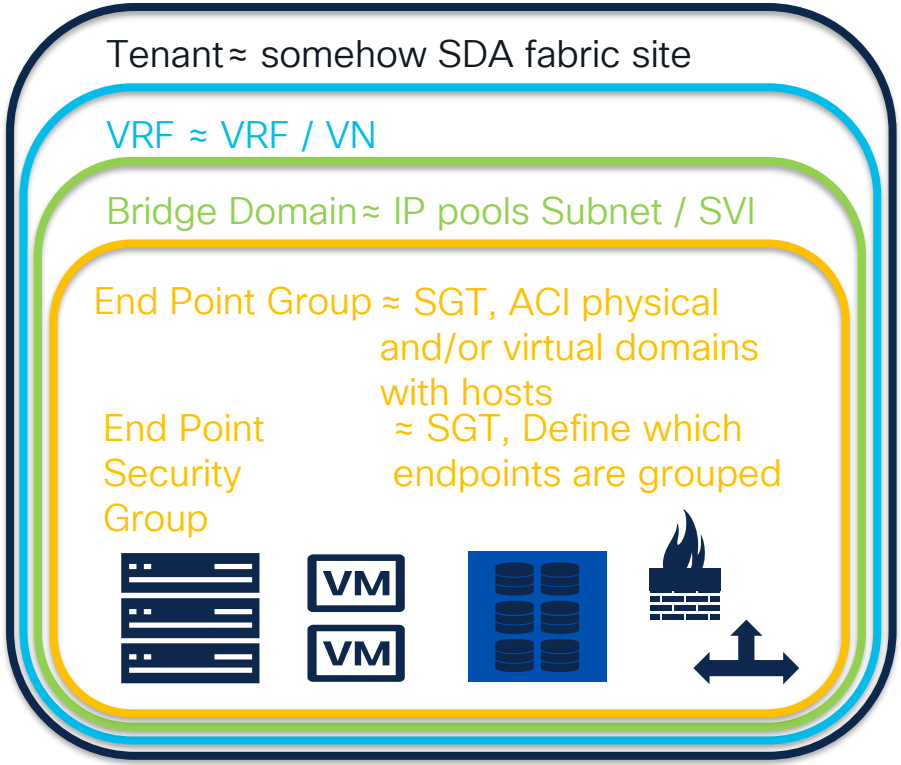


Application Centric Infrastructure building blocks

Built on Cisco Nexus 9000



The ACI Policy Model



BD = networker fun part

EPGs have two jobs:

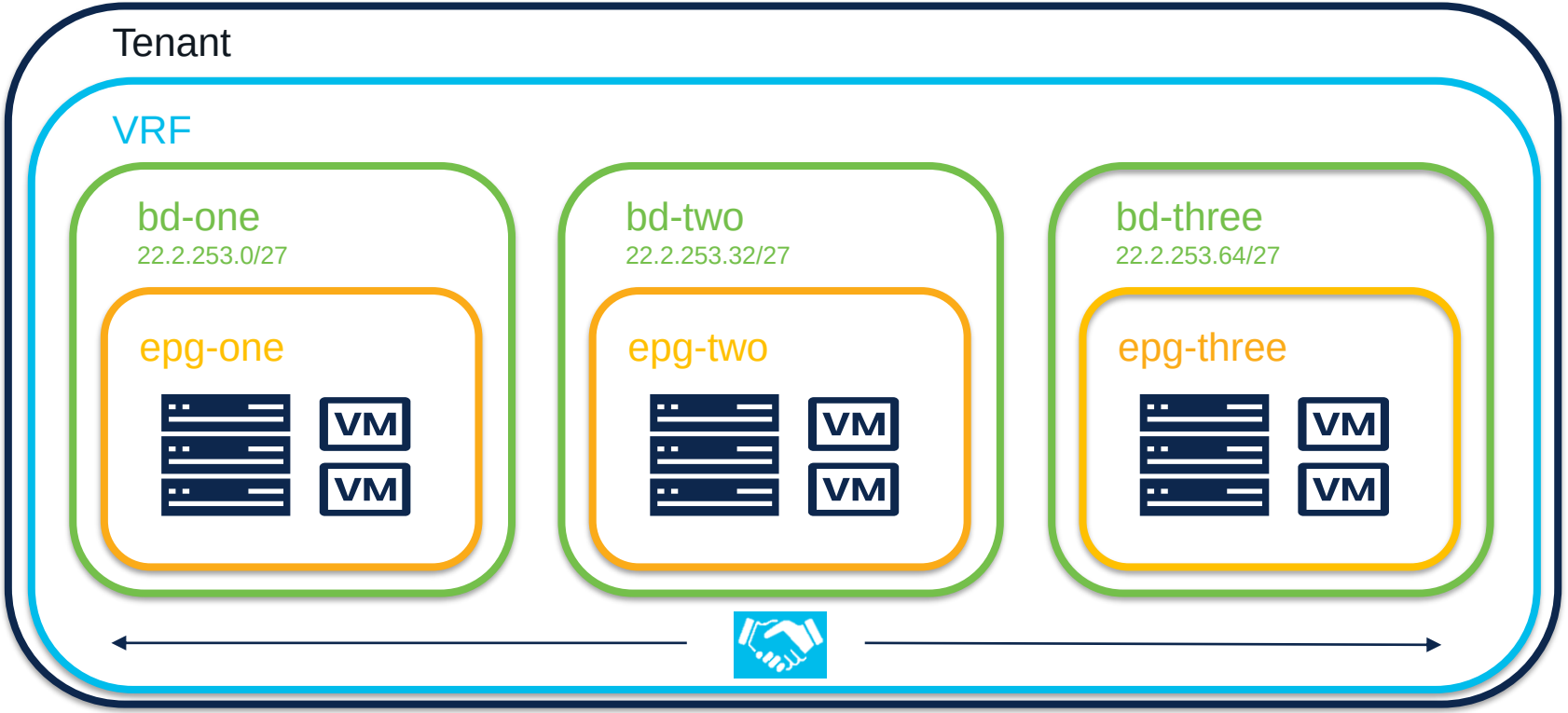
(1) Define which endpoints belong to EPG
(2) Endpoints communication paths

ESGs have two jobs:

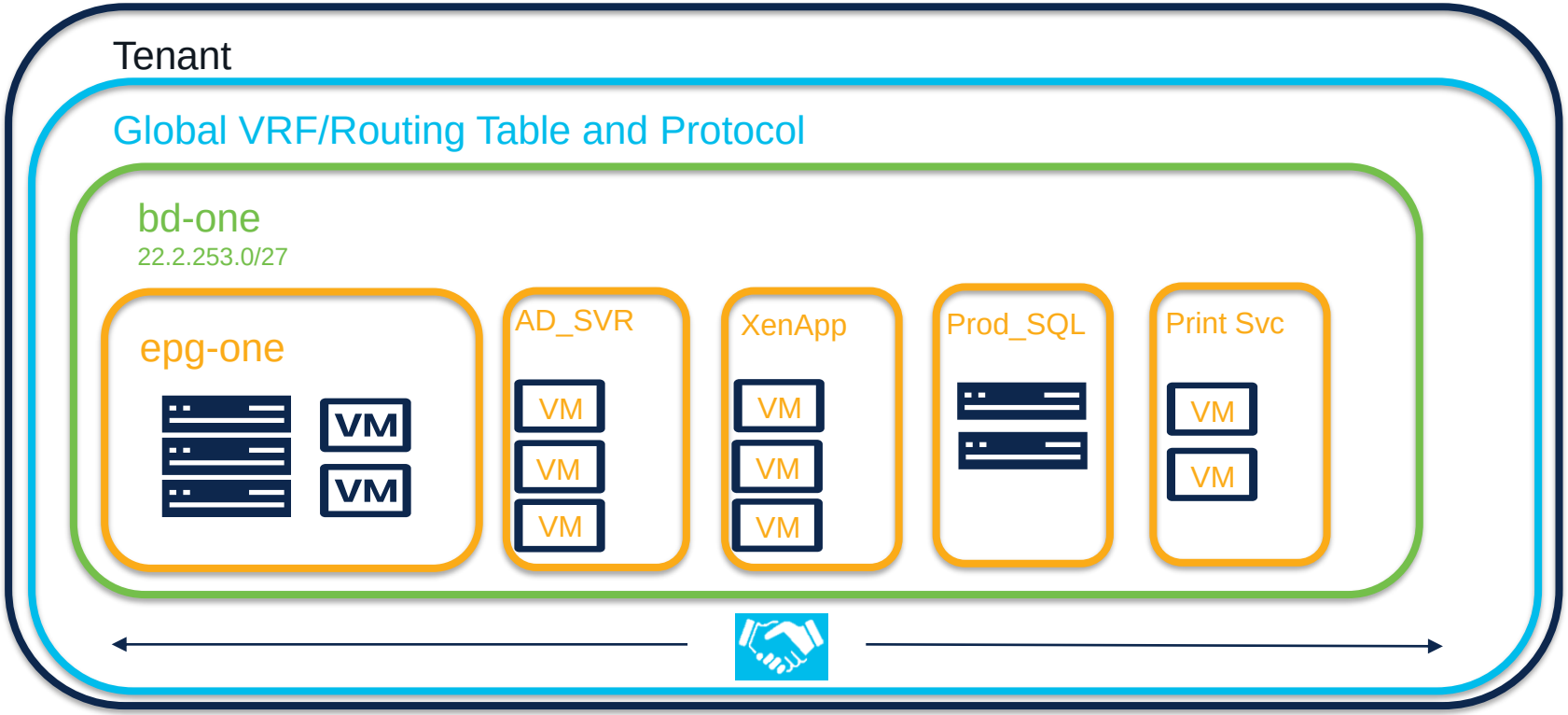
(1) Define which endpoints belong to ESG in much more flexible then EPGs
(2) Endpoints communication paths

Contracts ≈ More Intelligent ACLs, no Firewall

The ACI Policy Model – Starting off with ACI



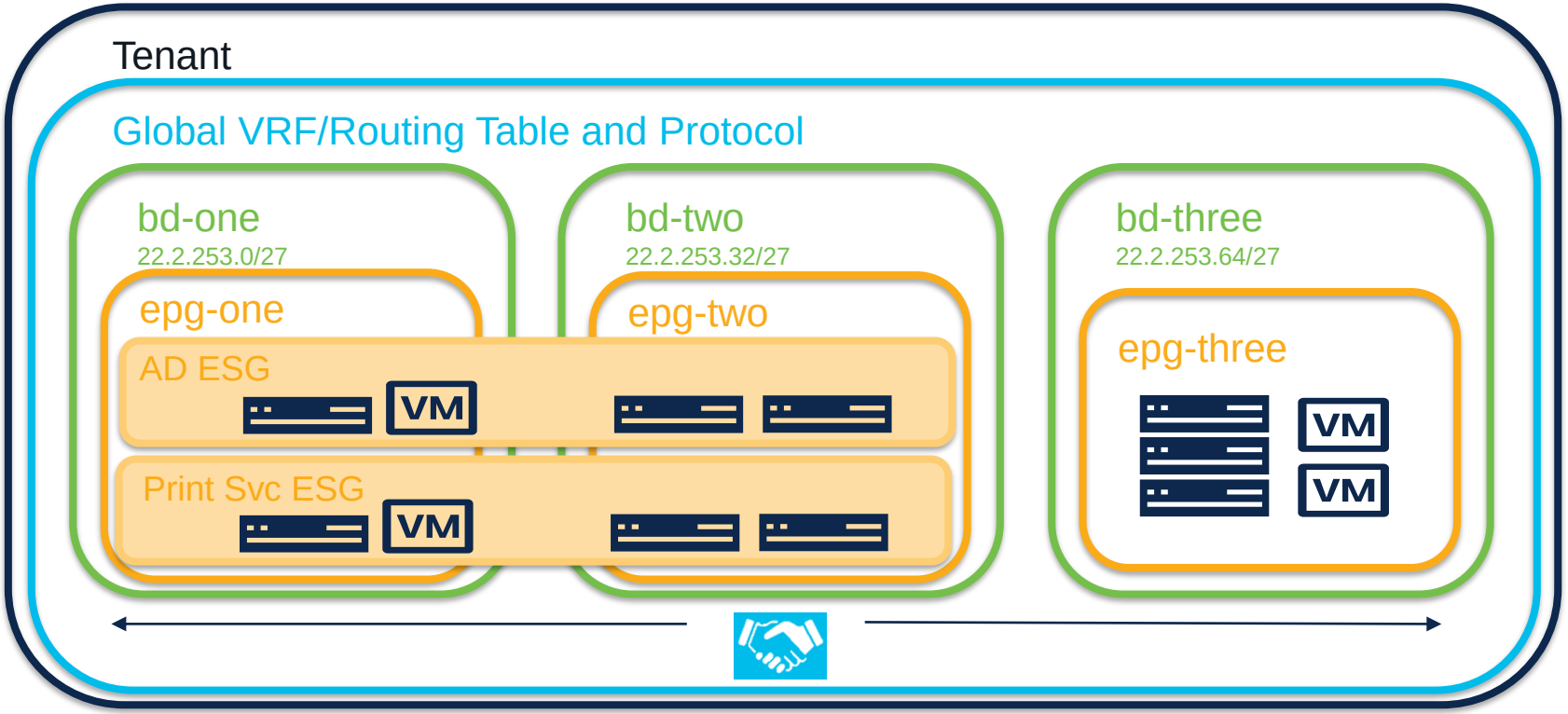
The ACI Policy Model – Extending the configuration
Endpoint Groups



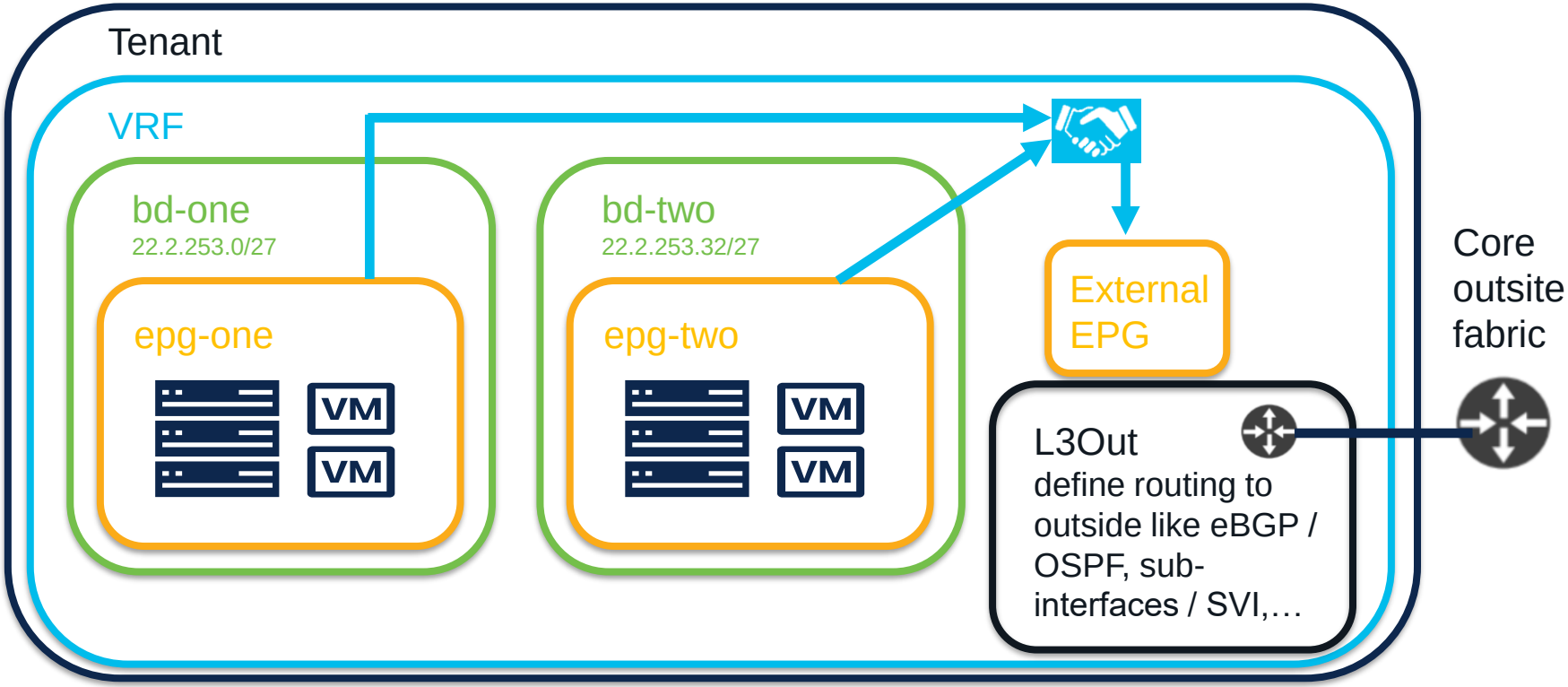
* Preferred group or vzAny Contract achieves the same outcome

The ACI Policy Model – Extending the configuration

Endpoint Security Groups (ESG) - ACI 5.0 and greater



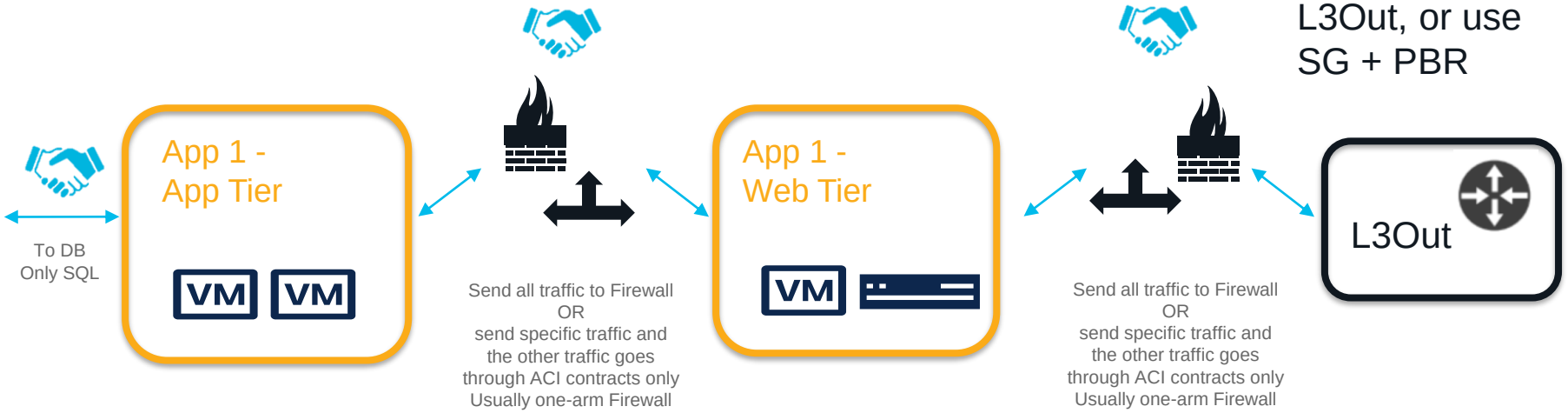
The ACI Policy Model – Add External Connection



* Preferred group or vzAny Contract achieves the same outcome

Advancing the ACI Configuration

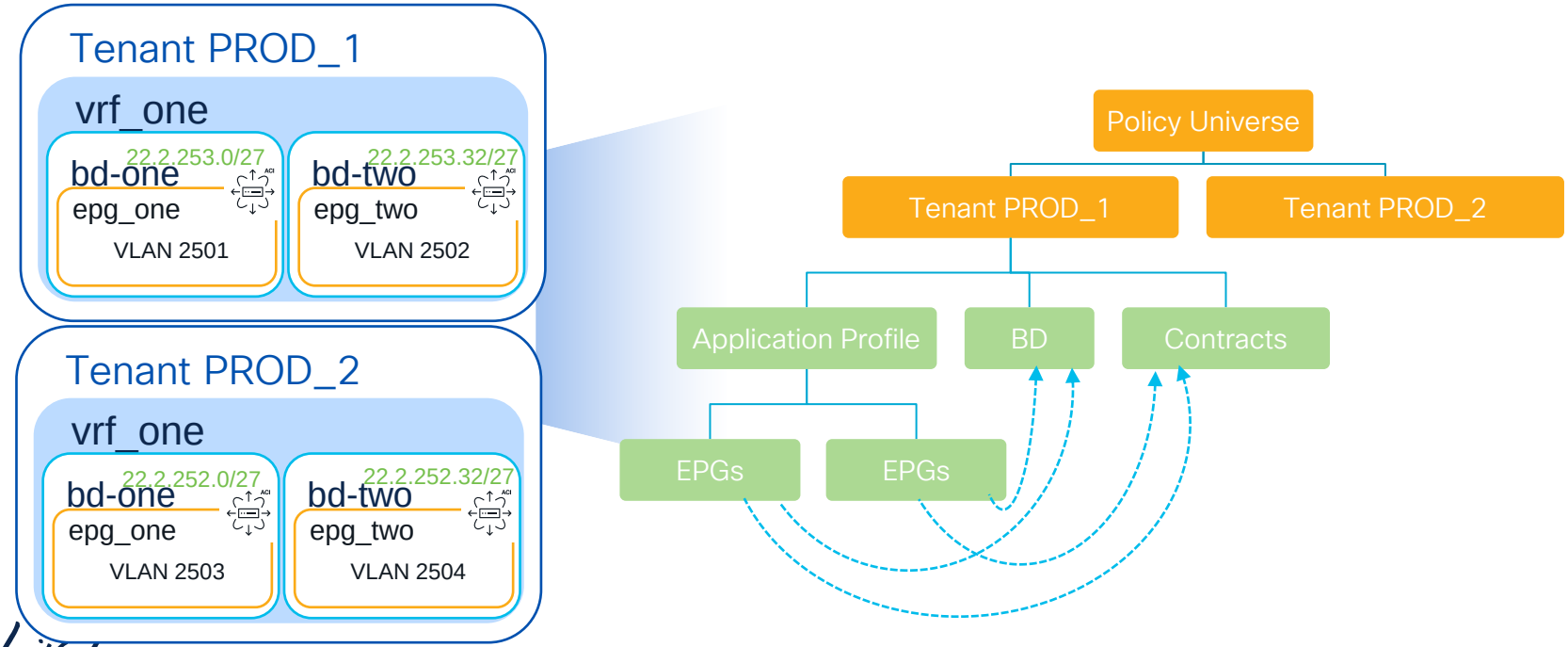
For North-South Firewall can be connected to L3Out, or use SG + PBR



Policy Based Redirect with Service Graphs

The key element is Data and abstraction
ACI Tenant

- Objects having dependencies to each other, as well sequence to create is important
- ACI-as-Code will take care



ACI YAML main.tf

```
terraform {      You, 5 months ago • adding ma
  backend "http" {
  }
}

module "nac-aci" {
  source  = "netascode/nac-aci/aci"
  version = "0.9.0"

  yaml_directories = ["data"]

  manage_access_policies     = true
  manage_fabric_policies     = true
  manage_pod_policies        = true
  manage_node_policies       = true
  manage_interface_policies  = true
  manage_tenants             = true

  write_default_values_file = "defaults.yaml"
}
```

```
terraform {
  nelazic, 7 months ago | 2 authors (nelazic and one other)
  backend "http" {
  }
}

Joerg Reinecke, 5 months ago | 3 authors (You and others)
module "nac-aci" {
  source  = "netascode/nac-aci/aci"
  version = "0.9.0"

  yaml_directories = ["data"]
      You, 7 months ago via  PR … • adding maste
  manage_access_policies     = false
  manage_fabric_policies     = false
  manage_pod_policies        = false
  manage_node_policies       = false
  manage_interface_policies  = false
  manage_tenants             = true

  write_default_values_file = "defaults.yaml"
}
```

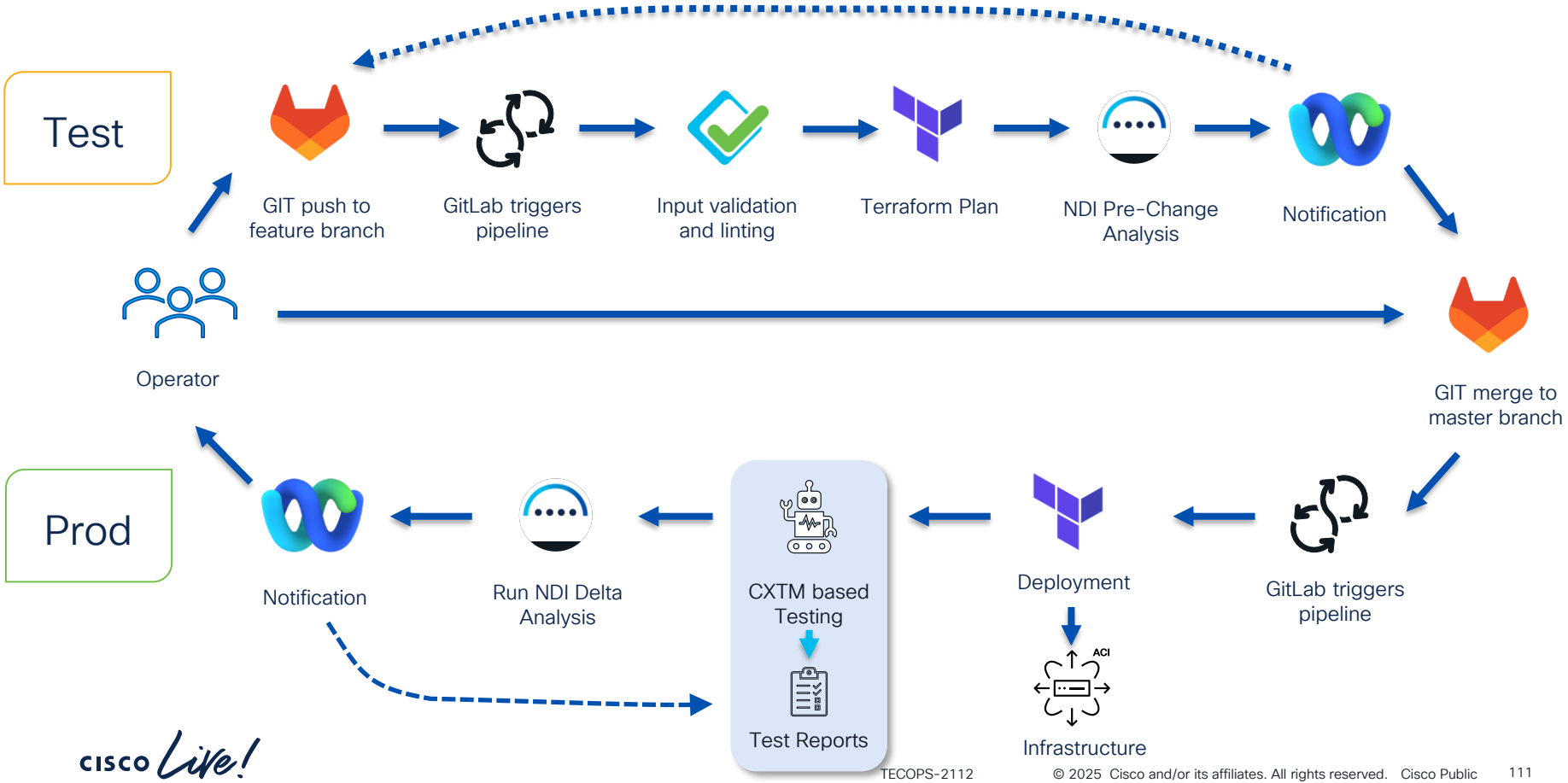
ACI YAML main.tf

- Define the modules been used
- Which parts of ACI config are applied
  - Uses "true" or "false"
- Example:
  - If only tenants configured, set all to "false" except for "manage_tenants = true"
  - Result is that all is ignored, except tenant YAMLs

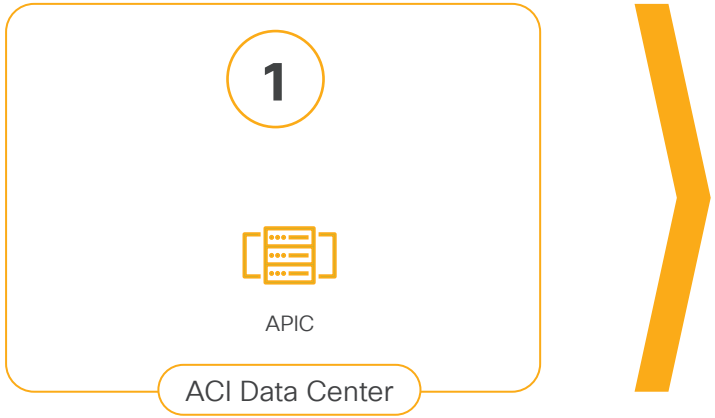
```
terraform {      You, 5 months ago • adding ma
  backend "http" {
  }
}

module "nac-aci" {
  source  = "netascode/nac-aci/aci"
  version = "0.9.0"

  yaml_directories = ["data"]

  manage_access_policies     = true
  manage_fabric_policies     = true
  manage_pod_policies        = true
  manage_node_policies       = true
  manage_interface_policies  = true
  manage_tenants             = true

  write_default_values_file = "defaults.yaml"
}
```

```
terraform {
   nelazic, 7 months ago | 2 authors (nelazic and one other)
  backend "http" {
  }
}

Joerg Reinecke, 5 months ago | 3 authors (You and others)
module "nac-aci" {
  source  = "netascode/nac-aci/aci"
  version = "0.9.0"

  yaml_directories = ["data"]
      You, 7 months ago via  PR … • adding maste
  manage_access_policies     = false
  manage_fabric_policies     = false
  manage_pod_policies        = false
  manage_node_policies       = false
  manage_interface_policies  = false
  manage_tenants             = true

  write_default_values_file = "defaults.yaml"
}
```

Example for ACI as Code workflow



DEMO Time



ACI.yaml

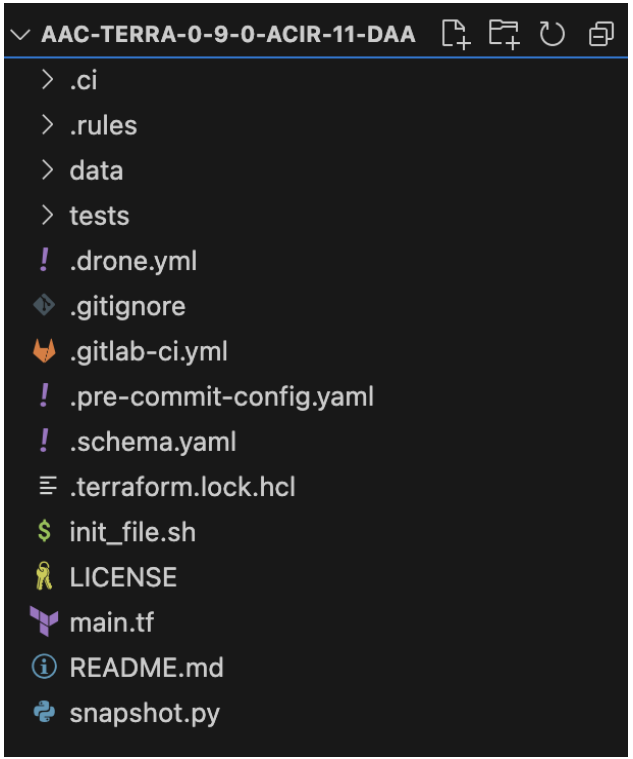
```yaml
---
apic:
  tenants:
    - name: PROD_1
      description: Cisco Live blue-print-1
      vrfs:
        - name: prod
          enforcement_direction: egress

      bridge_domains:

        - name: bd-one
          vrf: prod
          subnets:
            - ip: 22.2.253.1/27
      application_profiles:
        - name: production
          endpoint_groups:

            - name: epg-one
              bridge_domain: bd-one
              physical_domains:
                - DOM_PHY_SERVER
              static_ports:
                - node_id: 1011
                  pod_id: 1
                  port: 13
                  vlan: 2501
      l3outs:
# Transit to Services as Code Demo - SDA Munchen Hofbrauhaus
        - name: 'l3o-sdwan-1'
          vrf: prod
          domain: DOM_L3O_CORE
          description: Towards SDWAN BGP AS 64927

          node_profiles:   # Node profile and corrspondong link profile for IPv4
            - name: 'l3o_ipv4_NodeProfile'
              nodes:
                - node_id: 1001
                  pod_id: 1
                  router_id: 22.2.255.1
                - node_id: 1002
                  pod_id: 1
                  router_id: 22.2.255.2
```

Demo content

1. Show empty fabric
2. Recap what should be setup via pipeline
3. Explain YAML file structure and content of tenant
4. Run same pipeline then before
5. Verify all is setup
   - Ping from VMs to DGW
   - Ping from VMs to WAN and hosts in Campus failed

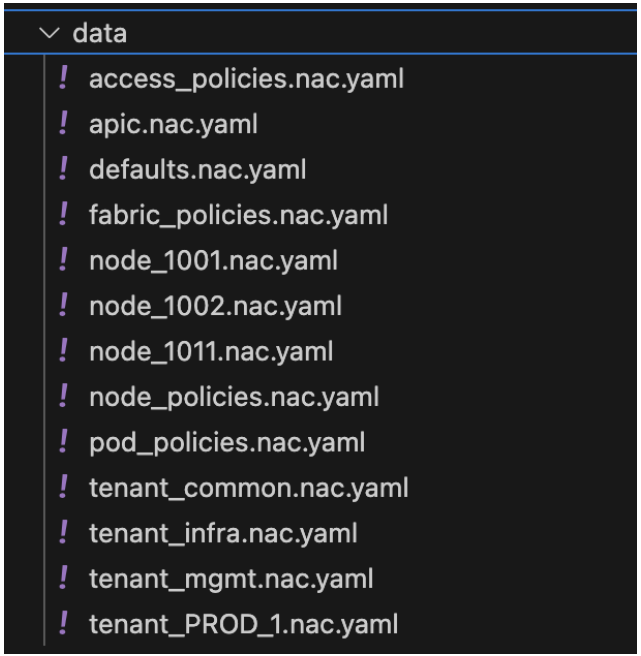
ACI YAML file structure

- Directory "data"
  - The "real data" for ACI, later more
- Directory ".rules" and "tests"
  - Verification tests on APIC after deploy
- File ".gitlab-ci.yaml"
  - Define the pipelines with different stages
- File "main.tf"
  - Define which parts of data directory is used, later more



ACI YAML data file structure

- These files define the ACI config itself
- All data can be in one file, or like here multiple
- To some extend YAMLs can be split in different repos, like one repo with fabric and access policies and other repo with only tenants.



ACI YAML tenant example

- "apic:" is the root object for data
- "tenants:" is start point for tenant definition
  - "-name: PROD_1" defines config for this tenant.
- "vrfs:" as well "bridge_domains" are children from the tenant PROD_1
  - Specific settings for them, are again children of the VRF and bridge domain.

```yaml
apic:
  tenants:
    - name: PROD_1
      description: Cisco Live blue-print-1
      vrfs:
        - name: prod
          enforcement_direction: egress

      bridge_domains:

        - name: bd-one
          vrf: prod
          subnets:
            - ip: 22.2.253.1/27
```

ACI YAML tenant example

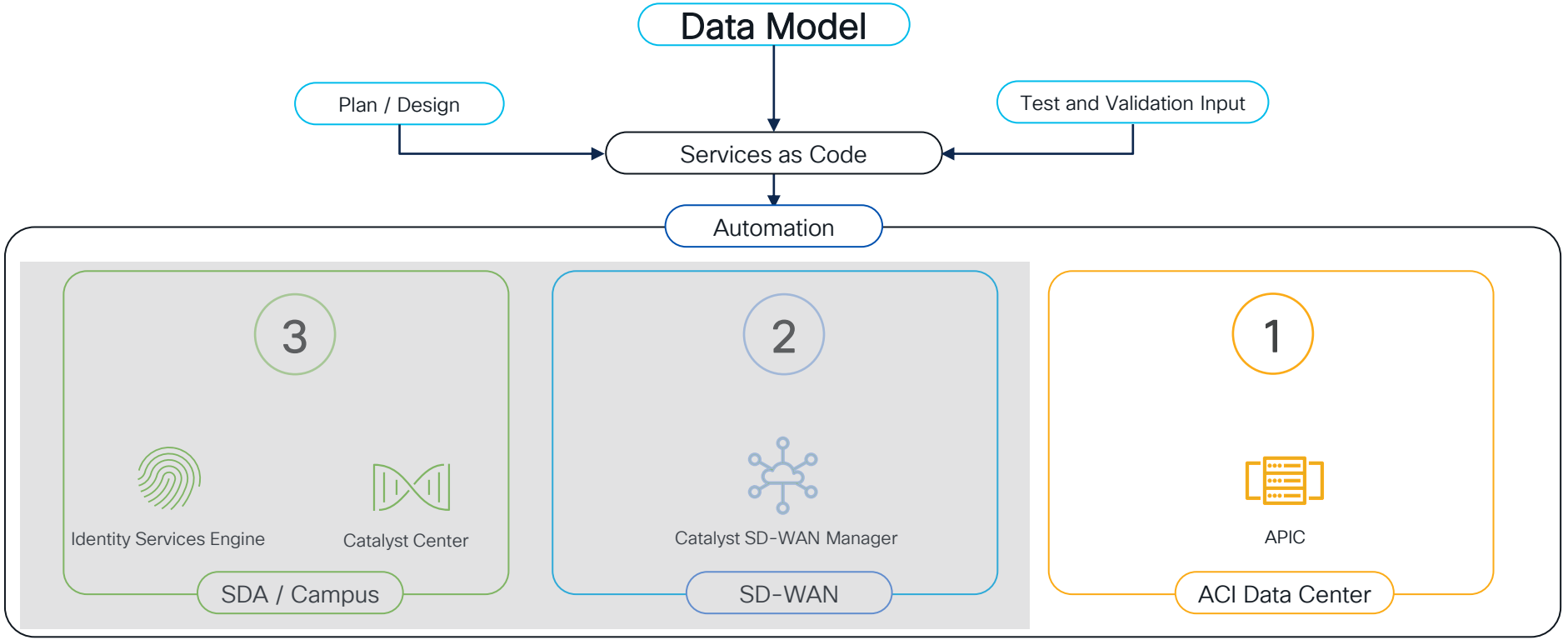
- Settings define one EPG
  - The name is "epg-one" with it's bridge domain
  - Define the physical domain belong to this EPG and the static ports.
  - Which for which contracts this EPG is provider or consumer.

```yaml
- name: epg-one
  bridge_domain: bd-one
  physical_domains:
    - DOM_PHY_SERVER
  static_ports:
    - node_id: 1011
      pod_id: 1
      port: 13
      vlan: 2501
    - node_id: 1011
      pod_id: 1
      port: 15
      vlan: 2501
    - node_id: 1011
      pod_id: 1
      port: 16
      vlan: 2501

  contracts:
    providers:
      - con-l3o
      - con-common-policy
```

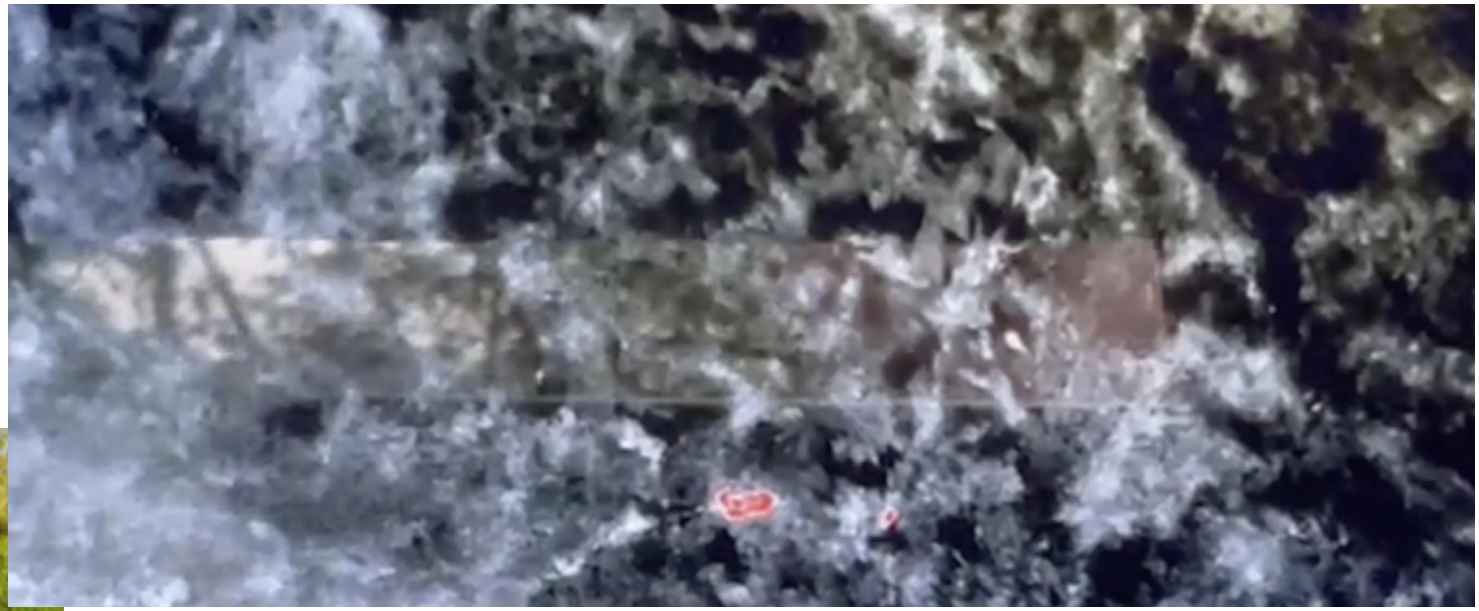
ACI Fabric baseline settings after fabric reset

- Discover the switches, until they are in "Active" status
  - Fabric / Inventory / Fabric Membership / Registered Nodes
- Enable GiPO, because it is "one time setting"
  - It will be done by AAC terraform as default, because best practice
  - System Settings / Fabric-Wide Settings / Reallocate Gipo - enable this
- Set MCP password
  - It will be done by AAC terraform as default, because best practice.
  - In CI/CD run it might cause problems, if AAC set this and not already done by GUI.
  - Fabric / Access Policy / Policies / Global / MCP
  - Enable + Enable POC PDU per VLAN + Key: "XYZ"
- Add AES encryption password
  - Required for 6.1.2 onward and best practice.
  - Not done by AAC by default but should be set.
  - System Settings / Global AES Password
  - Enable + AES encryption Password: XXXYYYZZZXXXYYYZZZ

Demo Time → ACI as Code



”Increase Efficiency with Service as Code for scalable, repeatable infrastructure automation."

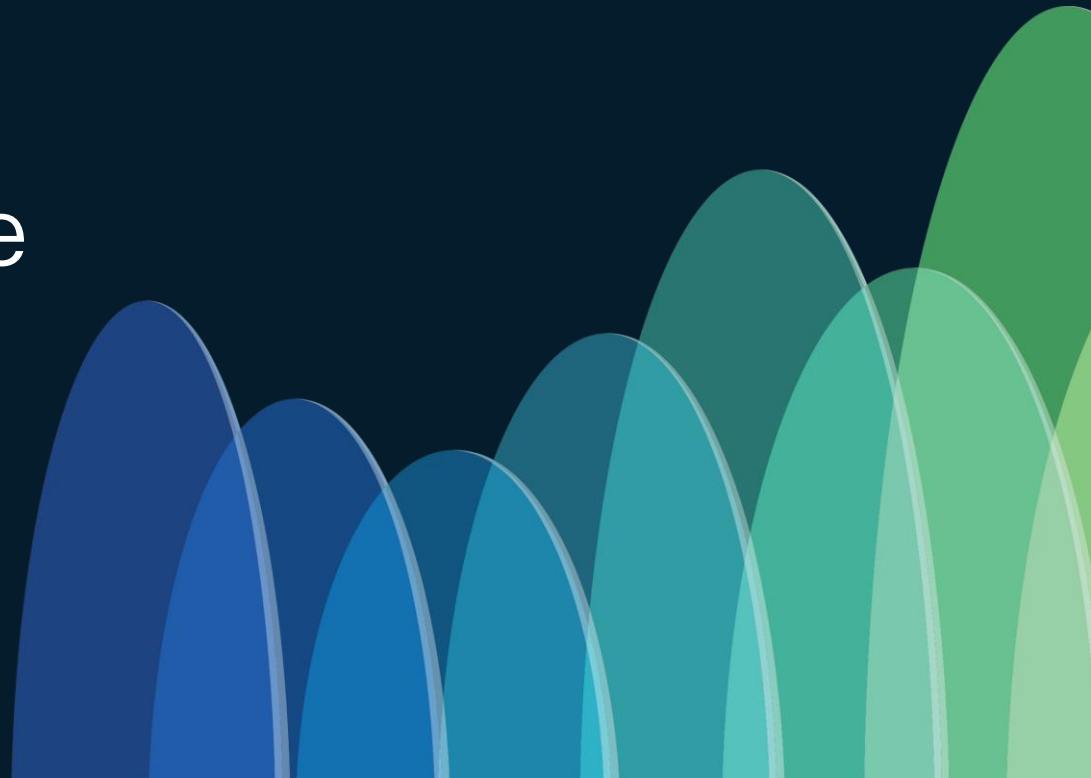


Source: https://www.drone-zone.de/

...automated flight and crisp views

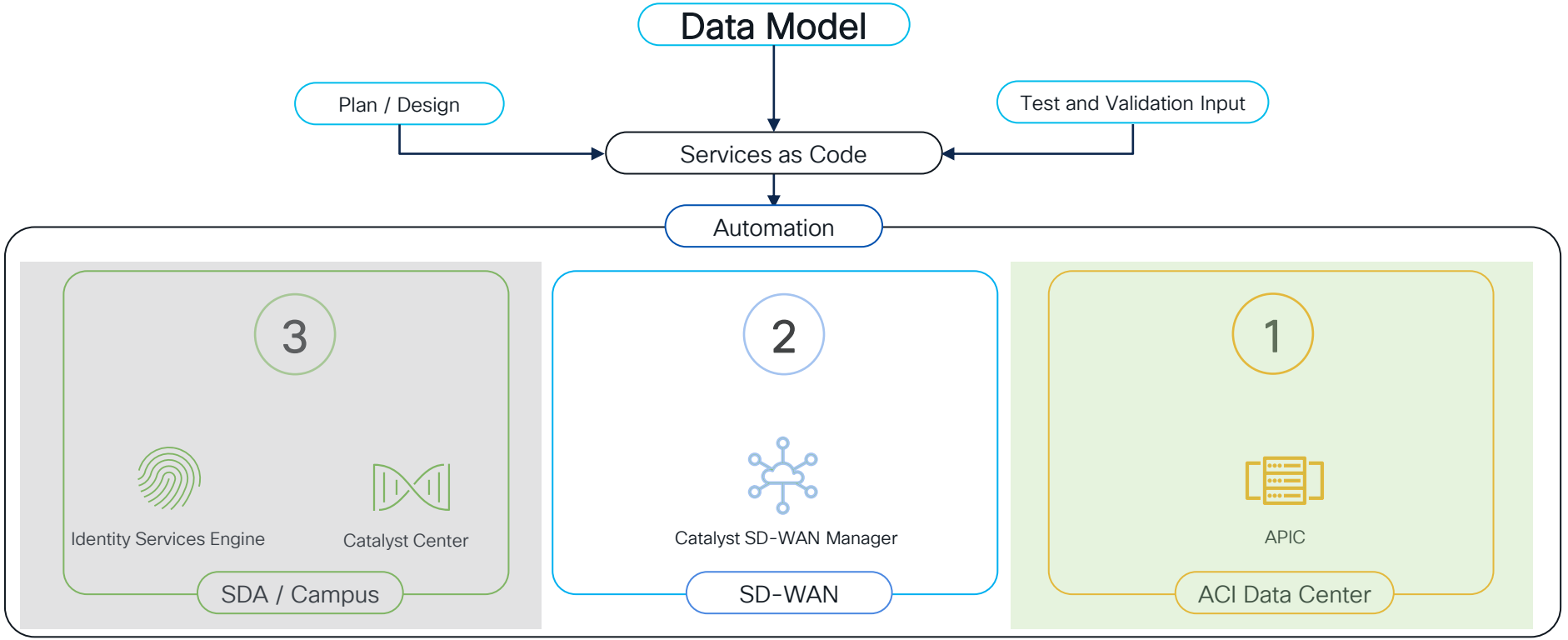
Automation “apply”

SDWAN as Code

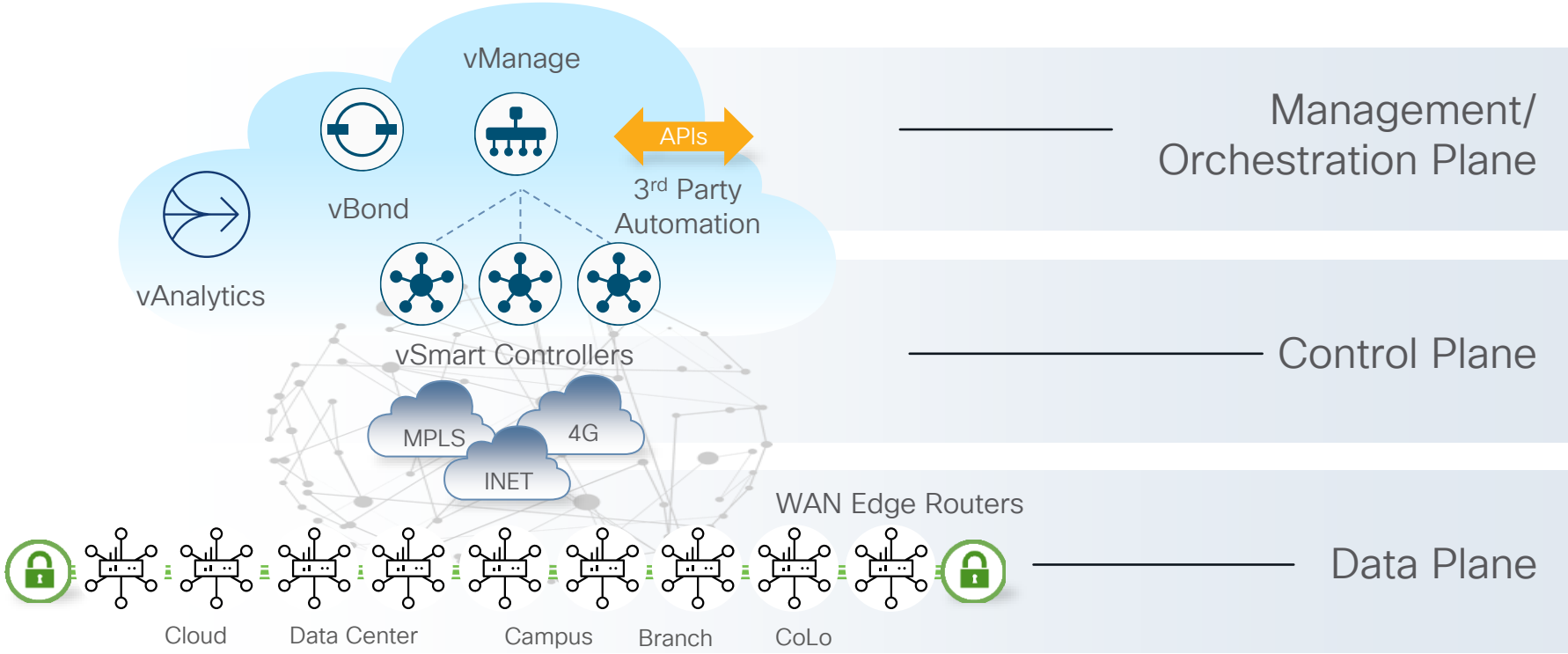


cisco Live!

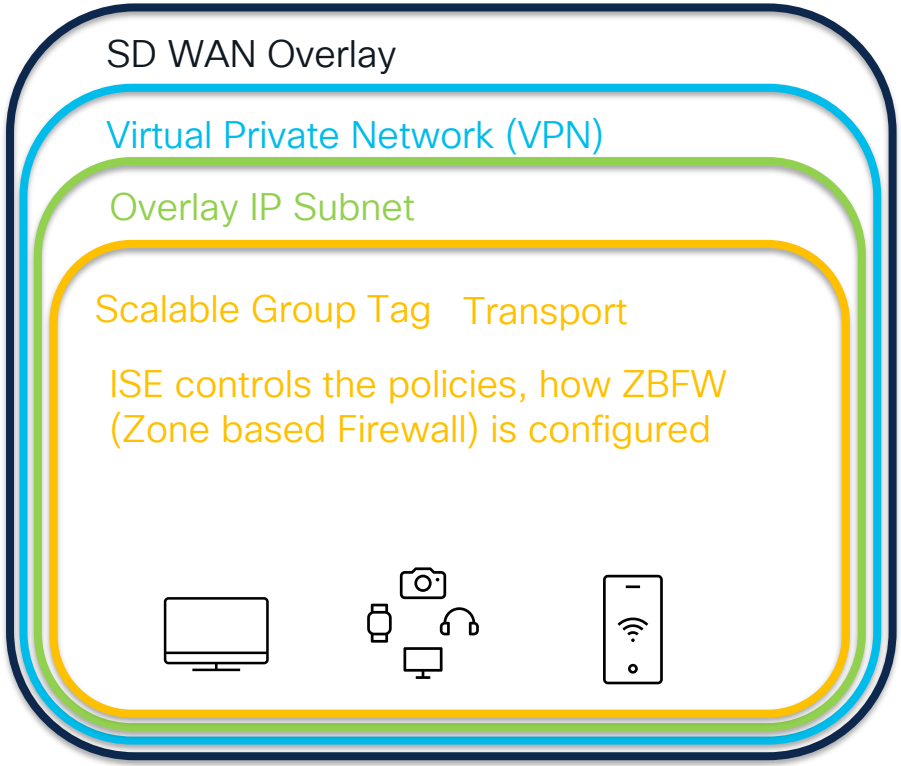
SDWAN as Code



Cisco SD-WAN Solution Overview



The SD WAN Policy Model



ISE Integrates with vSmart / vMange

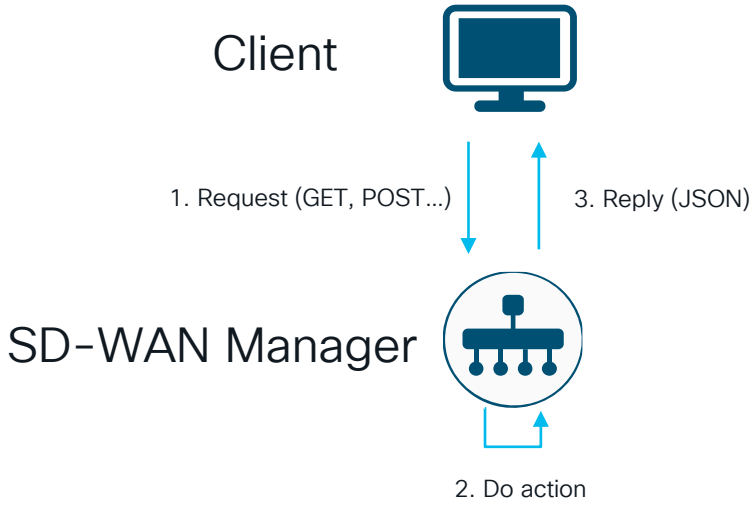
SGT are learned by cEdges

(1) Geed ZBFW objectless
(2) Rules can be applied on SGT

ZBFW on cEdges

SD-WAN Configuration Model and REST API

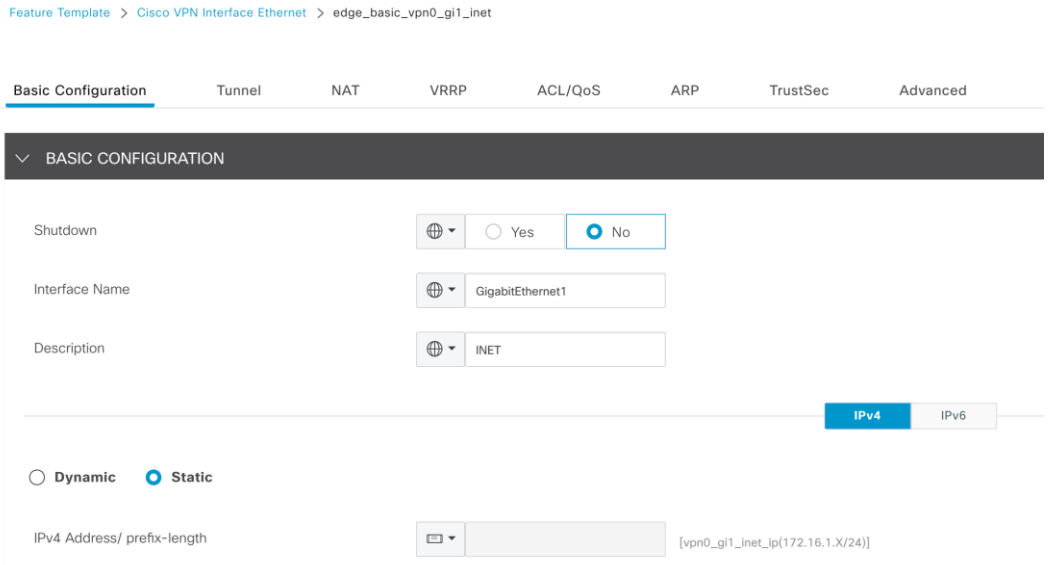
- SD-WAN stores all configuration as JSON objects (lists, policies, feature/device templates)
- SD-WAN configuration objects are fully accessible through a REST API



SDWAN API Feature Template

https://<manager_ip>/dataservice/ template/feature/object/a42ebf55-9496-49ca-bbfd-ff6bd866ae7a

Feature Template API Example

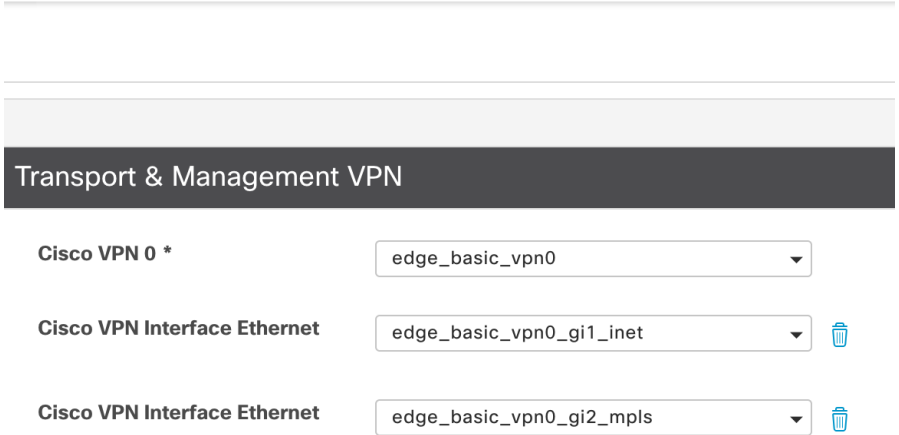



```
{
  "deviceType": [
    "vedge-C8000V"
  ],
  "templateType": "cisco_vpn_interface",
  "templateDefinition": {
    "if-name": {
      "vipObjectType": "object",
      "vipType": "constant",
      "vipValue": "GigabitEthernet1",
      "vipVariableName": "vpn_if_name"
    },
    "description": {
      "vipObjectType": "object",
      "vipType": "constant",
      "vipValue": "INET",
      "vipVariableName": "vpn_if_description"
    },
    "ip": {
      "address": {
        "vipObjectType": "object",
        "vipType": "variableName",
        "vipValue": "",
        "vipVariableName": "vpn0_gi1_inet_ip(172.16.1.X/24)"
      },
  ...
  },
  "templateId": "a42ebf55-9496-49ca-bbfd-ff6bd866ae7a",
  "templateName": "edge_basic_vpn0_gi1_inet",
  "templateDescription": "Do not modify!",
}
```

SDWAN API Device Template

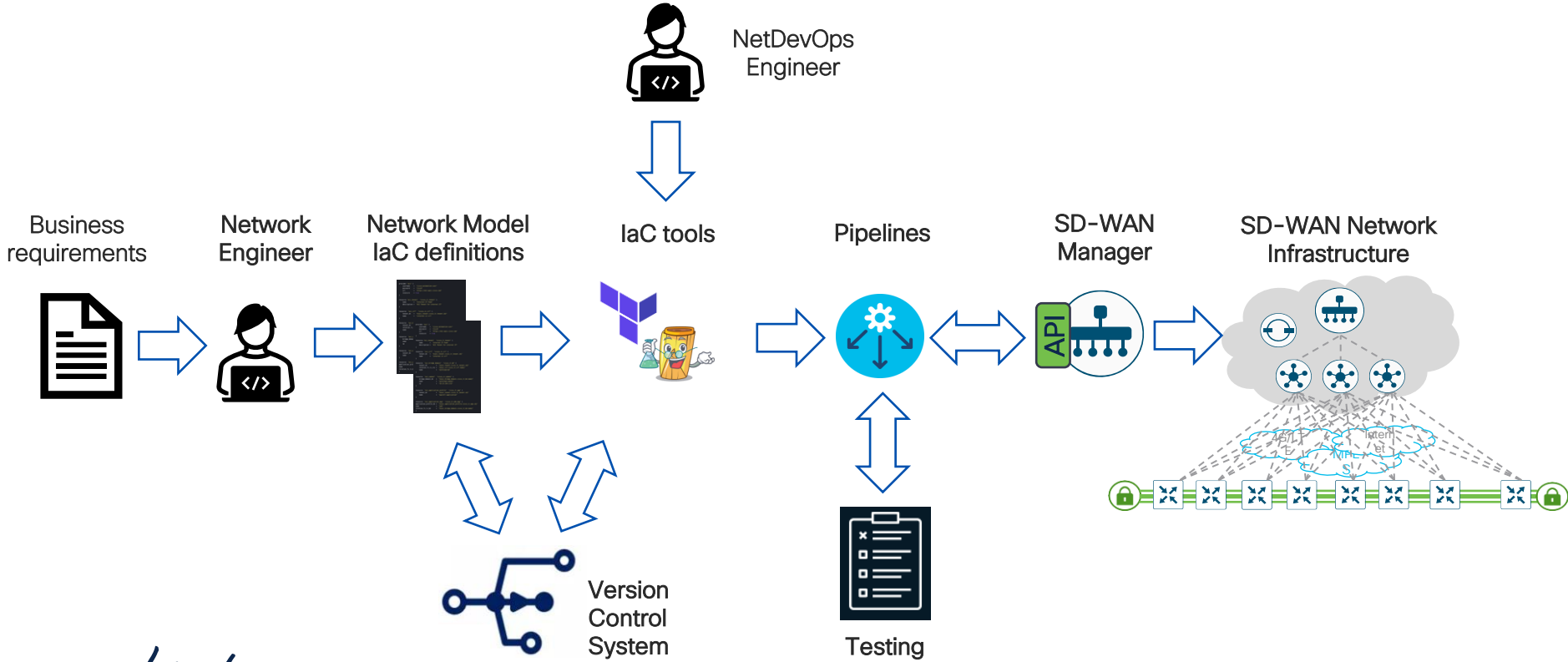
https://<manager_ip>/dataservice/ template/device/object/71d41fdc-1b5f-47c3-896b-9d59b24668cd

Device Template API Example

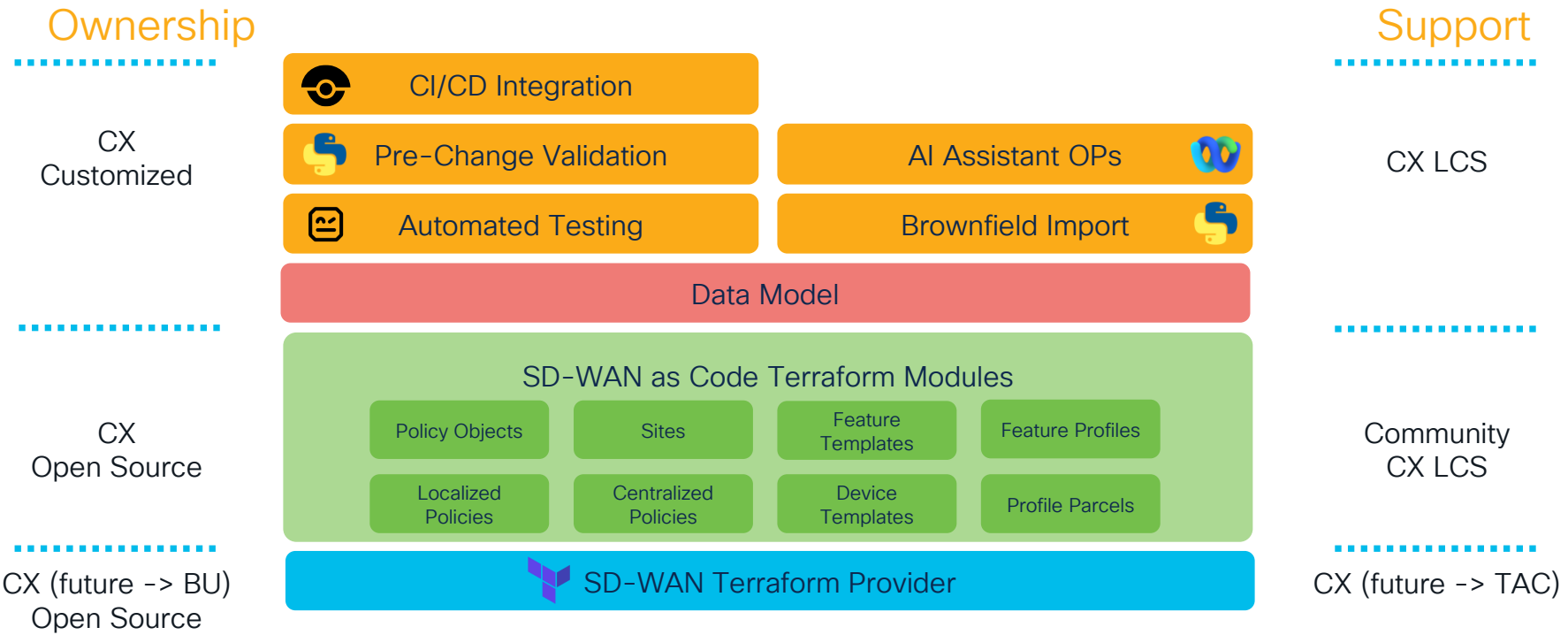



```
{
  "templateName": "edge_basic",
  "templateDescription": "Do not modify!",
  "deviceType": "vedge-C8000V",
  "generalTemplates": [
  ...
    {
      "templateId": "3b5d7f3e-1753-4085-8bd8-e98314dec642",
      "templateType": "cisco_vpn",
      "subTemplates": [
        {
          "templateId": "a42ebf55-9496-49ca-bbfd-ff6bd866ae7a",
          "templateType": "cisco_vpn_interface"
        },
        {
          "templateId": "eeb8babd-2b7e-40dc-939d-addc4eb7ffbd",
          "templateType": "cisco_vpn_interface"
        }
      ]
    },
  ...
  "templateId": "71d41fdc-1b5f-47c3-896b-9d59b24668cd"
}
```

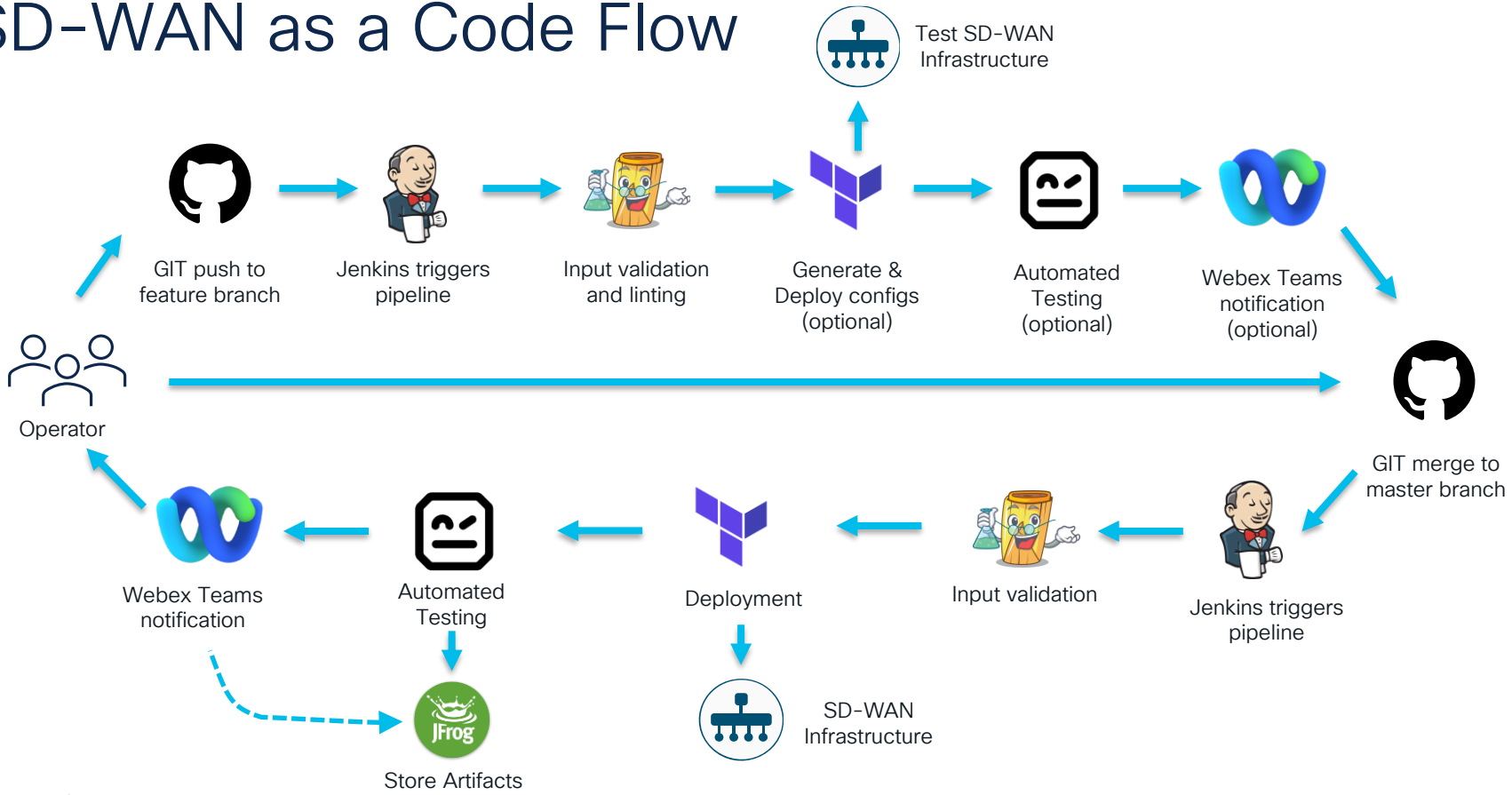
Cisco SD-WAN as Code Framework



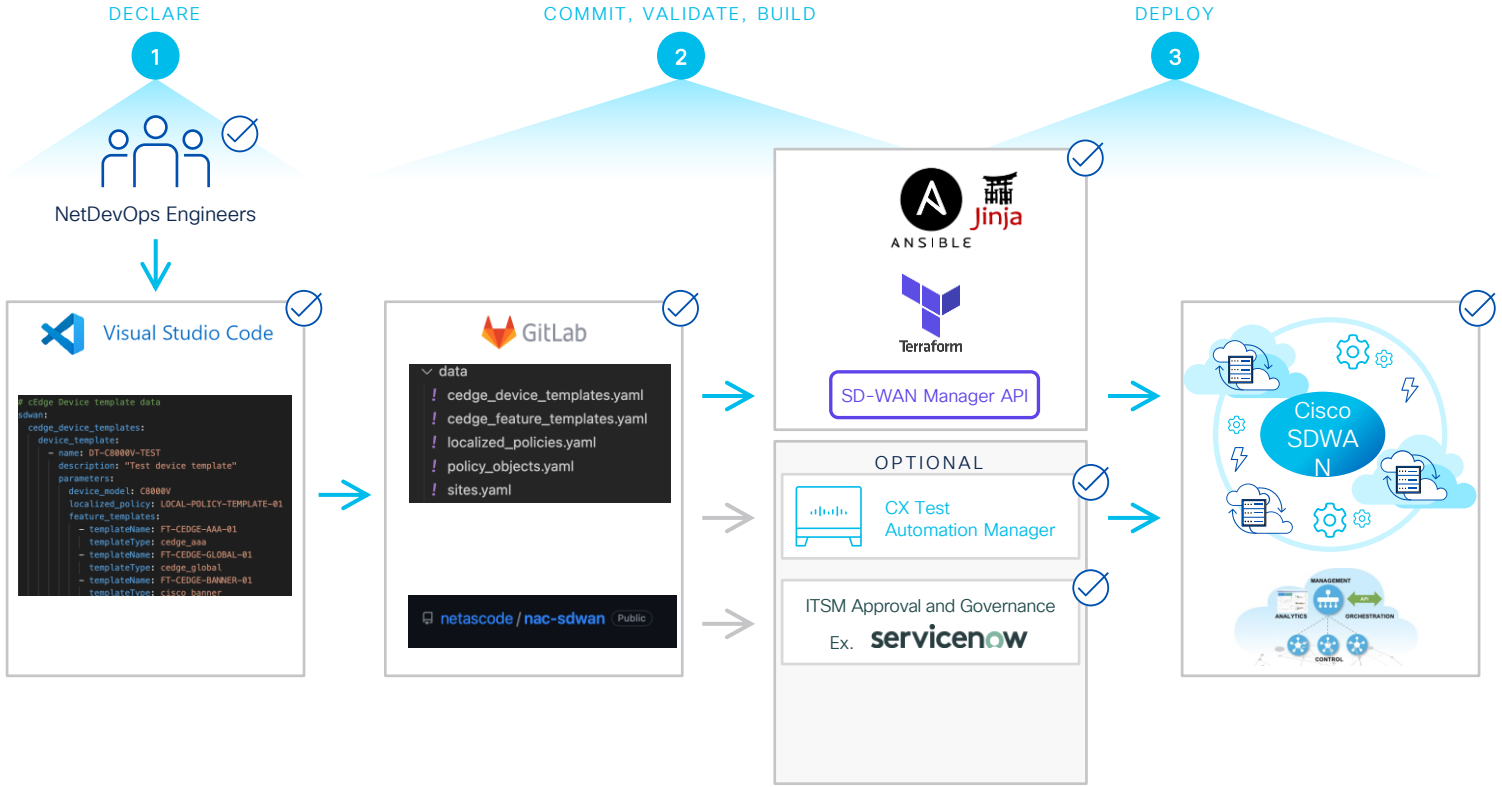
SD-WAN as Code Overview



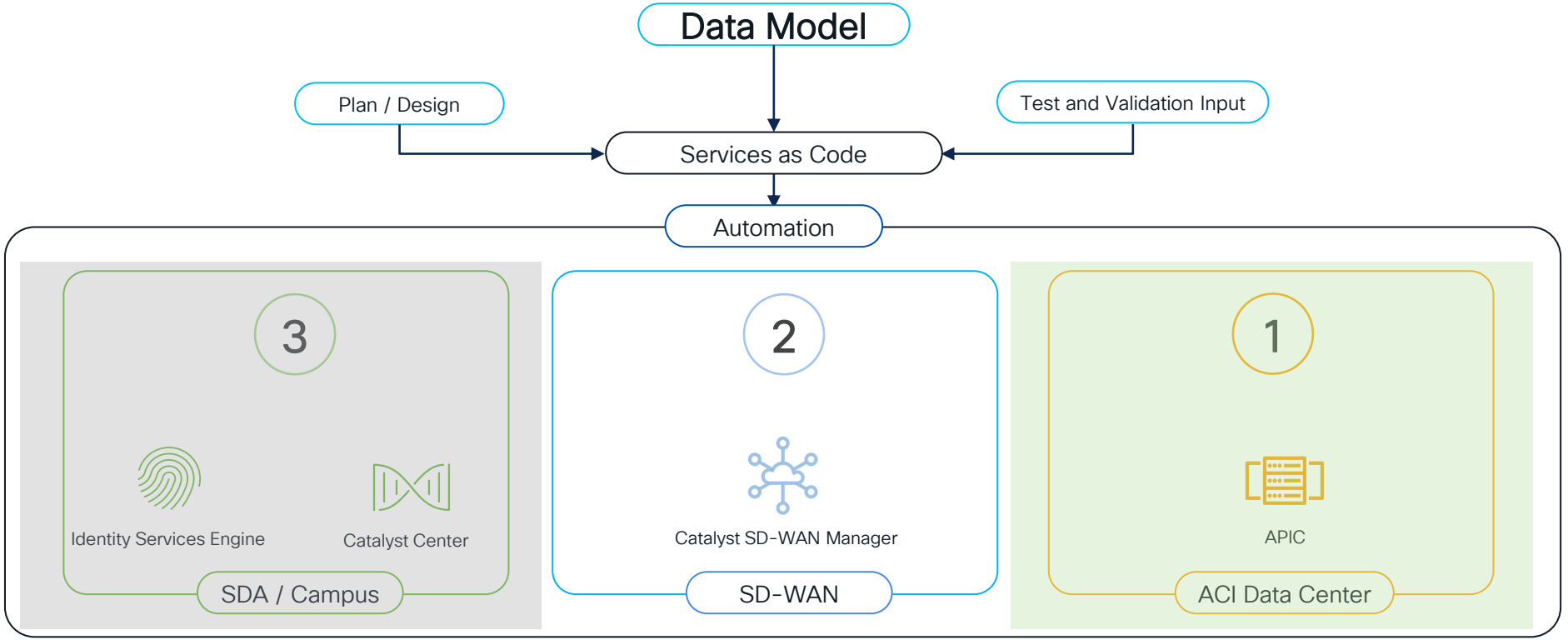
SD-WAN as a Code Flow



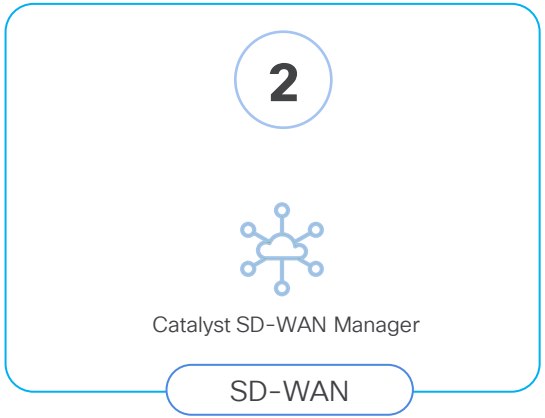
SDWAN as Code example



Demo Time → SDWAN as Code



DEMO Time



SDWAN.yaml 1 ●

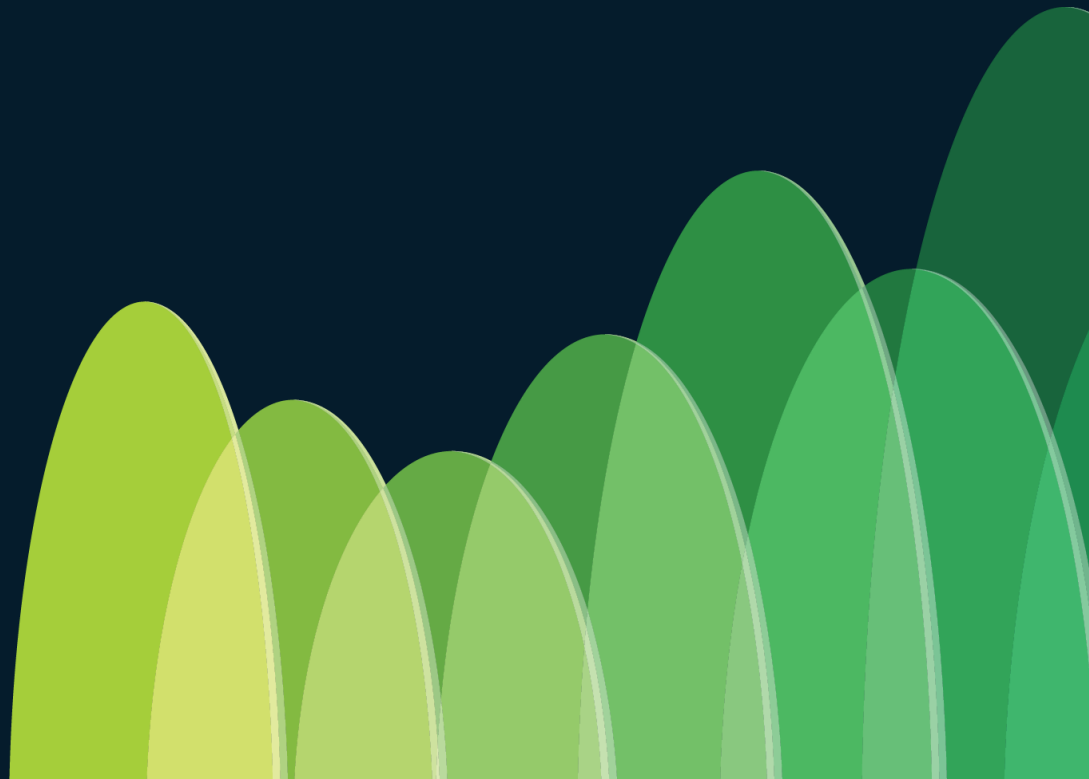
ers > maharbec > Documents > 06 - LAB > _SaC > 21 - AAC090 > SDWAN.yaml > {} sdwan > {} edge_feature_ter

```yaml
---
# SD-WAN Sites
sdwan:
  sites:
    - id: 221
      routers:
        - chassis_id: C8300-1N1S-4T2X-FDO2724M1K8
          model: C8300-1N1S-4T2X
          device_template: DT-C8300-1N1S-4T2X-SINGLECPE-Branch1
          device_variables:
            site_id: 221
            system_ip: 1.22.4.21
            system_hostname: CI83k-BER-HQ1
sdwan:
  edge_feature_templates:
    aaa_templates:
      - name: FT-CEDGE-AAA-01
        description: Local auth
        authentication_and_authorization_order:
          - local
    bgp_templates:
      - name: FT-CEDGE-BGP-VPN1-ACTIVE
        description: VPN 1 BGP
        ipv4_address_family:
          default_information_originate: true
          maximum_paths_variable: vpn1_bgp_ipv4_maximum_paths
          redistributes:
            - protocol: omp
              optional: false
              route_policy: RM-SITE-BGP-OUT-ACTIVE
          neighbors:
            - address_variable: vpn1_bgp_ipv4_neighbor1_address
              address_families:
                - family_type: ipv4-unicast
                  maximum_prefixes: 1000
              next_hop_self: false
    ntp_templates:
      - name: FT-CEDGE-NTP-01
        description: Base NTP template; no auth key; no ntp master
        servers:
          - hostname_ip: 10.49.216.10
            prefer: true
            source_interface_variable: ntp_server_source_interface
            vpn_id: 0
```

Demo content

1. Show empty SD WAN Manager
2. Explain YAML file structure and content of SDWAN Site / Templates
3. Run terraform to push templates and activate cEdge and BGP
4. Verify all is setup
   - Ping from VMs to DGW
   - Ping from VMs to WAN and hosts in Campus failed

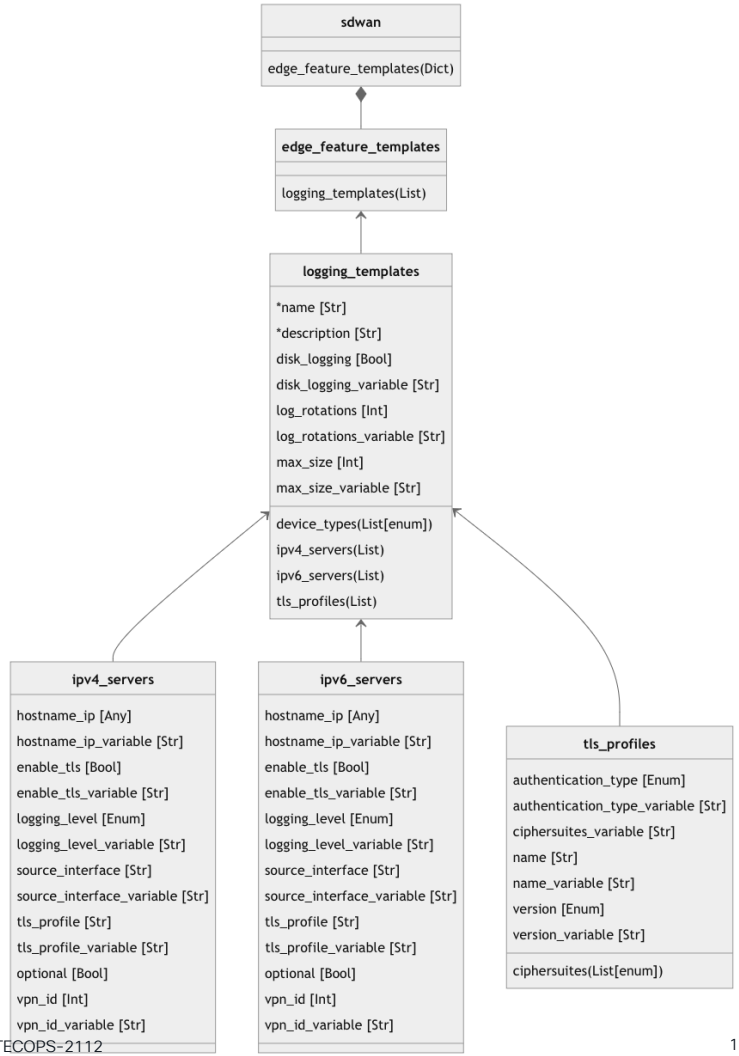
SDWAN
Data Model



cisco Live!

Data Model

- The data model describes the structure and format of the input data (desired state definition)
- A tool called yamale is used to define the schema, which is then used for syntactic validation
- Data model documentation available soom



YAML layout

- As different teams might be responsible for different parts of the infrastructure, it is of paramount importance to allow enough flexibility when defining and maintaining the SD-WAN configuration.
- The configuration can be split into multiple YAML files each for example covering a specific logical section of the configuration.
- SD-WAN as Code does not dictate a specific schema, but instead allows for full flexibilty to divide the configuration as needed.

```
---
defaults:
$ tree -L 2
.
├── data
│   ├── edge_feature_templates.yaml
│   ├── edge_device_templates.yaml
│   ├── policy_objects.yaml
│   ├── localized_policies.yaml
│   ├── centralized_policies.yaml
│   ├── emea_sites.yaml
│   ├── amer_sites.yaml
│   └── apac_sites.yaml
├── defaults
│   └── defaults.yaml
└── main.tf
```

Data Model Example
Policy Objects

Option 1: single file

```yaml
---
sdwan:
  policy_objects:
    color_lists:
      - name: internet-colors
        colors:
          - custom1
          - custom2
      - name: mpls-colors
        colors:
          - private1
          - private2
    ipv6_data_prefix_lists:
      - name: DPL-PRIVATE
        prefixes:
          - fe40:5657:6df1:d34f::/64
          - fc00::/7
```

Option 2: multiple files

```yaml
---
sdwan:
  policy_objects:
    color_lists:
      - name: internet-colors
        colors:
          - custom1
          - custom2
      - name: mpls-colors
        colors:
          - private1
          - private2
```

```yaml
---
sdwan:
  policy_objects:
    ipv6_data_prefix_lists:
      - name: DPL-PRIVATE
        prefixes:
          - fe40:5657:6df1:d34f::/64
          - fc00::/7
```

Data Model Example
Feature Template

If value not defined, then it's SD-WAN default.

```
sdwan:
  edge_feature_templates:
    ethernet_interface_templates:
      - name: FT-EDGE-WAN-TLOC1
        description: "EDGE TLOC1 with dynamic IP"
        interface_description_variable: vpn0_tloc01_if_description
        interface_name_variable: vpn0_tloc01_if_name
        ipv4_address_dhcp: true
        dhcp_distance: 1
        ipv4_nat: true
        ipv4_nat_type: interface
        shaping_rate_variable: vpn0_tloc01_shaping_rate
        shutdown_variable: vpn0_tloc01_if_shutdown
        tunnel_interface:
          allow_service_icmp: true
          color: custom1
          restrict: true
          ipsec_encapsulation: true
          ipsec_preference_variable: vpn0_tloc01_tunnel_ipsec_preference
    <….>
    bgp_templates:
      - name: FT-EDGE-BGP-VPN1-ACTIVE
        <snipped>
          neighbors:
            - address_variable: vpn1_bgp_ipv4_neighbor1_address
              address_families:
                - family_type: ipv4-unicast
                  maximum_prefixes: 1000
              next_hop_self: false
              password_variable: vpn1_bgp_ipv4_neighbor1_password
              remote_as_variable: vpn1_bgp_ipv4_neighbor1_remote_as
              shutdown_variable: vpn1_bgp_ipv4_neighbor1_shutdown
              optional: true
```

Global Value

Variable (all variables ends with "_variable"

Optional key

Data Model Example
Device Template

```
---
sdwan:
  edge_device_templates:
    - name: DT-C8000V-TEST
      description: "Test device template"
      device_model: C8000V
      aaa_template: FT-AAA-01-TEST
      bfd_template: FT-BFD-01-TEST
      system_template: FT-SYSTEM-01-TEST
      <omitted for brevity>
      localized_policy: LOCAL-POLICY-TEMPLATE-01-TEST
      vpn_0_template:
        name: FT-VPN0-01-TEST
        ethernet_interface_templates:
          - name: FT-VPN0-ETH1
          - name: FT-VPN0-ETH2
      vpn_512_template:
        name: FT-VPN512-01-TEST
      vpn_service_templates:
        - name: FT-VPN1-01-TEST
          ethernet_interface_templates:
            - name: FT-VPN0-ETH1
          ospf_template: FT-OSPF-02-TEST
```

Data Model Example
Sites

```
---
sdwan:
  sites:
    - id: 2101
      routers:
      - chassis_id: C8K-CC678D1C-8EDF-3966-4F51-ABFAB64F5ABE
        model: C8000V
        device_template: DT-C8000V-TEST
        device_variables:
          site_id: 2101
          system_ip: 10.0.2.101
          system_hostname: SITE2101-C8KV-01
          vpn0_tloc01_if_name: GigabitEthernet1
          vpn0_tloc01_if_description: INET-1
          vpn0_tloc01_if_shutdown: false
          vpn0_tloc01_shaping_rate: 1000000
          <omitted for brevity>
          vpn10_bgp_ipv4_neighbor2_address: TEMPLATE_IGNORE
          vpn10_bgp_ipv4_neighbor2_shutdown: TEMPLATE_IGNORE
          vpn10_bgp_ipv4_neighbor2_remote_as: TEMPLATE_IGNORE
          vpn10_bgp_ipv4_neighbor2_password: TEMPLATE_IGNORE
```

TEMPLATE_IGNORE

Value cane be used with optional variables

Encrypted strings, hash values

All parameters in the feature templates that include vManage-encrypted strings (values that start with $CRYPT_CLUSTER$) must be updated with encrypted strings generated in the target vManage.

Parameters with hash values (parameters that start with 6, SHA512 hash, 9, Cisco Type-9 hash, or Cisco Type-7 hash) should also be updated to use the desired clear-text value.

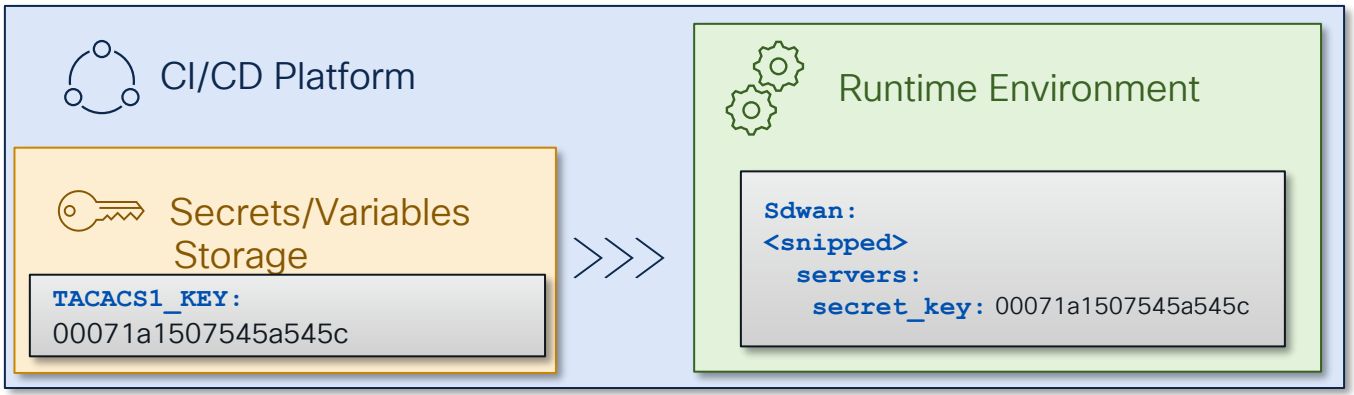
```
aaa_templates:
    - name: FT-EDGE-AAA-01
      description: Local auth
      authentication_and_authorization_order:
        - local
      users:
        - name: admin
          optional: false
          password:
$6$Al6owF//yUhkjq5D$Q3VsfDUNBkB5M0XXXYjNxpBR6MVXeOnN9LcHudt538AjvN9DRNaLP71
SYbzwzqh5KRk.gIJPYHFPCO2ozxQFC.
          privilege_level: 15
          secret: $9$xTpl9PHA7QMxIt$v7NY.c7mo0NxtIh/owLLOFQH/H1KNp0WA1UwTRMRp/k
        - name: failsafe
          optional: false
          password:
$6$.5ElETCi9kMuSY/8$C2ikiVQmGMiAXkvArECedHIwo3tPtmL/76jzhCfnQHHVZ/fgNiYQRtT1TbdQ/a
xflP2xuh/a2ef70ZVlcD2ch/
          privilege_level: 15
          secret: $9$k4TwNjUuLAB8R4$g7gbI9hC55dzHuLitRdoPqs7tr/wpF8N9lcd/yKeaBI
```

```
tacacs_server_groups:
      - name: TACACS-GROUP1
        source_interface: Loopback511
        vpn_id: 511
        servers:
          - address: 10.1.1.1
            port: 49
            timeout: 120
            key:
$CRYPT_CLUSTER$0A8/tHNPjmeM6FwSrCCU2Q==$4XN7WfJqZ7TgJWwt2Ob50A==
            secret_key: 00071a1507545a545c
          - address: 10.1.1.2
            key:
$CRYPT_CLUSTER$0A8/tHNPjmeM6FwSrCCU2Q==$4XN7WfJqZ7TgJWwt2Ob50A==
            port: 49
            secret_key: 00071a1507545a545c
```

Secrets

The configuration might contain sensitive information that should not be stored in cleartext in the configuration. One common approach to handling secrets in the context of CI/CD Platforms is by injecting sensitive values as environment variables during runtime.

```
sdwan:
  edge_feature_templates:
    aaa_templates:
      key tacacs_server_groups::
        servers:
          secret_key: !env TACACS1_KEY
```

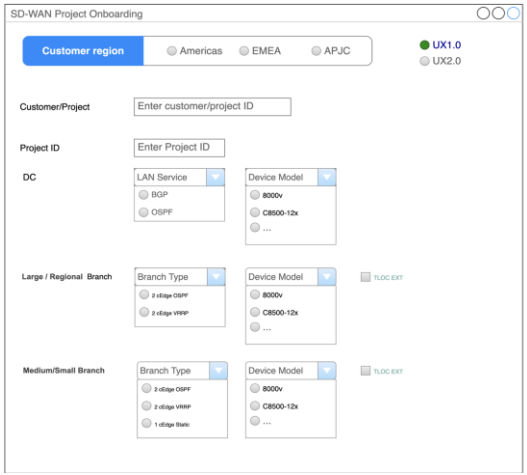


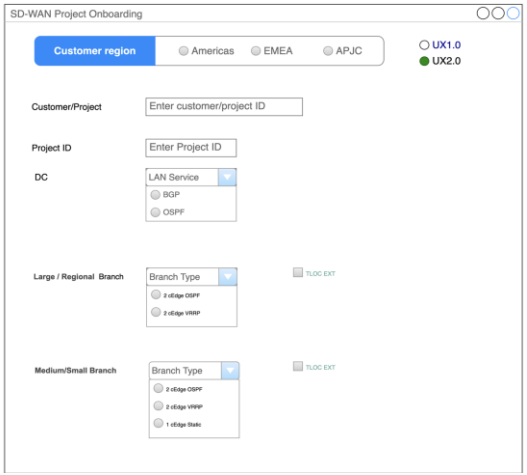
Onboarding Tool for Greenfield Deployments

- Tool available at https://ddot.cisco.com/

In Development

- Bootstrapping of “as Code” repository structures
  - Provide an easy way to get started with a customer config
  - Collect adoption statistics
  - Today is ACI as Code supported, however SDWAN will be supported soon (under development)





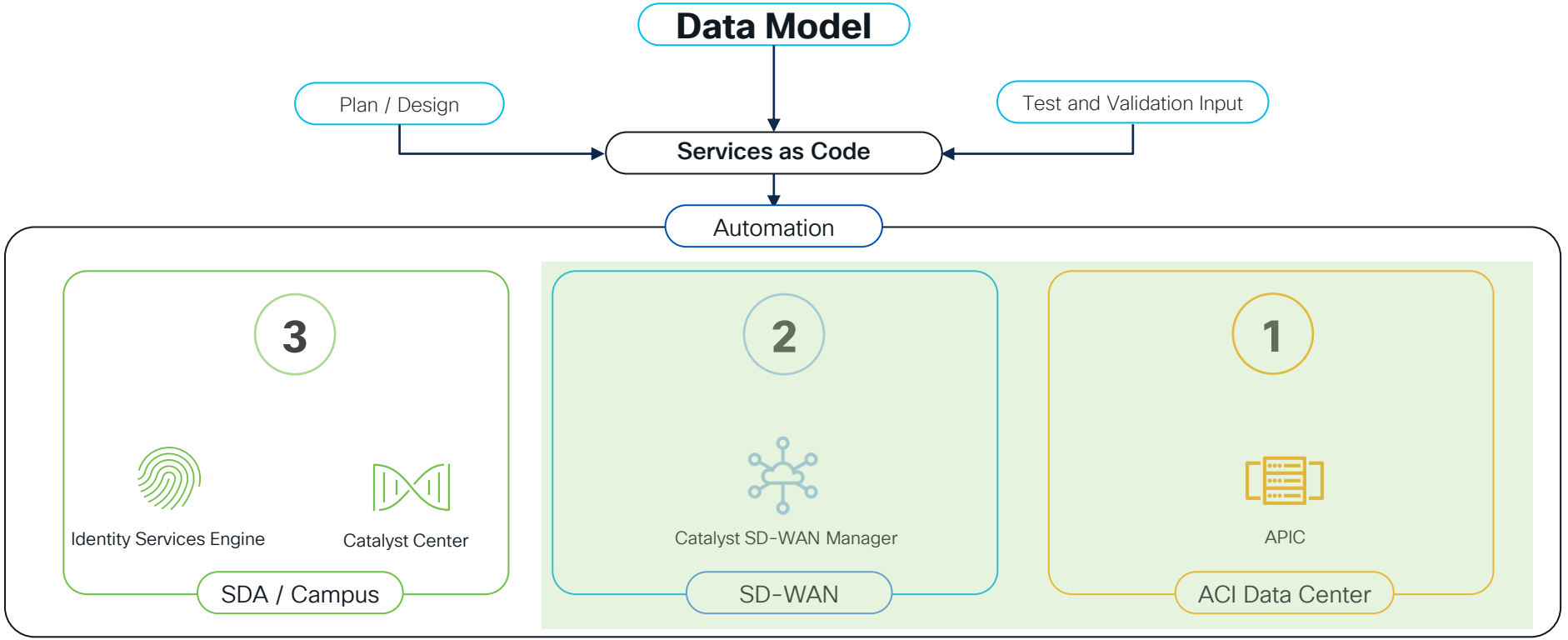

```
`-- data
    |-- centralized_policies.nac.yaml
    |-- edge_device_templates_Branch_T1.nac.yaml
    |-- edge_device_templates_Branch_T2.nac.yaml
    |-- edge_device_templates_Branch_T3.nac.yaml
    |-- edge_device_templates_DC.nac.yaml
    |-- edge_feature_templates.nac.yaml
    |-- localized_policies.nac.yaml
    |-- policy_objects.nac.yaml
    |-- sites_Branch1.nac.yaml
    |-- sites_Branch2.nac.yaml
    `-- sites_DC.nac.yaml
```

Automation “apply”

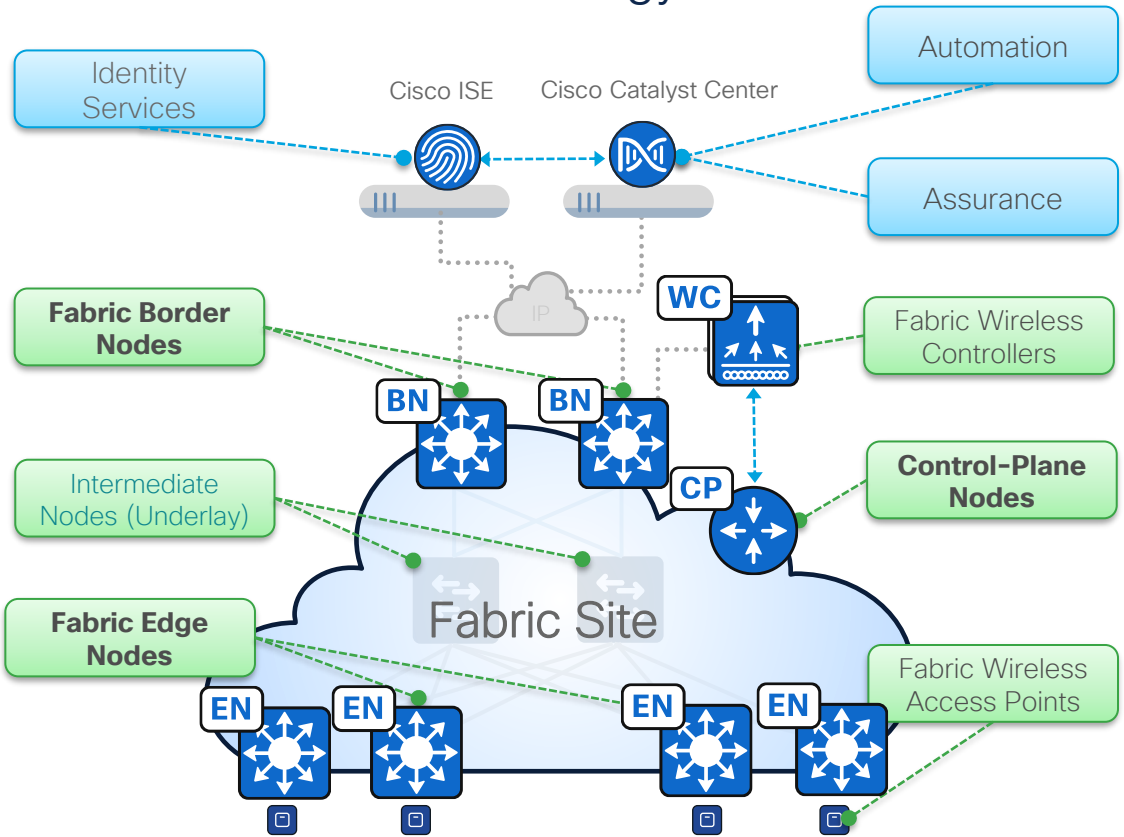
Catalyst Center (for SDA) as Code

cisco Live!

Catalyst Center as Code (for e.g. SDA)



Cisco SD-Access
Fabric Roles & Terminology



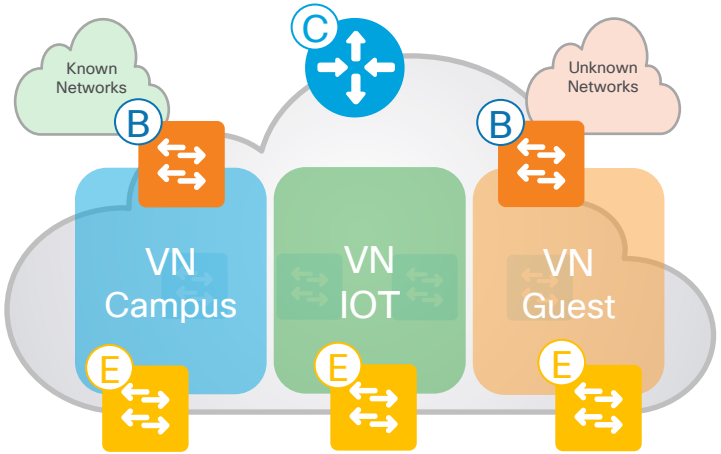
- **Network Automation** – Simple GUI and APIs for intent-based Automation of wired and wireless fabric devices
- **Network Assurance** – Data Collectors analyze Endpoint to Application flows and monitor fabric device status
- **Identity Services** – NAC & ID Services (e.g. ISE) for dynamic Endpoint to Group mapping and Policy definition

- **Control-Plane Nodes** – Map System that manages Endpoint to Device relationships
- **Fabric Border Nodes** – A fabric device (e.g. Core) that connects External L3 network(s) to the SD-Access fabric
- **Fabric Edge Nodes** – A fabric device (e.g. Access or Distribution) that connects Wired Endpoints to the SD-Access fabric
- **Fabric Wireless Controller** – A fabric device (WLC) that connects Fabric APs and Wireless Endpoints to the SD-Access fabric

Virtual Network – **Macro** Segmentation

VN or VPN or VRF are maintaining a separate Routing table for each instance

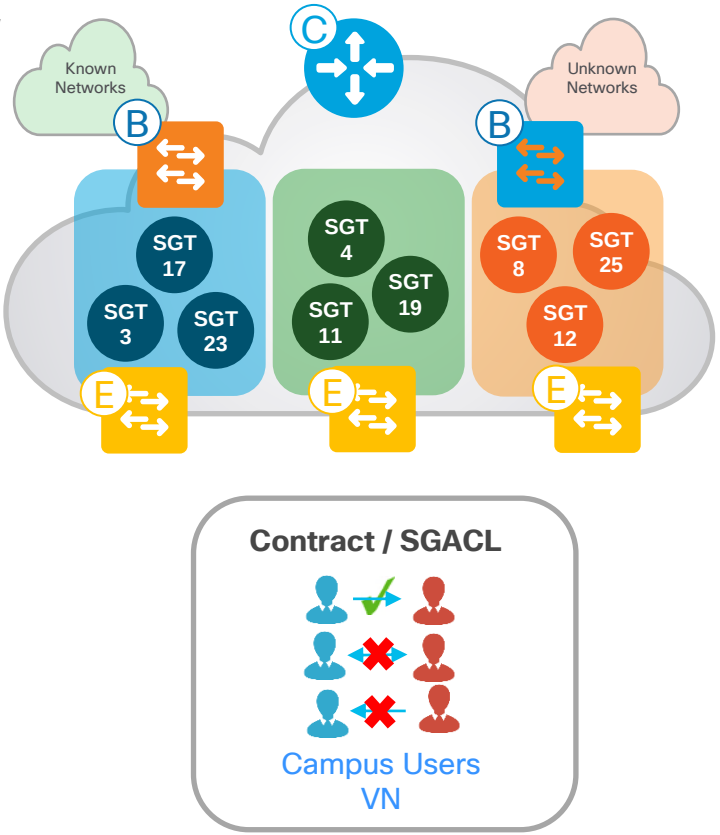
- SDA Control-Plane (LISP) uses Instance ID to maintain separate VRF topologies (“Default” VRF is Instance ID “4098”)
- SDWAN uses VPN ID
- Endpoint ID and / or prefixes are routed and advertised within a Virtual Network
- VNs = VRFs = VPN maintain complete isolation within a VRF
- Default Policy:
  within VN allow any to any
  between VNs no communication (needs Fusion dev)



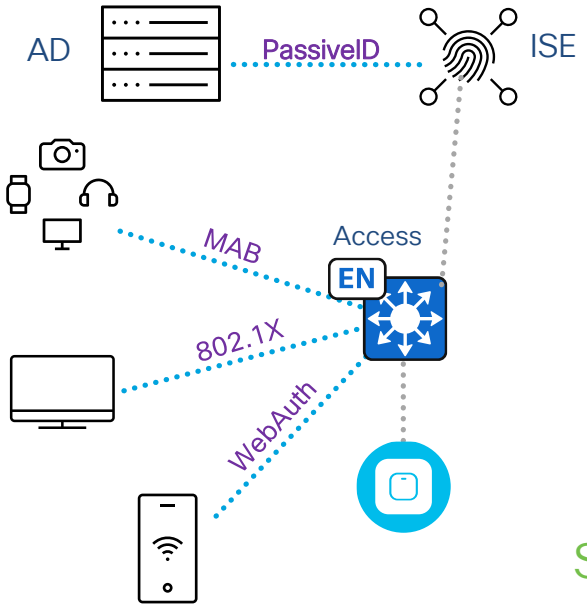
Security Groups – **Micro** Segmentation

Security Group Tag (SGT) is a logical policy object to "group" Users and/or Devices

- Users / Nodes use "SGTs" to IP and assign a unique Security Group Tag (SGT) to Endpoints
- SDA Edge Nodes add a SGT to the Fabric encapsulation
- cEdges (if enabled) transport SGT end to end
- SGTs are used to manage address-independent "Group-Based Policies"
- Edge or Border Nodes use SGT to enforce local Security Group ACLs (SGACLs)
- SGTs with SGACL can permit / deny traffic within a VN



Security Group Tags (SGT) Classification



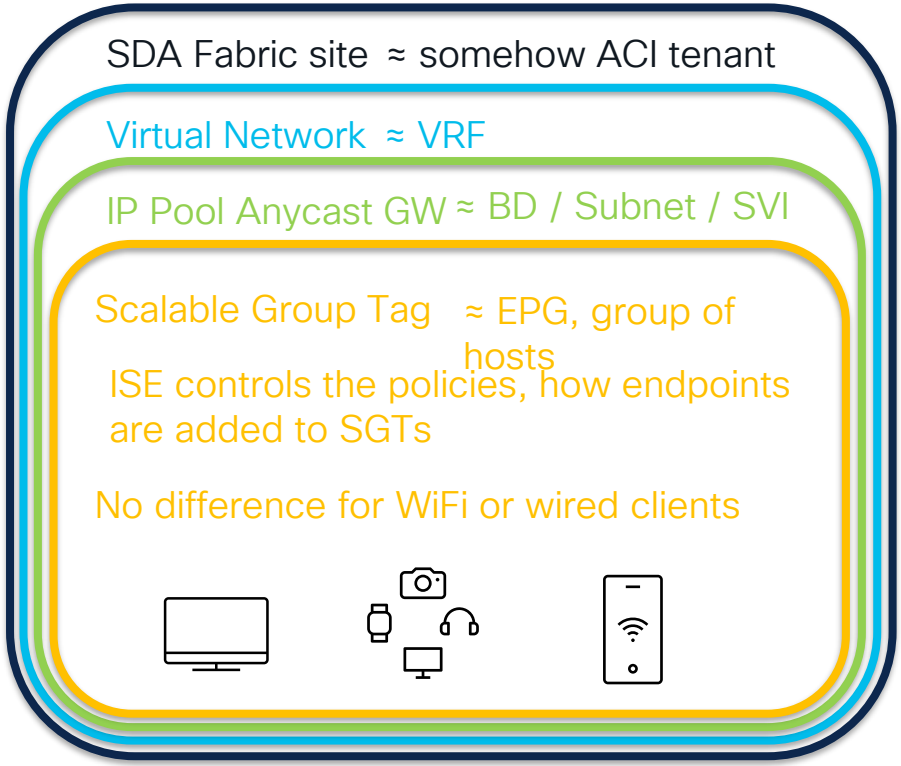
Classify

- MAB = Mac Authentication Bypass
- Web Authentication
- 802.1x by eg supplicant (Cert, User etc)
- Static binding

SGT Binding

- By Port
- By VLAN
- By IP Subnet
- By IP Address

The SDA Policy Model

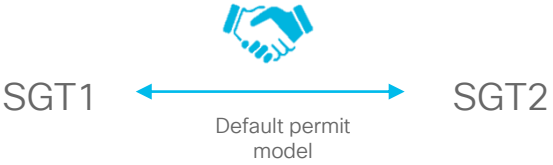


ISE policies can be based on 802.1x with certificates, MAB, posture, ...

SGT have two jobs:

(1) Group the hosts
(2) Enforce policies between hosts

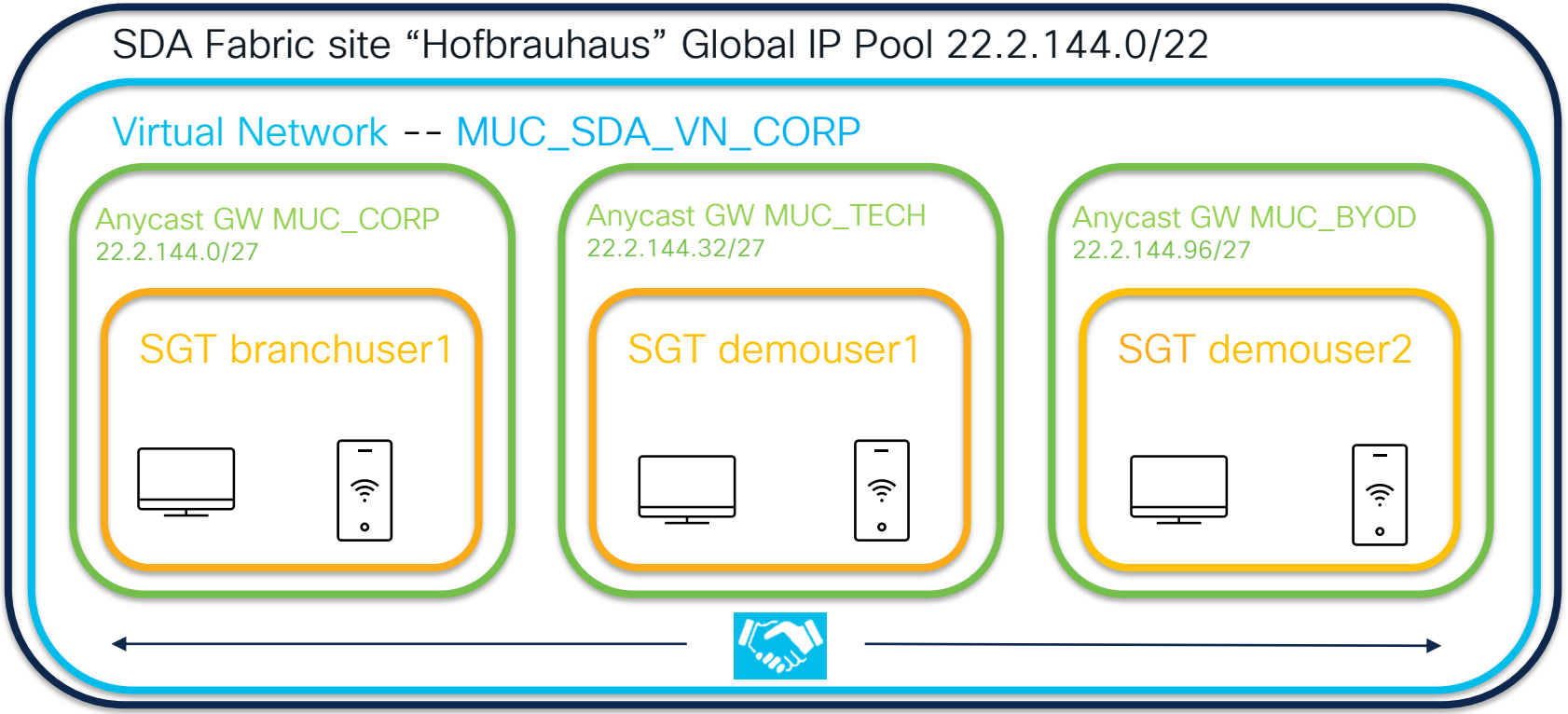
ISE policies define as well if and how SGTs can communicate to each other



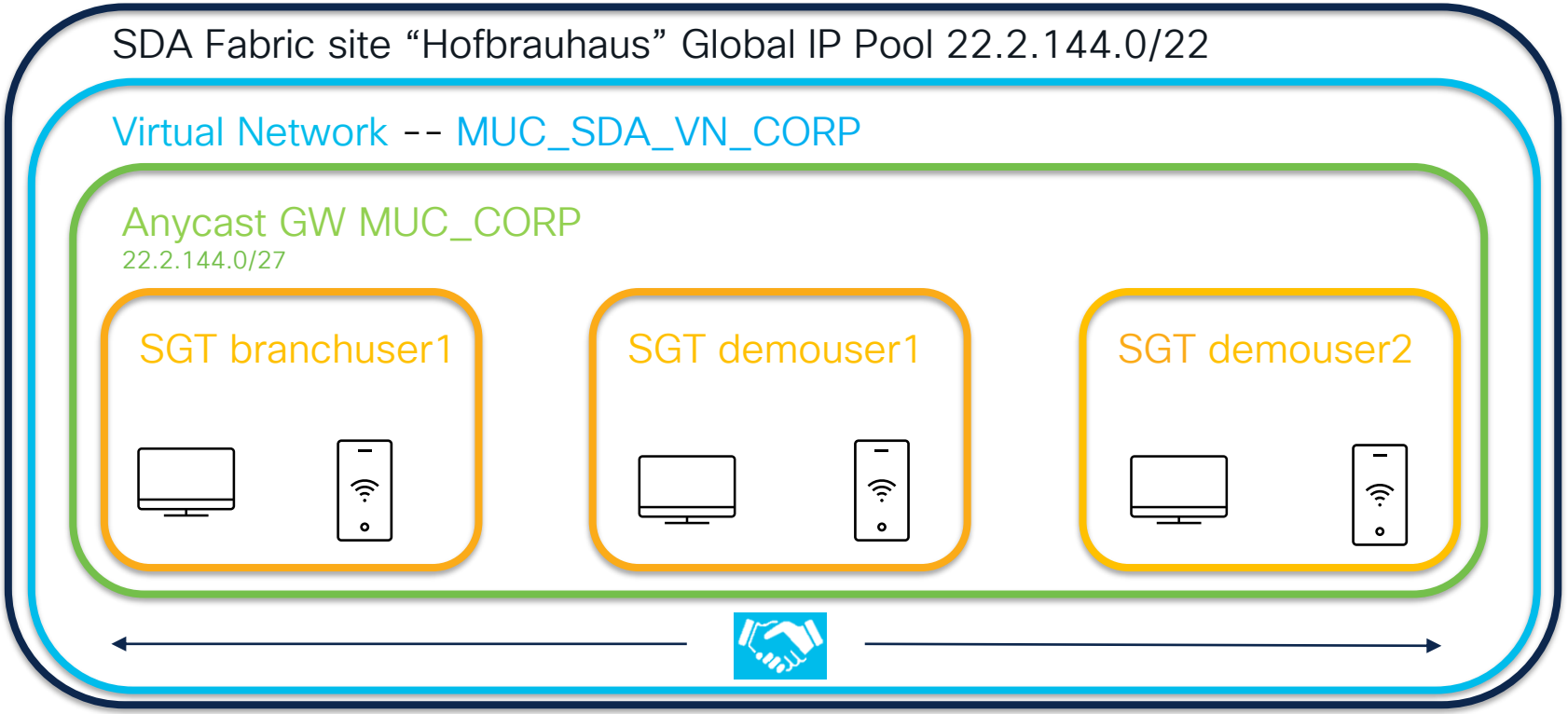
SGT policies ≈ More Intelligent Access Lists no Firewall

* Preferred group or vzAny Contract achieves the same outcome

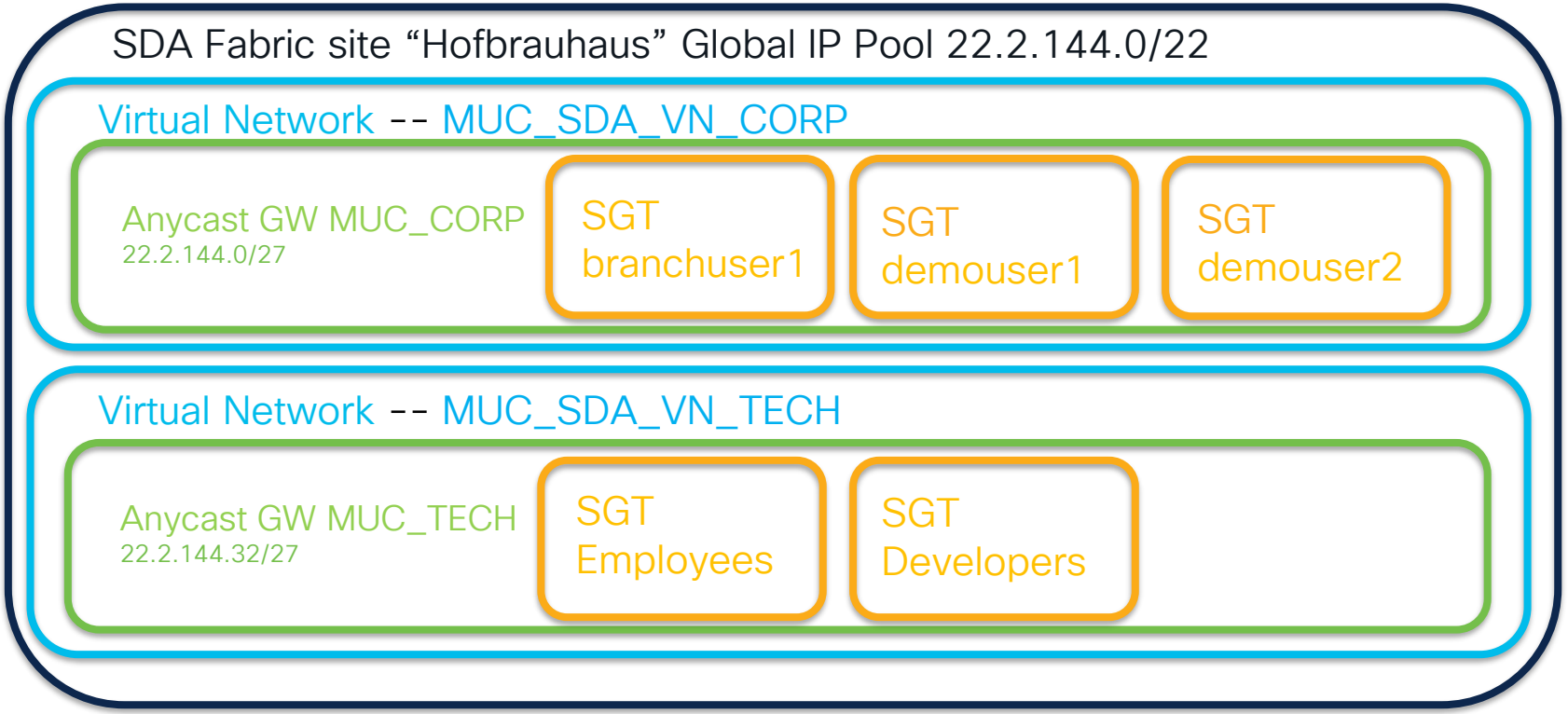
The SDA Policy Model – Starting off with SDA



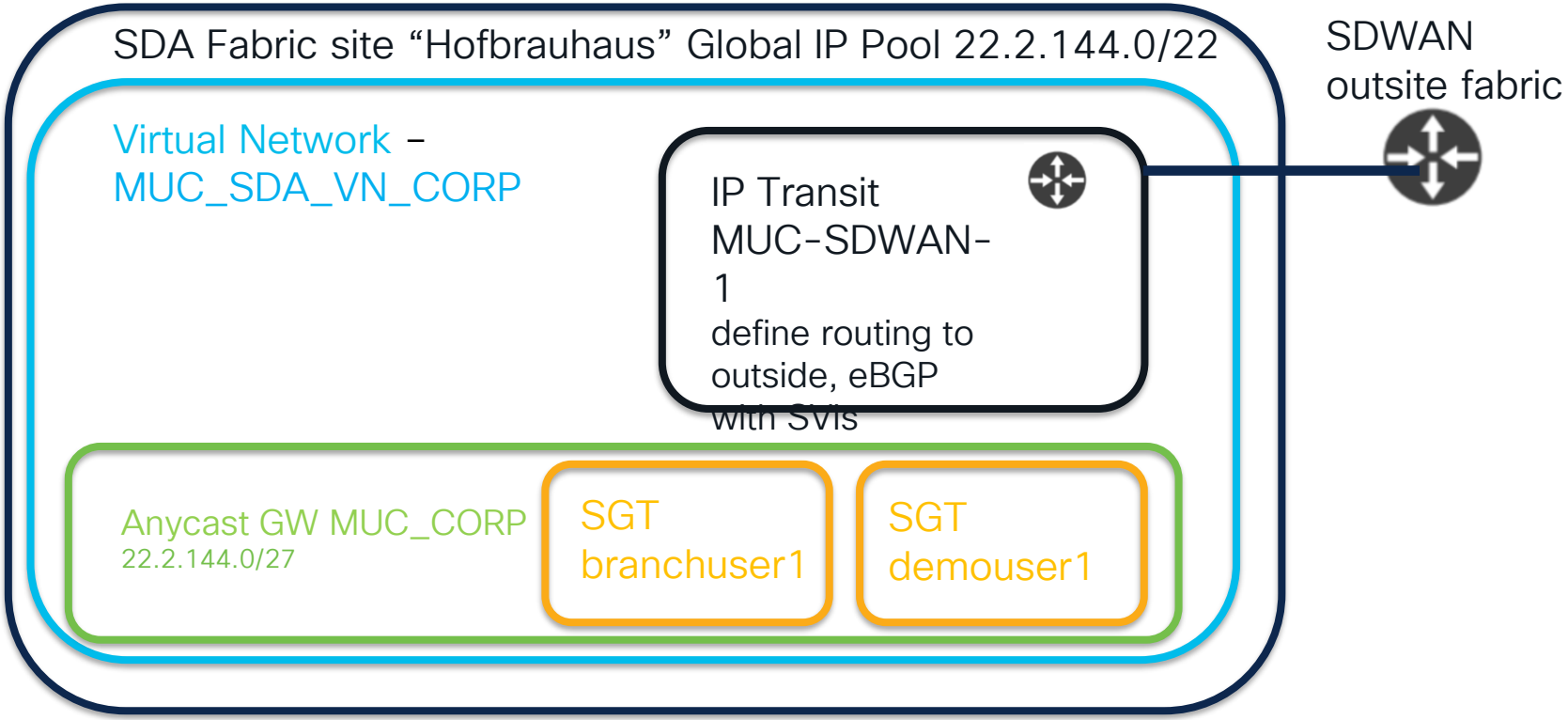
The SDA Policy Model – One IP pool multiple SGTs



The SDA Policy Model – Multiple VNs

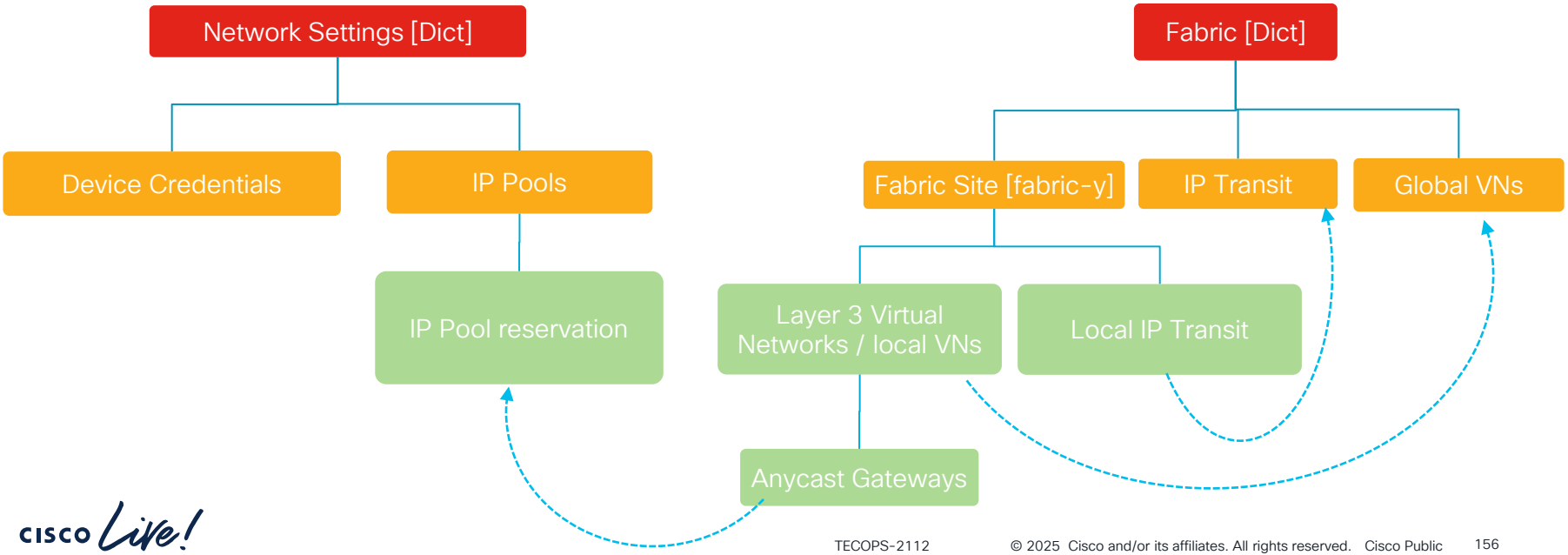


The SDA Policy Model – IP Transit

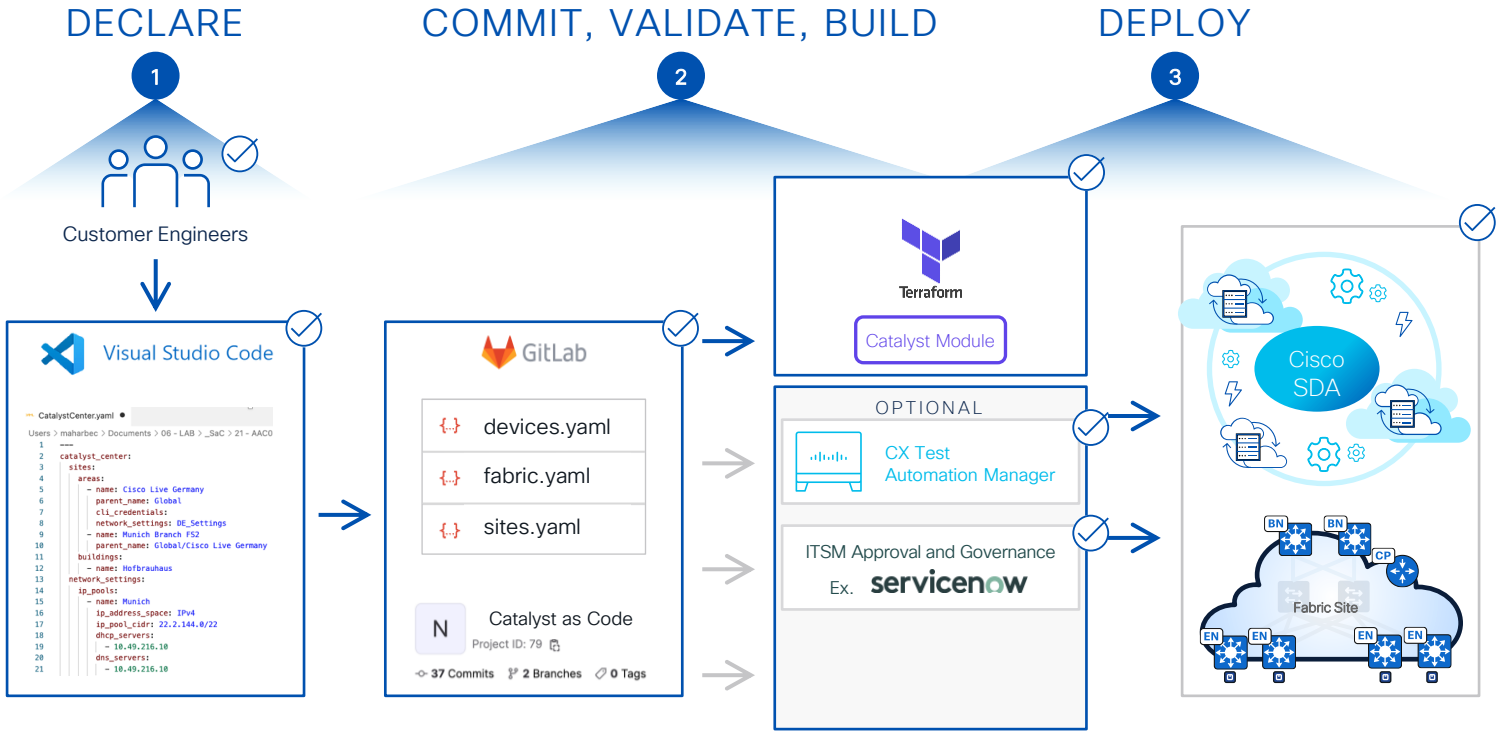


The key element is Data and abstraction
SDA Fabric

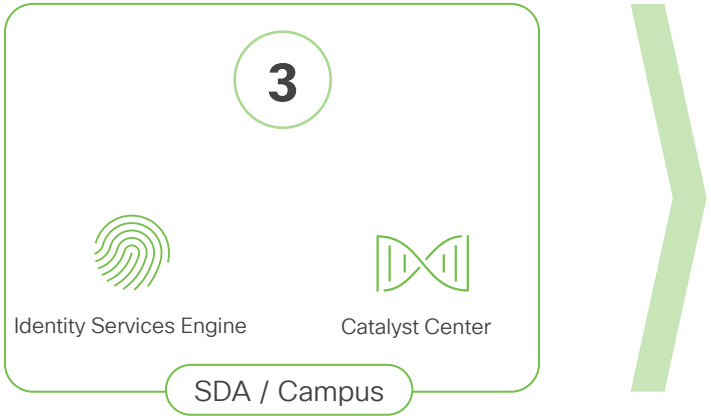
- Every Controller or every device configuration follows a data structure
- Objects having dependencies to each other, as well sequence to create is important
  - Catalyst Center as Code will take care



Services as Code for Catalyst Center flow option



DEMO Time



CatalystCenter.yaml

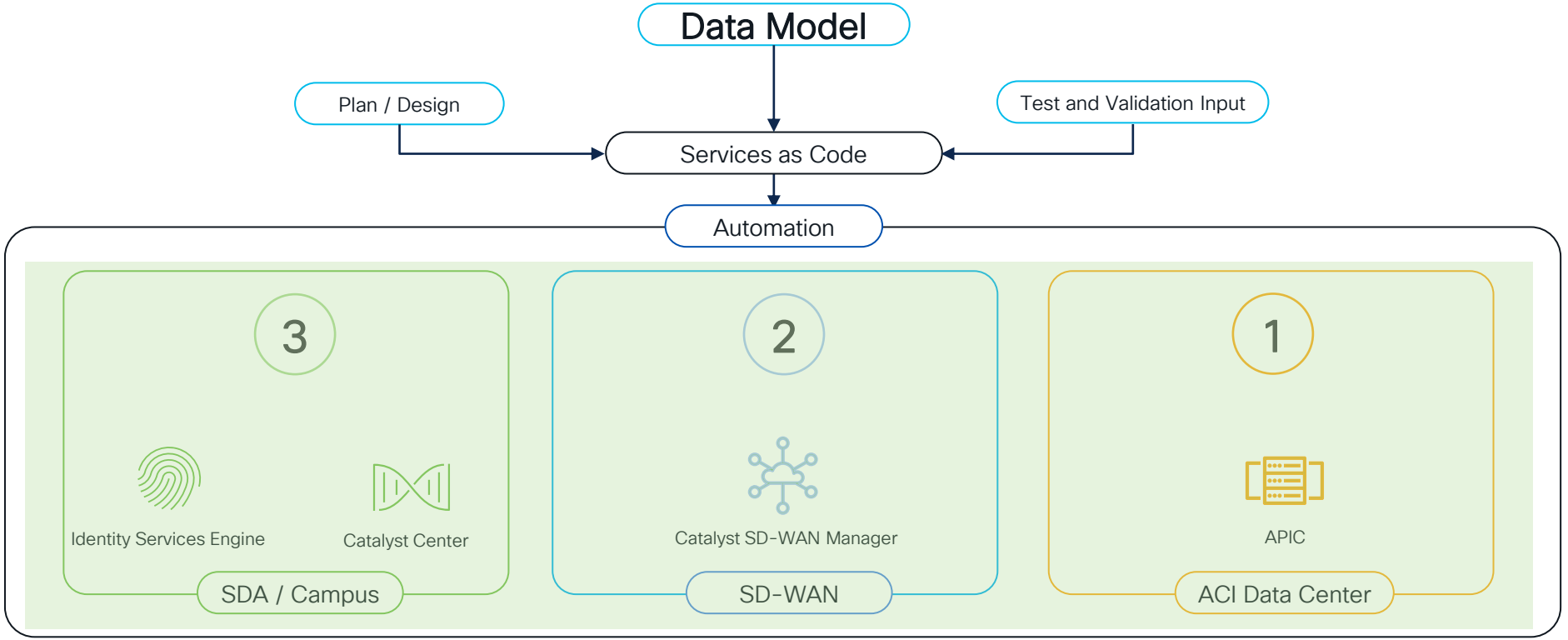
Users > maharbec > Documents > 06 - LAB > _SaC > 21 - AAC090 > CatalystCenter.yaml > {} catalyst_center > {}

```yaml
---
catalyst_center:
  sites:
    areas:
      - name: Cisco Live Germany
        parent_name: Global
        cli_credentials:
        network_settings: DE_Settings
      - name: Munich Branch FS2
        parent_name: Global/Cisco Live Germany
    buildings:
      - name: Hofbrauhaus
  network_settings:
    ip_pools:
      - name: Munich
        ip_address_space: IPv4
        ip_pool_cidr: 22.2.144.0/22
        dhcp_servers:
          - 10.49.216.10
        dns_servers:
          - 10.49.216.10
        ip_pools_reservations:
          - name: MUC_CORP
            prefix_length: 27
            subnet: 22.2.144.0
            type: Generic
  fabric:
    transits:
      - name: TRANSIT-MUC-SDWAN1
        type: ip_transit
        routing_protocol_name: BGP
        autonomous_system_number: 64927
    fabric_sites:
      - name: Global/Cisco Live Germany/Munich Branch FS2/Hofbrauhaus
        authentication_template_name: Closed Authentication
        fabric_type: FABRIC_SITE
        l3_virtual_networks:
          - name: MUC_SDA_VN_CORP
        anycast_gateways:
          - name: MUC_CORP
            vlan_name: VLAN_CORP
            vlan_id: 2021
            traffic_type: DATA
            wireless_pool: false
```

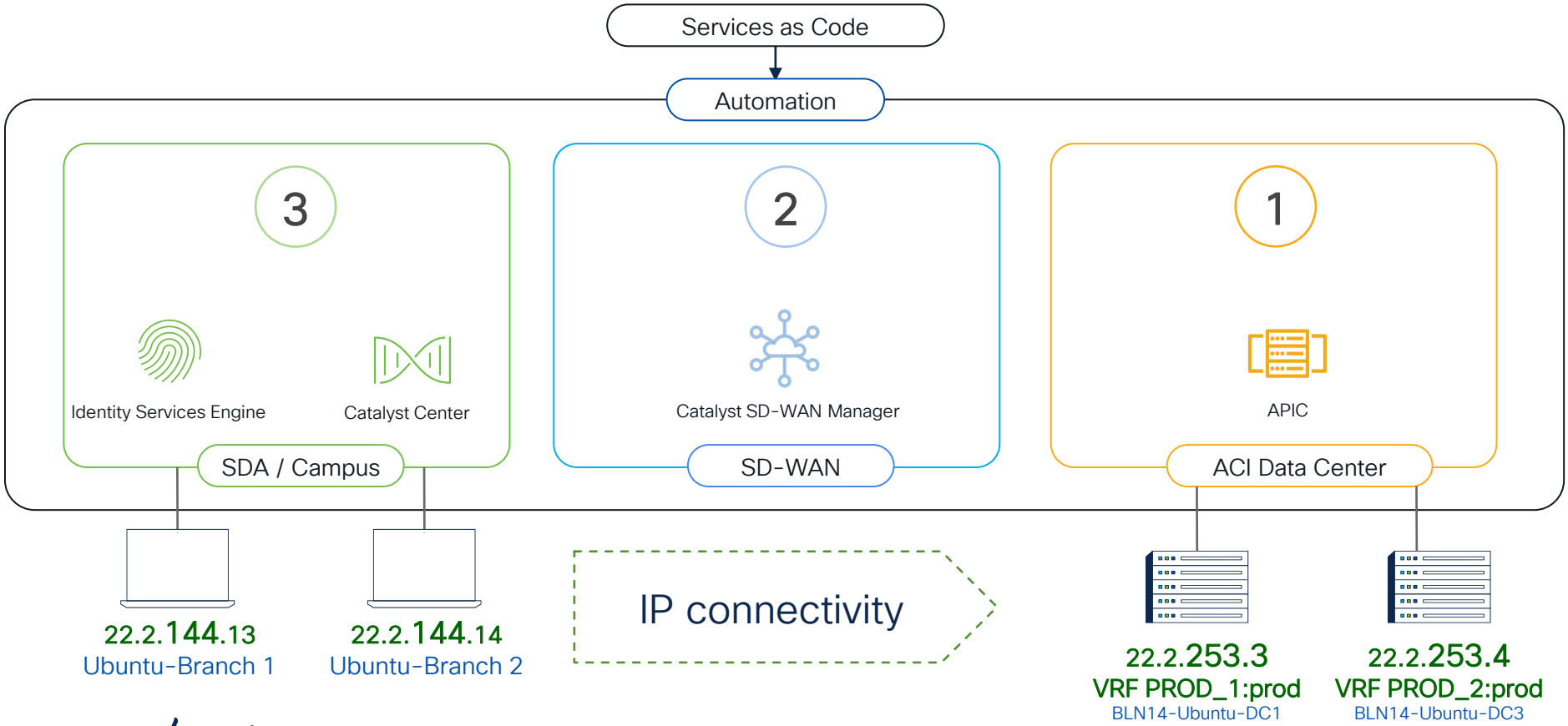
Demo content

1. Show empty Catalyst Center no Munich no IP etc
2. Explain YAML file structure and content of SDA Site / Templates
3. Run terraform to push Design Hierarchy, IP Pools, etc. templates – SDA Fabric Settings
4. activate BN, CP, EN (FIAB), activate Closed Authentication
5. Verify all is setup
   - Ping from VMs to DGW
   - Ping from VMs to WAN and hosts in Campus

Demo Time → SDA as Code



Demo Time → Outcome



Compare the domains

	SDA	SD WAN	ACI
Management	Cisco Catalyst Center	Cisco SDWAN Manager	APIC
Control Plane	LISP	vSmart - OMP	Coop - BGP
Underlay	Based on RLOC (GRT)	Based on TLOC (VN 0)	ISIS - VTEP
Data Plane	VXLAN	IPSec / MPLS	VXLAN
Macro Segment	VN Infra VN (GRT) + User VN	VPN VN 512 OOB + VN #	VRF + Tenant
Micro Segment	SGT Security Group Tag	Carries SGT	EPG / ESG End Point Groups

Compare the domains

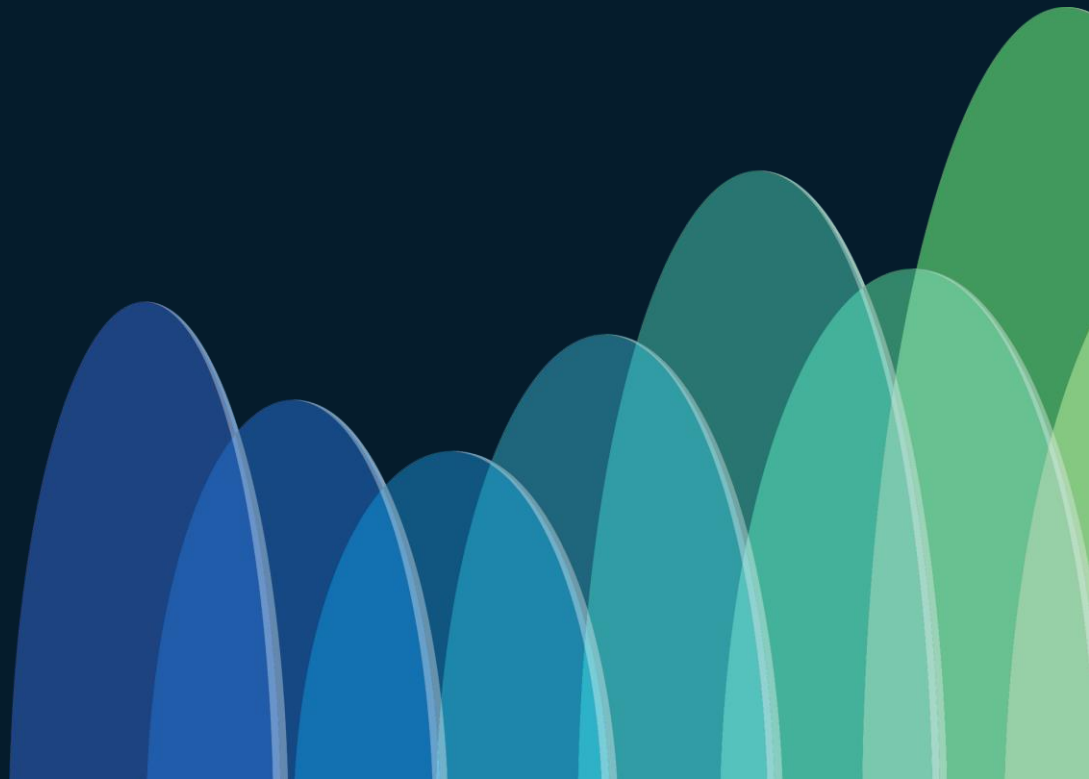
	SDA	SD WAN	ACI
Management	Cisco Catalyst Center	Cisco SDWAN Manager	APIC

Service as Code handles the details; focus on the outcomes.

”Increase Efficiency with Service as Code for scalable, repeatable infrastructure automation."

...CICD

Next Steps in Automation

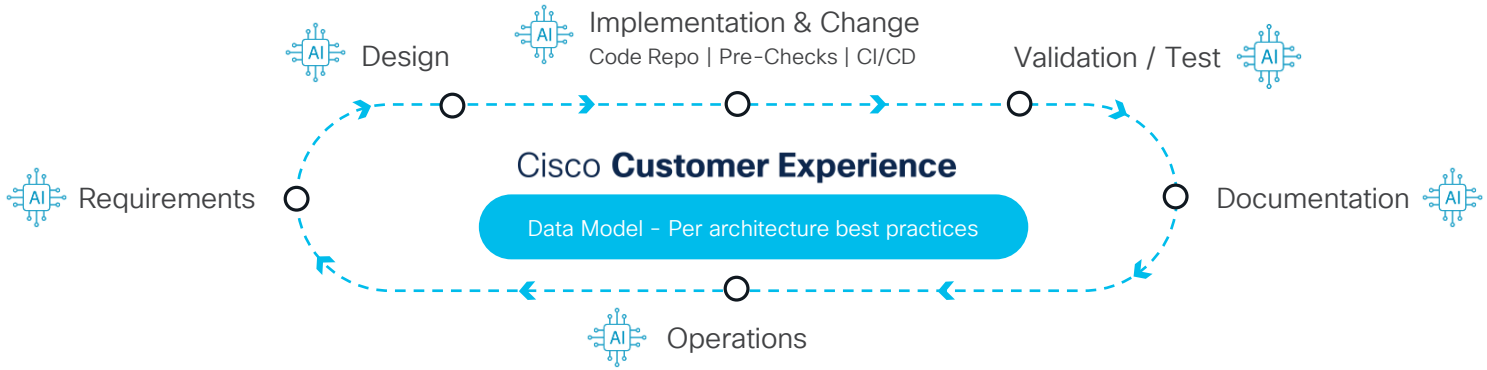


cisco Live!

Ups ...

- Didn't we mention IPv6 for VMs in Tenant PROD_1 and epg_one?
  - Maybe routing to outside for IPv6 would be helpful as well ...
  - And IPv6 policies should be the same then IPv4 ...
- Let's check our blue-prints, adopt the repo and use best-practices for git and let the CI/CD do the work ...
- Verify IPv6 connection

Network as Code



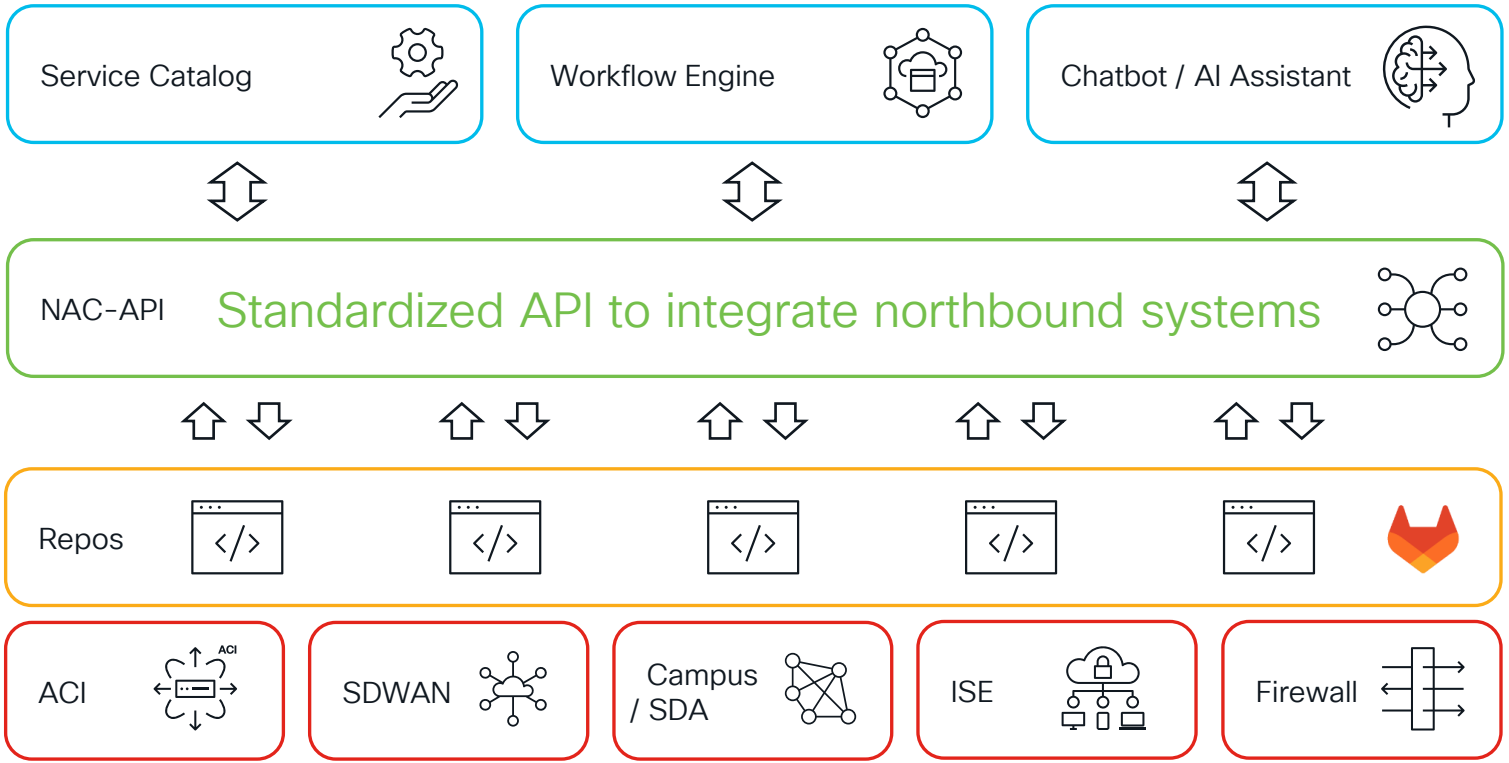
Easy consumption of many architectures in the same way

Digital Infrastructure interaction – API

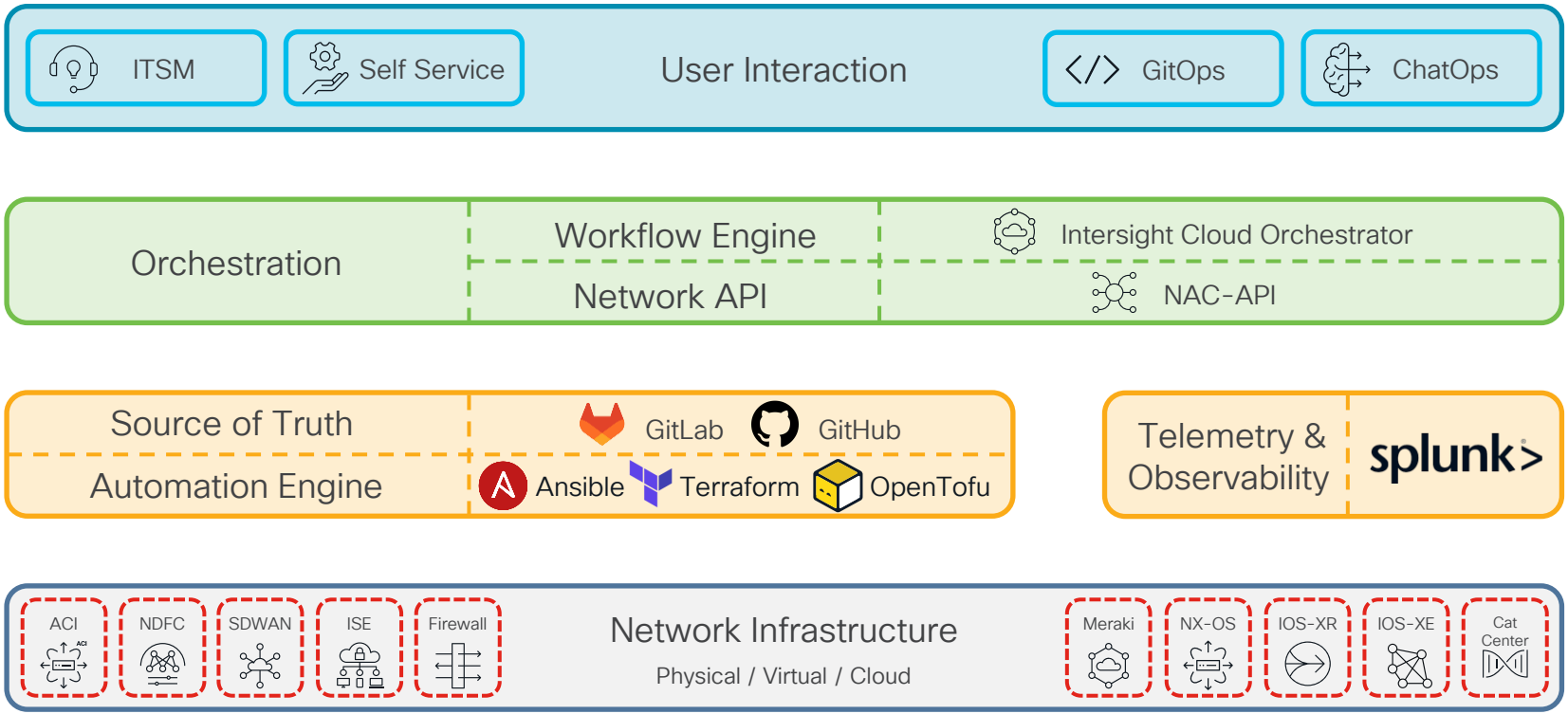
ACI | NDFC | SDWAN | ISE | Firewall | Meraki | Catalyst Center | NX-OS | IOS-XR | IOS-XE | Splunk | Thousand Eyes | 3rd parties

Shown in this presentation

Services as Code – Cross Domain Automation



Cross domain more detailed



Services as Code value insights (1)

My network as designed doesn't match reality and all networks look different

Network changes consume too many resources with too many errors

I spend more of my time reacting to network needs

The date model ensure constancy across your IT

Network changes are automated and simplified to release resources and avoid errors

Network / IT needs are abstracted and simplified

Services as Code value insights (2)

“

I’m questioning network and security compliance

“

I’m challenged to apply automation

“

Skill shortages and low innovation

“

The services as code journey ensures compliance and enables easy audits

“

Cisco enables your automation journey for and with you!

“

Leveraging the power of NetDevOps let you enhance your skills and innovate

Services as Code benefits

Version control

Configuration files can be versioned,

- track changes,
- collaborate,
- roll back to previous versions if needed.

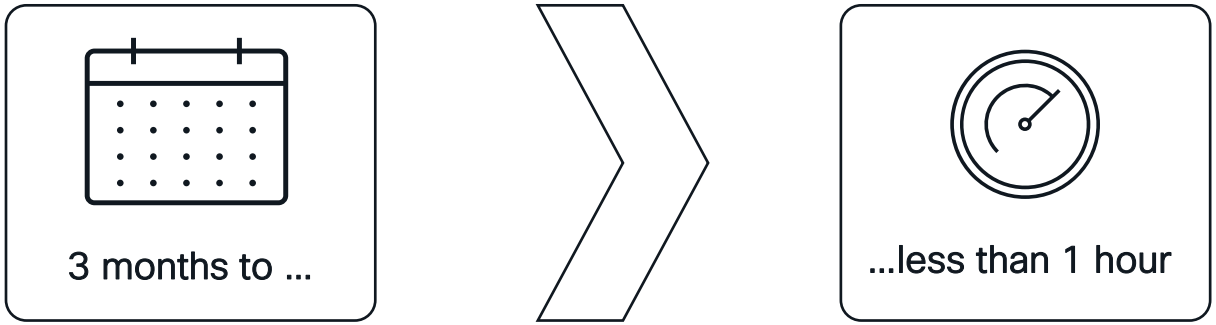
Automation

- Code-based configuration allows for automation of provisioning, deployment, and scaling processes.
- Infrastructure can be created or modified programmatically, making it easier to manage complex and dynamic environments.

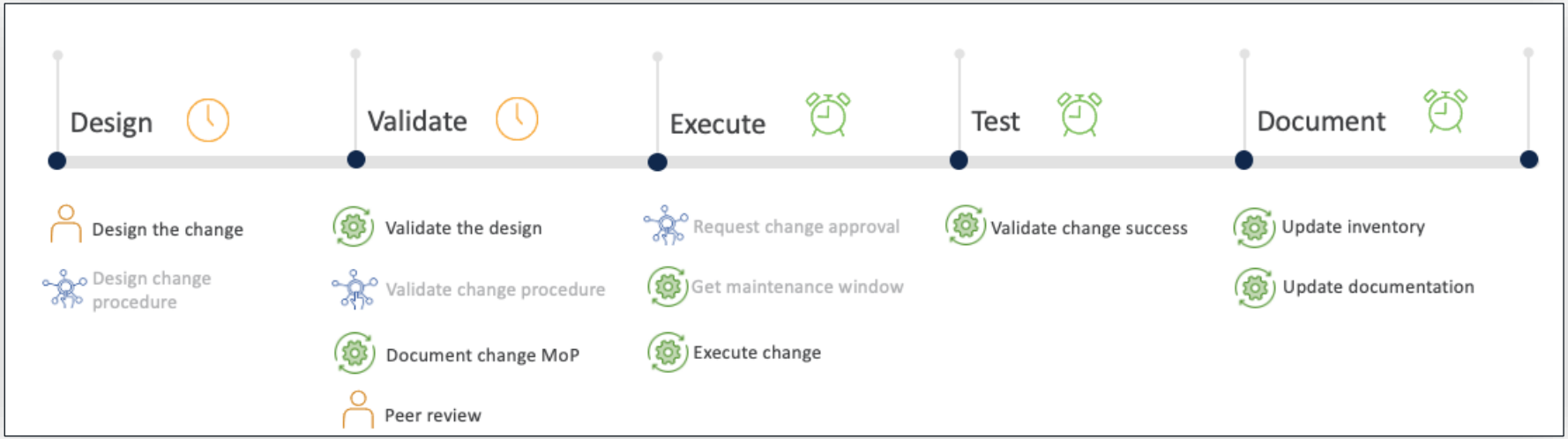
Testing and validation

- Configuration code can be tested, validated, and integrated into continuous integration and continuous deployment (CI/CD) pipelines,
- improving the reliability and quality of infrastructure changes.

Customer Value Realization



Digitized Implementation and Operation

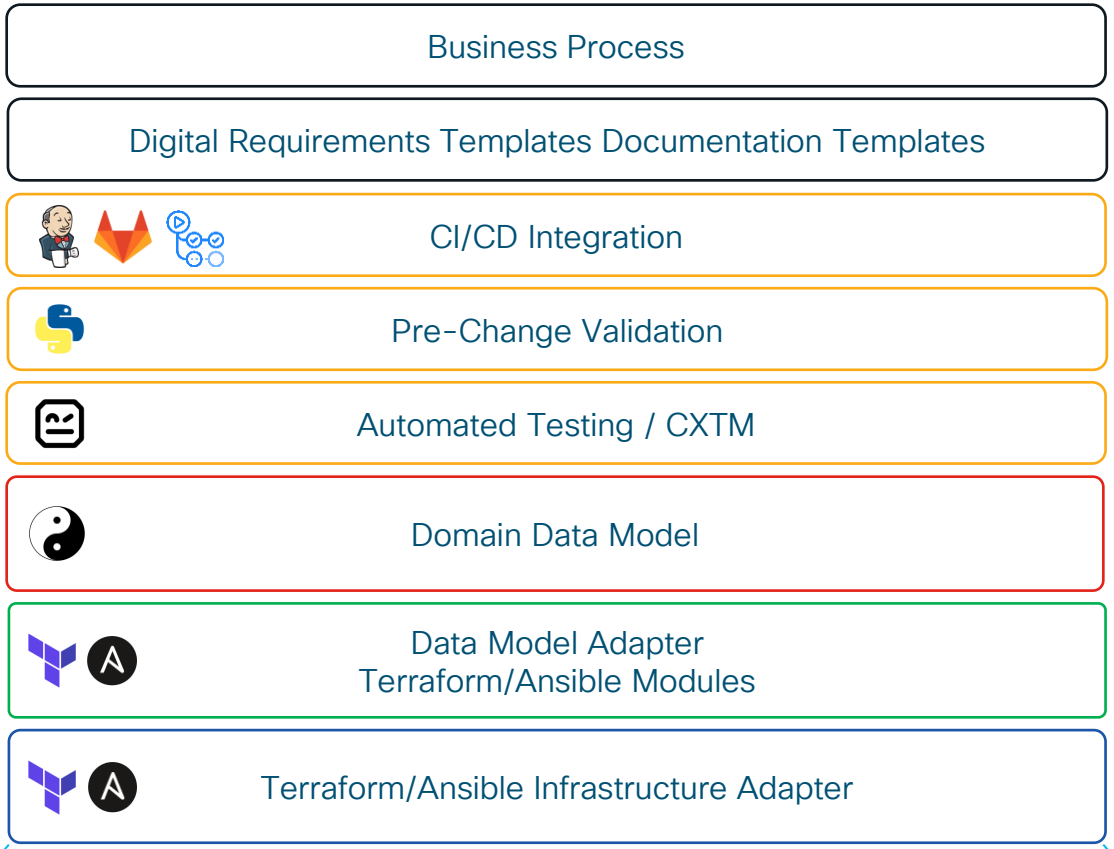


X as Code & Associated Services

Provides direct support, which streamlines open-source feedback and simplifies updates as YOU don't need to manage/copy source-code

Cisco Services team provided best practices & highly recommended add-ons

Cisco Services team provides environment setup, direct support and knowledge transfers to reduce typical open-source ramp up time and installation efforts by 80%



Summary and Conclusion

You can do everything yourself with infinite time and money

Unique design

Long build time

Questionable outcome

Extensive skillset

No warranty

Focus on build

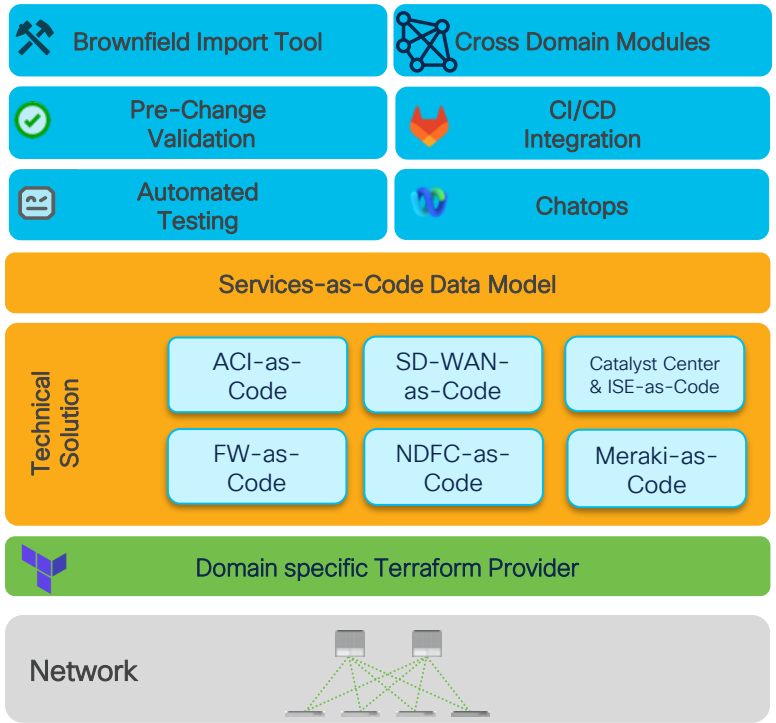
No anticipated costs

Service as Code to let you focus

You can do everything yourself with infinite time and money

Unique design	⇦ ⇨	Proven design
Long build time		Time2Market
Questionable outcome	⇦ ⇨	Turn-key service
Extensive skillset		CX lowers skills barrier
No warranty	⇦ ⇨	Cisco CX support
Focus on build		Focus on use
No anticipated costs	⇦ ⇨	Cost efficiency & control

Building Blocks for Service-as-Code



CX Best Practices documented in scripts and templates and available via Cisco CX Service-as-Code offer. These are the key differentiator for Cisco's automation services. Available via:

- **AS-T for solution deployment**, best practises and on-boarding
- **LCS for continues support** operation, TACSW Lifecycle Management and support for complete tool chain

Solution Specific data model and specific tools, owned by CXPM.

Automation library maintained by Cisco and available as open-source. These are **available and customers are already using them** to build custom automation solutions.

Services as Code Cisco Lifecycle Services

With Services as Code, enterprises can define elements of their network infrastructure as software. To seamlessly operate and optimize the network, the software can be versioned and managed at scale to expedite provisioning, configuration, testing, and deployment. Benefits include:

- Readiness assessment
- People, process, and solutions enablement
- Solution set-up and continuous integration
- Library of automation use cases and test scripts enabling frequent deployment of changes
- Quarterly Business Review reports
- Ongoing 24x7 technical support

Shifting your infrastructure and operations strategies to focus on driving business outcomes with automation.

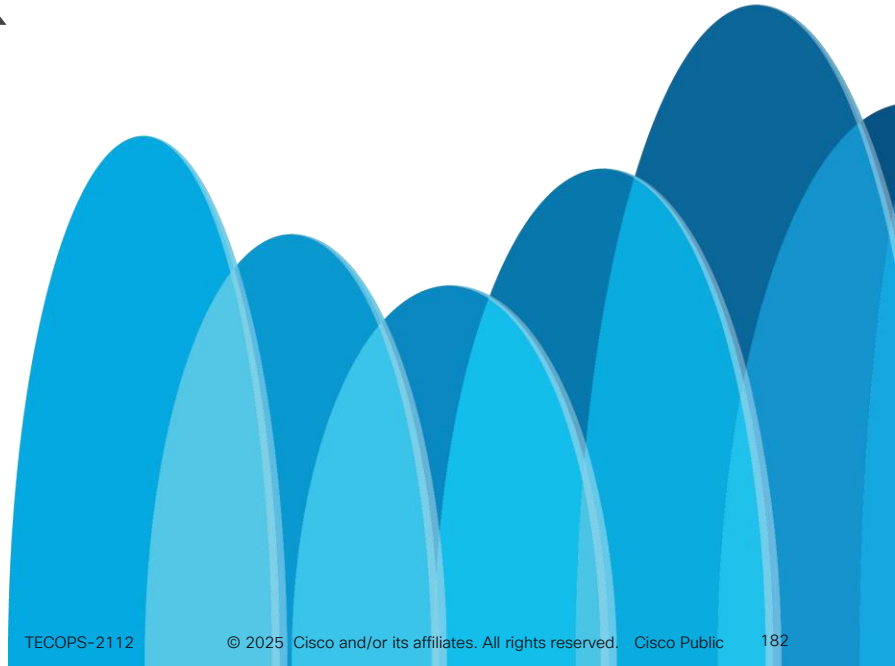
”Increase Efficiency with Service as Code for scalable, repeatable infrastructure automation."

My volunteering (hobby) with and for sustainability

If you did not like the session, send me a message and I will respond within a week...

...and send you back a session I did not like.

3 more slide pls !!!



Action items for you after this session

Relax its sounds more complex than it is ☺ Therefore:

- Download ACI as Code or SDWAN as Code and start enjoying
  - Play with the Data Model
  - Apply against your LAB or simulator
  - Send us feedback

References

- ACI/Nexus-as-Code
  https://cisco.com/go/nexusascode
- Demo Repository
  https://github.com/netascode/BRKDCN-2673-Demo
- ACI Terraform Provider
  https://registry.terraform.io/providers/CiscoDevNet/aci/latest
- Pre-Change Validation Tool
  https://github.com/netascode/iac-validate
- Test Automation Tool
  https://github.com/netascode/iac-test
- NX-OS, IOS-XE, IOS-XR Terraform Providers
  https://registry.terraform.io/search/providers?q=netascode
- If you want to have more details, please contact us via session Webex room

- Public:
  - iac-validate: Used for pre-deployment validation
  - terraform-provider-sdwan: Terraform provider for SD-WAN
  - terraform-sdwan-nac-sdwan: Terraform modules for SD-WAN as a Code

Next Steps

1. Join us in CX WoS for discussion
2. Try out CX as Code approach (GIT)
3. Ask us for a detailed workshop
4. Codify and Data-fy your IT

As Angus said:

Have a Drink as Code on me

Webex App

Questions?

Use the Webex app to chat with the speaker after the session

How

1. Find this session in the Cisco Events mobile app
2. Click “Join the Discussion”
3. Install the Webex app or go directly to the Webex space
4. Enter messages/questions in the Webex space

Webex spaces will be moderated by the speaker until February 28, 2025.

Fill Out Your Session Surveys

Participants who fill out a minimum of 4 session surveys and the overall event survey will get a unique Cisco Live t-shirt.

(from 11:30 on Thursday, while supplies last)

All surveys can be taken in the Cisco Events mobile app or by logging in to the Session Catalog and clicking the ‘Participant Dashboard’

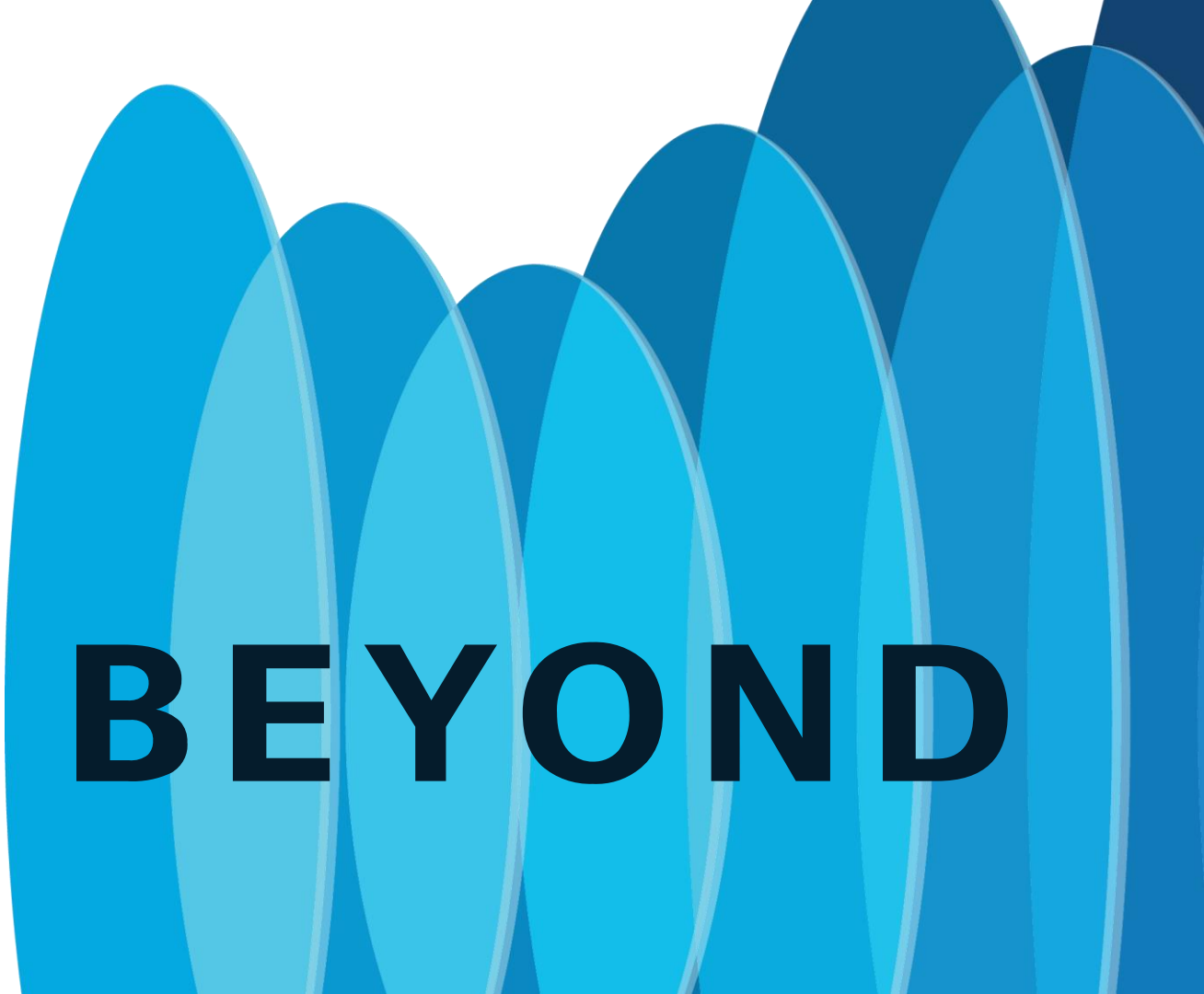
Content Catalog

Continue your education

- Visit the Cisco Showcase for related demos
- Book your one-on-one Meet the Engineer meeting
- Attend the interactive education with DevNet, Capture the Flag, and Walk-in Labs
- Visit the On-Demand Library for more sessions at ciscolive.com/on-demand. Sessions from this event will be available from March 3.


cisco
Thank you
cisco Live!

cisco Live!



GO BEYOND