LFN Developer & Testing Forum

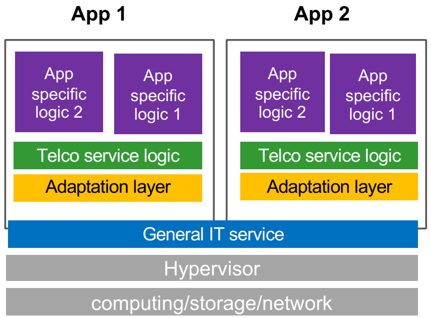
XGVela
Seed Code Design Walk-through

XGVela Team

XGVela Architecture Overview

1. Application tailoring:

- The NFs / applications are further decomposed according to the microservices architecture
- Strip away the parts that have nothing to do with the application itself



App 1

App specific logic 2

App specific logic 1

Telco service logic

Adaptation layer

App 2

App specific logic 2

App specific logic 1

Telco service logic

Adaptation layer

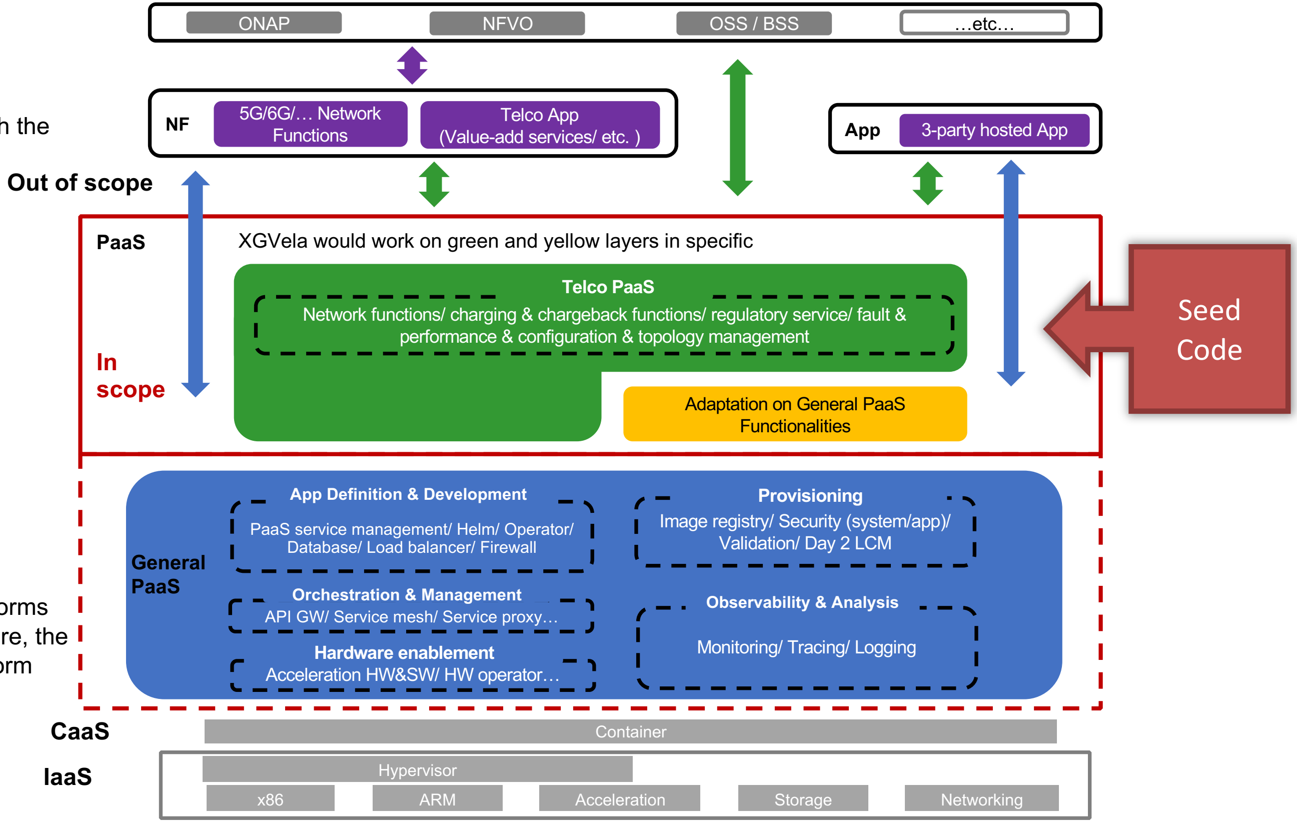
General IT service

Hypervisor

computing/storage/network

2. Platform addition:

- Support the coexistence of multiple resource forms
- Based on network element software architecture, the implementation of the service rely on the platform
- Provides unified capabilities through API



ONAP

NFVO

OSS / BSS

…etc…

NF

5G/6G/… Network Functions

Telco App (Value-add services/ etc.)

App

3-party hosted App

Out of scope

PaaS

XGVela would work on green and yellow layers in specific

In scope

Telco PaaS

Network functions/ charging & chargeback functions/ regulatory service/ fault & performance & configuration & topology management

Adaptation on General PaaS Functionalities

Seed Code

General PaaS

App Definition & Development

PaaS service management/ Helm/ Operator/ Database/ Load balancer/ Firewall

Provisioning

Image registry/ Security (system/app)/ Validation/ Day 2 LCM

Orchestration & Management

API GW/ Service mesh/ Service proxy…

Observability & Analysis

Monitoring/ Tracing/ Logging

Hardware enablement

Acceleration HW&SW/ HW operator…

CaaS

Container

IaaS

Hypervisor

x86

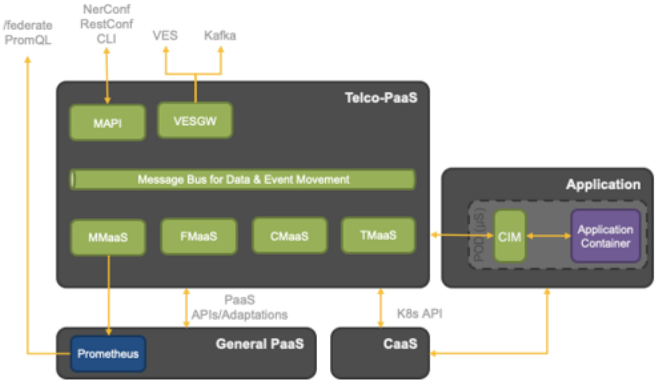
ARM

Acceleration

Storage

Networking

Seed Code Status



Status

✓ Seed code upload to GitHub

◔ Build Integration (GitHub Actions, Maven)

○ Code coverage

Key Stats

- 592 unique files
- ~57K LOC, ~8K Comments
- Primary languages - Go, Java

Following Telco-PaaS functions are seeded from Mavenir MTCIL.

- CMaaS: Configuration Management as a Service
  Provides a consistent and versioned view of configuration using Yang and NetConf. Supports Day-0, 1 and 2 configuration flows. Interfaces with k8s for configuration discovery and push.
- TMaaS: Topology Management as a Service
  Automatically discovery k8s services and builds 3GPP ManagedObjects for NFs, manages NF and µService states, supports LCM.
- FMaaS: Fault Management as a Service
  Application and platform events, TCA (via MMaaS/Prometheus), event subscriptions and ONAP VES 7.1 compliant NBI.
- VESGW: ONAP VES Gateway
  Built on ONAP/VESPA project with enhancement to support multi NF streams.
- CIM: CNF Interface Module
  A sidecar that provides a local integration and adaptation API layer for applications.
- Helm based packaging framework
- *Metrics Management as a Service (MMaaS) - In Progress*
  Uses Prometheus (in General PaaS) for metrics collection. Implements the control plane for configuring Prometheus for NF service discovery, KPIs and TCA.

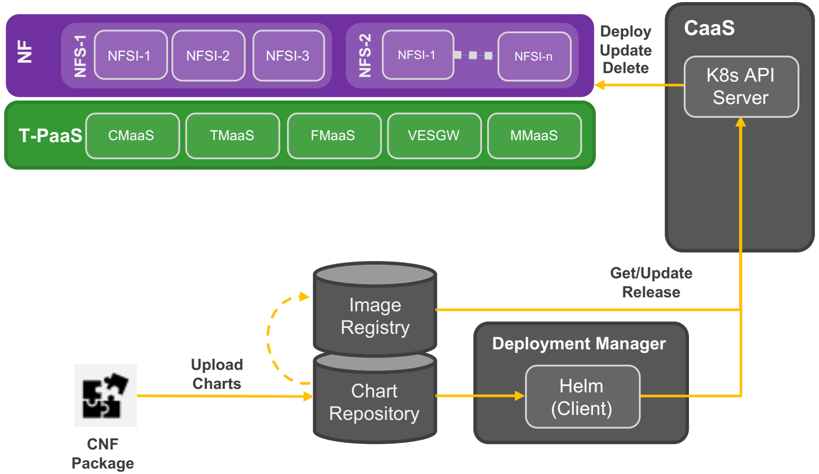
Code Structure

https://github.com/XGVela

Project	µService	Description	Cardinality	Notes
CMaaS	cmaas	Configuration Management	1	Single instance stateless service. Supports HA via Kubernetes probes and recovery.
FMaaS	fmaas	Fault Management	1..n	N-Active stateless service.
TMaaS	tmaas	Topology Management	1..n	N-Active stateless service.
	tmaas-gw	TMaaS Kubernetes Client	1..2	Active-Standby instances for HA.
NBI	vesgw	Notification Gateway	1,3,5	N-Active stateless service.
MMaaS	mmaas	Metrics Management	1	Single instance stateless service. Supports HA via Kubernetes probes and recovery.
Packaging	cnf-packaging	Helm based packaging framework	-	
	xgvela-builder	Build and package XGVela Telco-PaS	-	
Integration	cim	A sidecar that provides a local integration and adaptation API layer for applications.	-	

CNF Package

https://github.com/XGVela/cnf-packaging



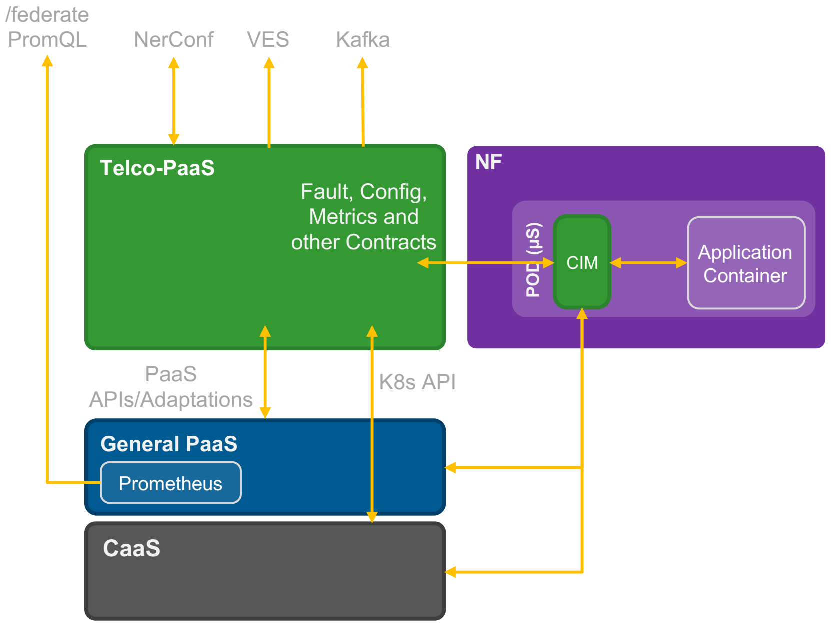
- Fully compliant Helm package
- Consists of collection of directories and files pertaining to CNF and CNFC/µService.
- Can be encapsulated in a CSAR pacakge

Sample Structure,

```
AMF
- Chart.yaml                # A YAML file containing information about the NF
- requirements.yaml         # OPTIONAL: A YAML file listing dependency for the NF
- values.yaml               # The default deployment configuration values for the NF
- LICENSE                   # OPTIONAL
- README.md                 # OPTIONAL
- alerts/alerts.yaml        # OPTIONAL: TCA rules
- config/
  - mgmt/{.yang, .json}     # OPTIONAL: NF/NFC configuration yang and json files.
  - static/{.json}          # OPTIONAL: NF/NFC static files
- eventdef/{.json}          # OPTIONAL: Event static and override parameters for enrichment.
- metrics/{metrics.yaml}    # OPTIONAL: Metrics recording rules
- dashboard/{.json}         # OPTIONAL: Grafana dashboard
- charts/                   # Exists only at root/NF level. Contains sub folders for each NFC.
  - amf-ee/
    - Chart.yaml
    - requirements.yaml
    - values.yaml
    - alerts/
    - config/
    - eventdef/
    - metrics/
 - amf-gw/
    :
 - amf-pathmgmt/
    :
 - amf-slicemgmt/
    :
 :
```

CIM

https://github.com/XGVela/cim



- CIM (CNF interface module) provides a local integration and adaptation API layer for applications.

- Deployed as a sidecar to application and platform containers.

- Implements various single node design patterns to enable loose coupling of application containers to the infrastructure.

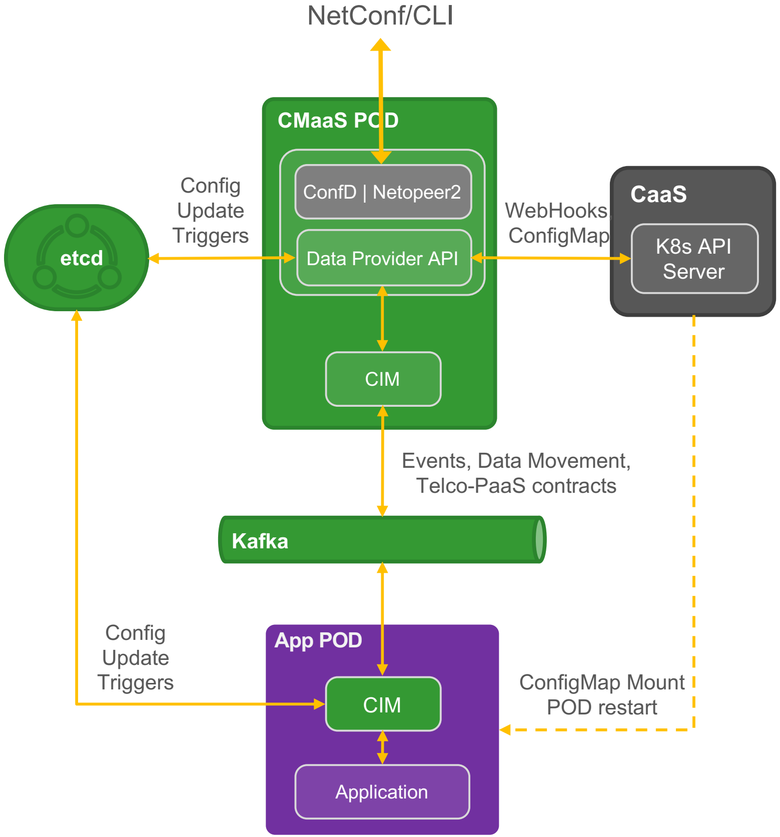
- Interfaces with application over REST for APIs and NATS for messaging and events.

CMaaS

https://github.com/XGVela/cmaas

LF NETWORKING

LFN Developer & Testing Forum



- Netopeer2 integration is ongoing. A future release might support option for the end user to pick between ConfD or Netopeer2.

- NF/NFC configuration are stored in ConfigMap. Addresses Day-0, 1 delivery.

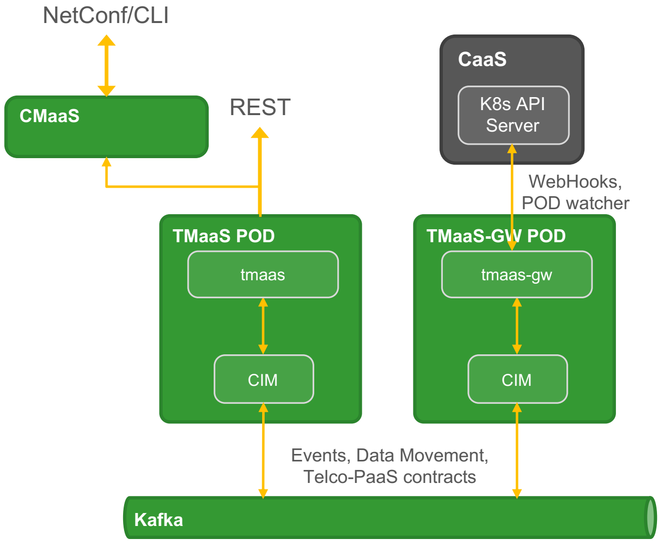
- Watches k8s for NF deployment/update.

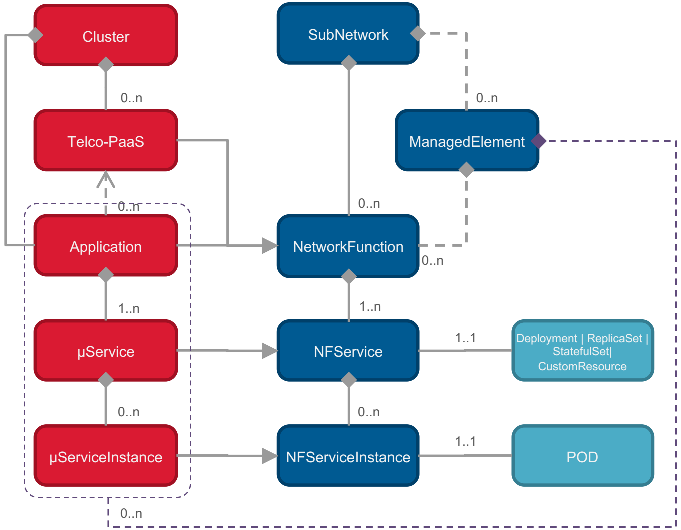
- On new NF deployment, loads any management configuration yang and json from ConfigMap and provisions the NetConf server module.

- Day-2 configuration changes are delivered via k8s rolling update or by direct API calls to application containers via etcd and CIM per application need.

TMaaS

https://github.com/XGVela/tmaas
https://github.com/XGVela/tmaas-gw





Logical Objects Managed Objects K8s Objects

POD annotation

```
xgvela.org/tmaas {
    telcoPaasId: <>
    dnPrefix: <>
    nfId: <>
    vendorName: <>
}
```

- Watches k8s for NF deployments, updates
- Constructs/updates MangedObjects (extended from 3GPP)
- Employs ETSI states for NF and NFC

ETSI NF States,

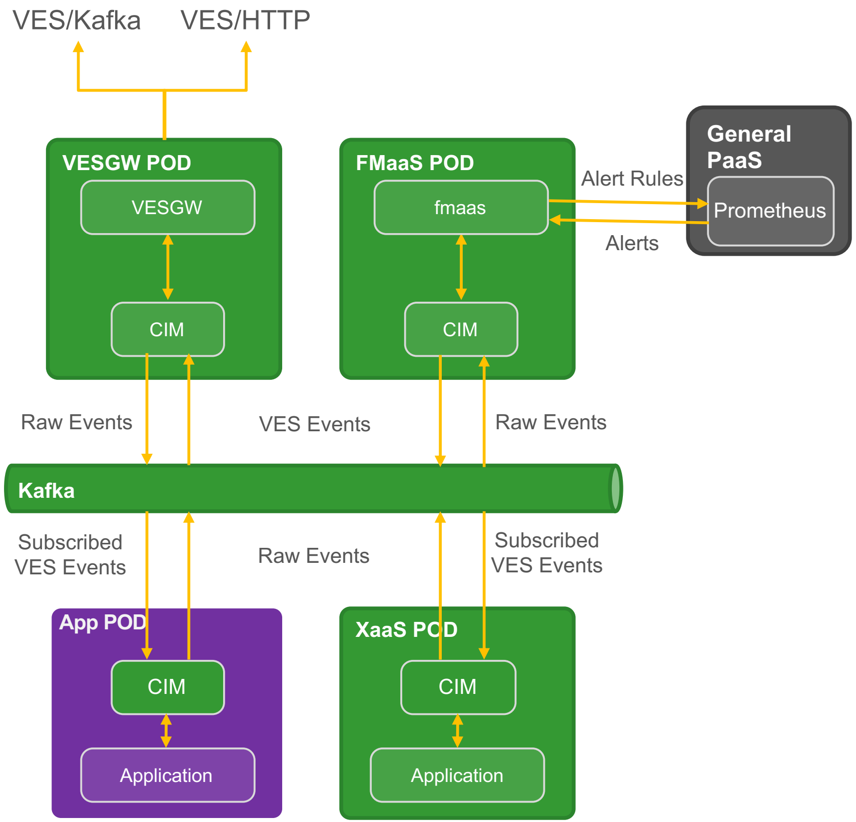
- NULL
- INSTANTIATED_NOT_CONFIGURED
- INSTANTIATED_CONFIGRED_ACTIVE
- INSTANTIATED_CONFIGURED_INACTIVE

Correlated from k8s probes and resource events,

- Startup Probe
- Liveness Probe
- Readiness Probe
- …

FMaaS

https://github.com/XGVela/fmaas



- Implements VES 7.1 specifications. Following domains are supported currently,
  - Fault
  - Heartbeat
  - Notification
  - TCA Alerts
  - Measurement

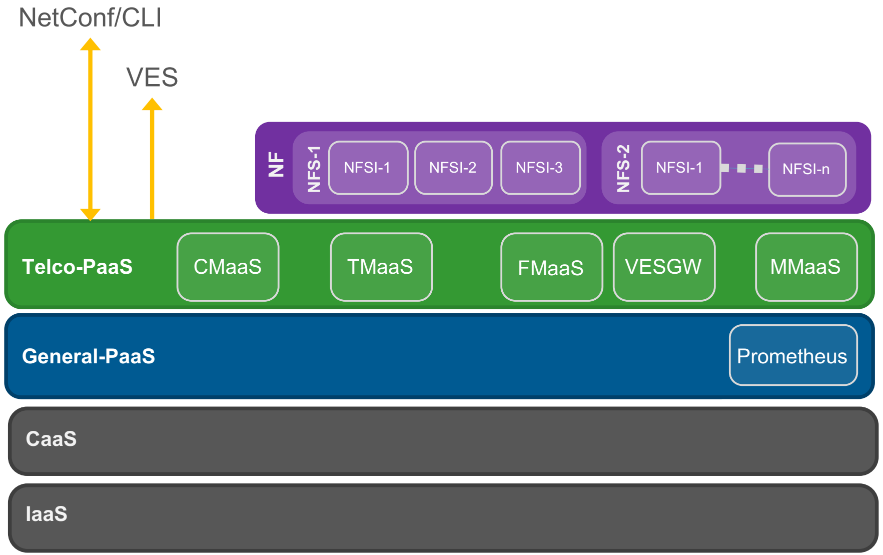
- Interfaces with applications via CIM and with Prometheus for TCA.

- Enriches and correlates events.

- VESGW implements notification service and supports push towards a VES collector or Kafka endpoints.
  - Primarily ONAP/VESPA extended to run in PaaS, outside of the NFs and supporting multiple NF streams.

- CIM provide APIs for applications to subscribe to and get notified about certain events based on nfId, category or eventName.

Basic Setup and Use cases



- When deployed, NFs are auto-initialised with dynamic Day-1 configuration

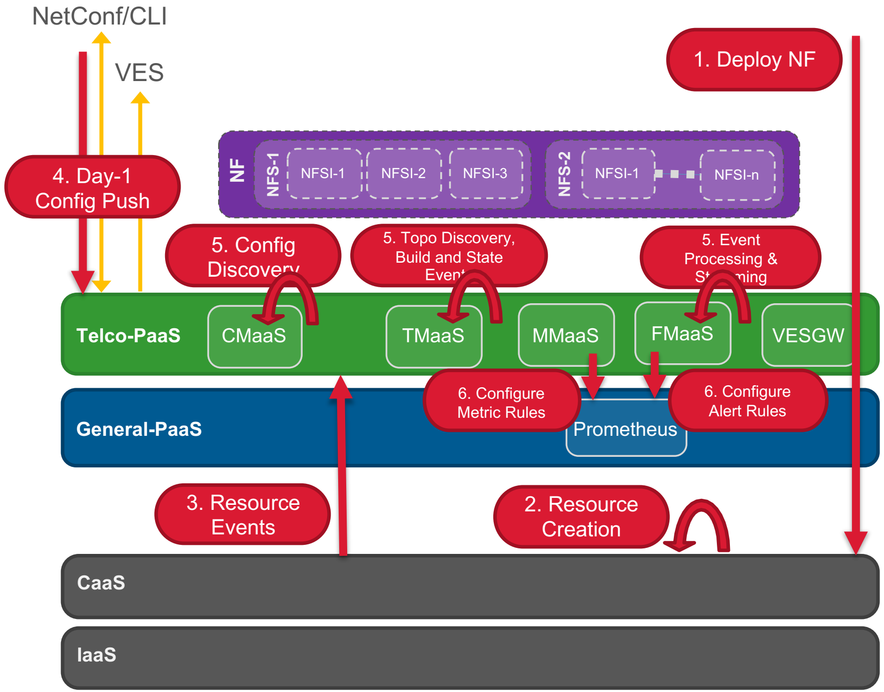
- When deployed, NF topology is automatically constructed, NFV states correlated, and state events are generated.

- When deployed, notifications and faults are generated and notified northbound over VES

- When deployed, NF is automatically configured to run PODs with active or standby roles as per the policy.

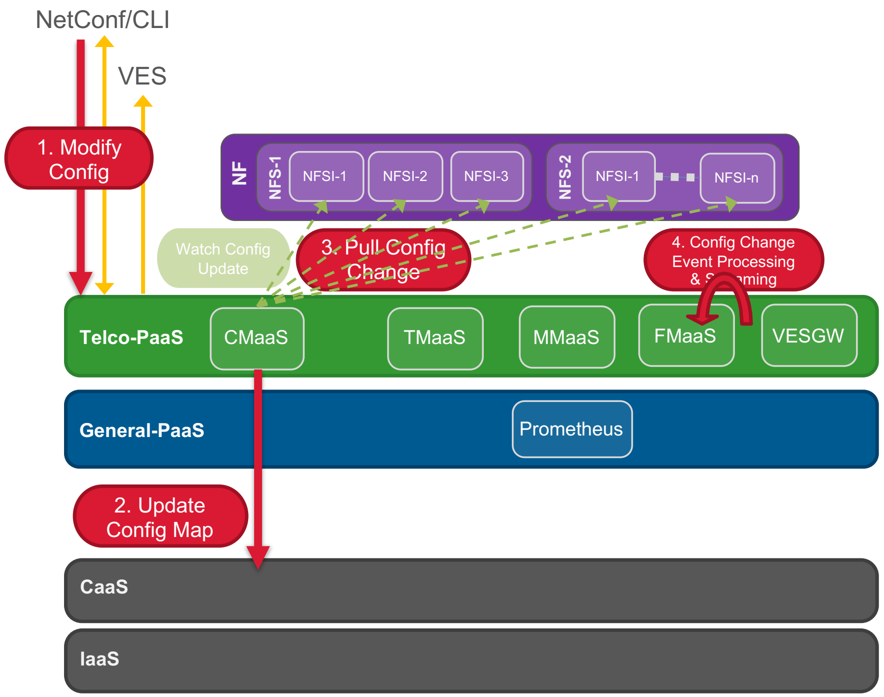
- When a configuration is changed via CLI (or over NetConf), changes are pushed to applicable µServices.

Deployment & Discovery



- When deployed, NFs are auto-initialised with dynamic Day-1 configuration
- When deployed, NF topology is automatically constructed, NFV states correlated, and state events are generated.
- When deployed, notifications and faults are generated and notified northbound over VES

Dynamic Config Update



- When a configuration is changed via CLI (or over NetConf/RestConf NBI), changes are pushed to applicable µServices.

LF
NETWORKING

LFN Developer & Testing Forum