MTCIL Demo

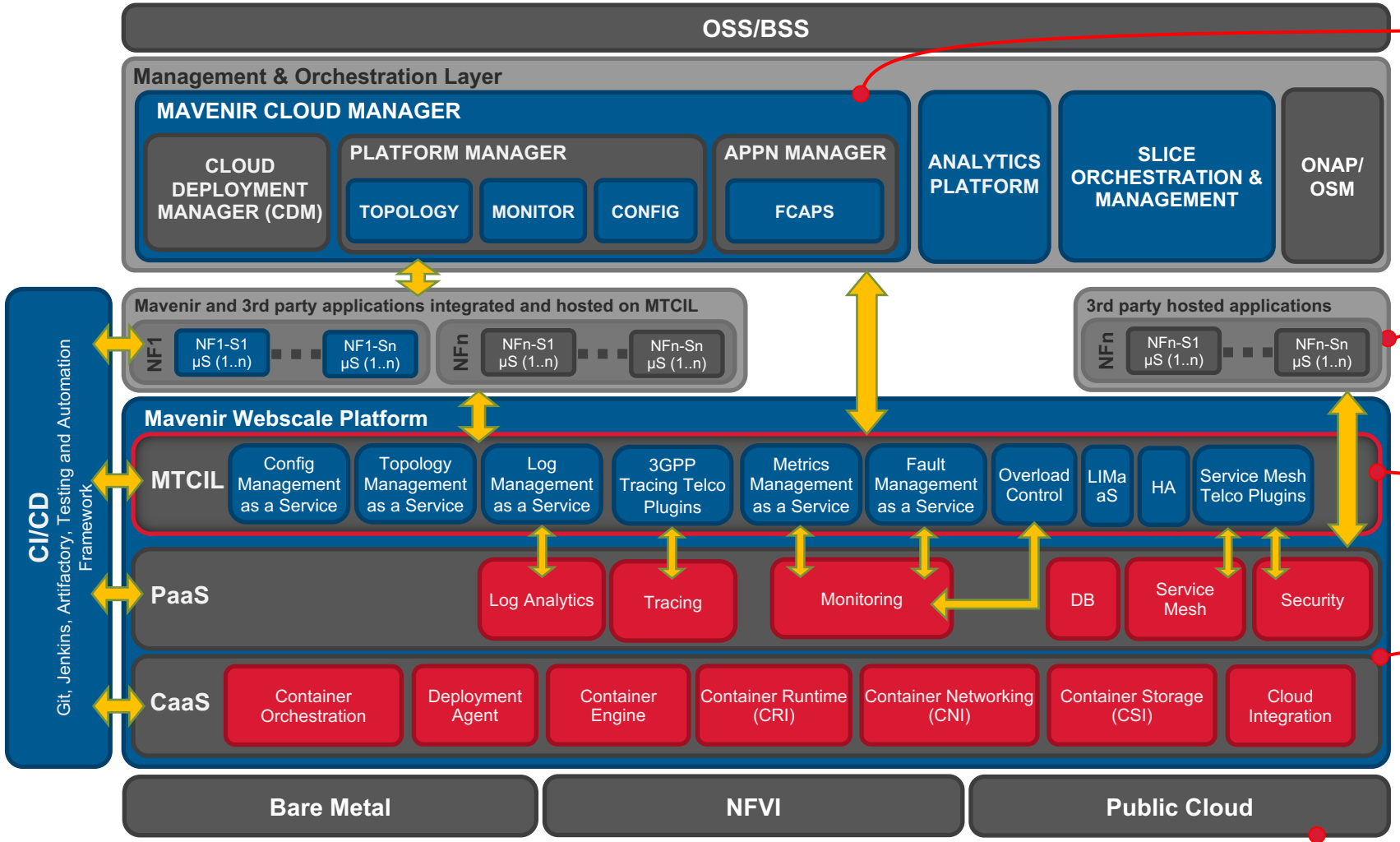
15th Oct 2020

Sandeep Karkala (sandeep.karkala@mavenir.com)
Sayan Sarkar (sayan.sarkar@mavenir.com)
Luis Zalce (luis.zalce@mavenir.com)

Overview

- Architecture Overview
  - Contribution to XGVela
- Design Overview
  - Management Model
  - Interfaces
  - Packaging
- Demo Use cases

Architecture Overview



Mavenir Cloud Manager (MCM)

- GUI based portal for deployment, monitoring and management of multiple clusters.
- Integrated with the **Mavenir Deployment Manager (MDM)** for ease of CaaS/PaaS and CNF deployment via HELM Charts and Ansible Playbooks

3rd party applications loosely integrated with MWP

Mavenir Telecom Cloud Integration Layer (MTCIL)

- MCF exposes 3GPP and Cloud-Native compliant North and South bound interfaces for OSS, MANO and PaaS integrations.
- Exposes APIs and CNF packaging procedures using open interfaces and tools.
- 3rd Party Network Functions can optionally consume these interfaces to enable Telco NF capabilities.

Generic **PaaS** & **CaaS** microservices assembled by Mavenir using CNCF certified opensource Cloud-Native software.

Support for container-based CNF deployment across Bare Metal, NFVI (containers in VMs) and Public Cloud.

Mavenir value add components:

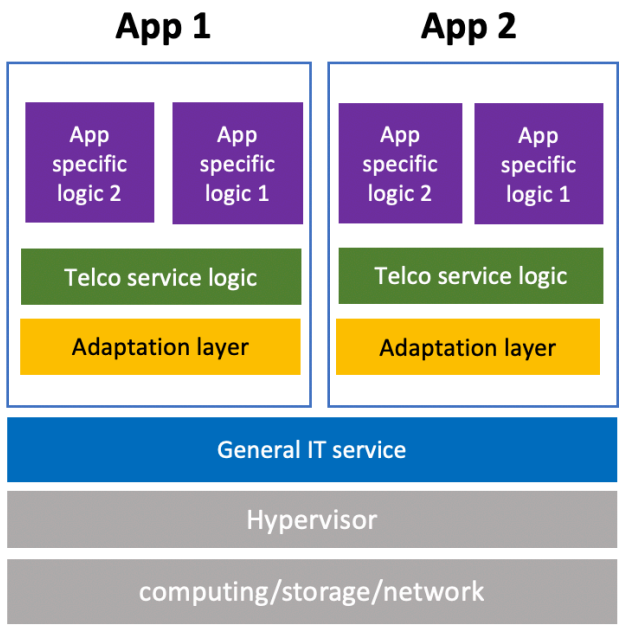
- Mavenir Webscale Platform (MWP)
- Mavenir Telco Cloud Integration Layer (MTCIL)
- Mavenir Cloud Manager (MCM)
- Mavenir Cloud Deployment Manager (CDM)

MAVENIR

Contribution to XGVela

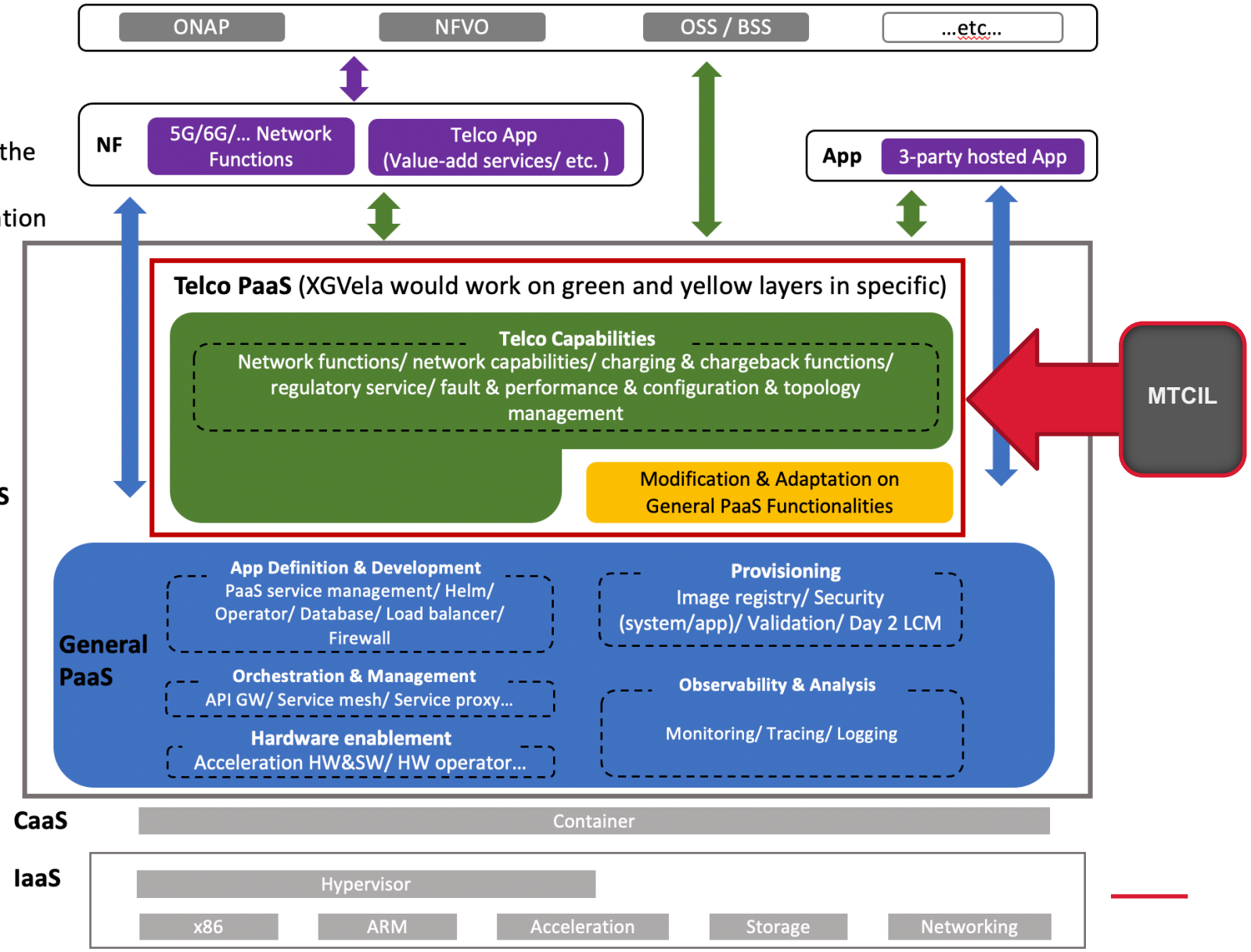
1. Application tailoring:

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

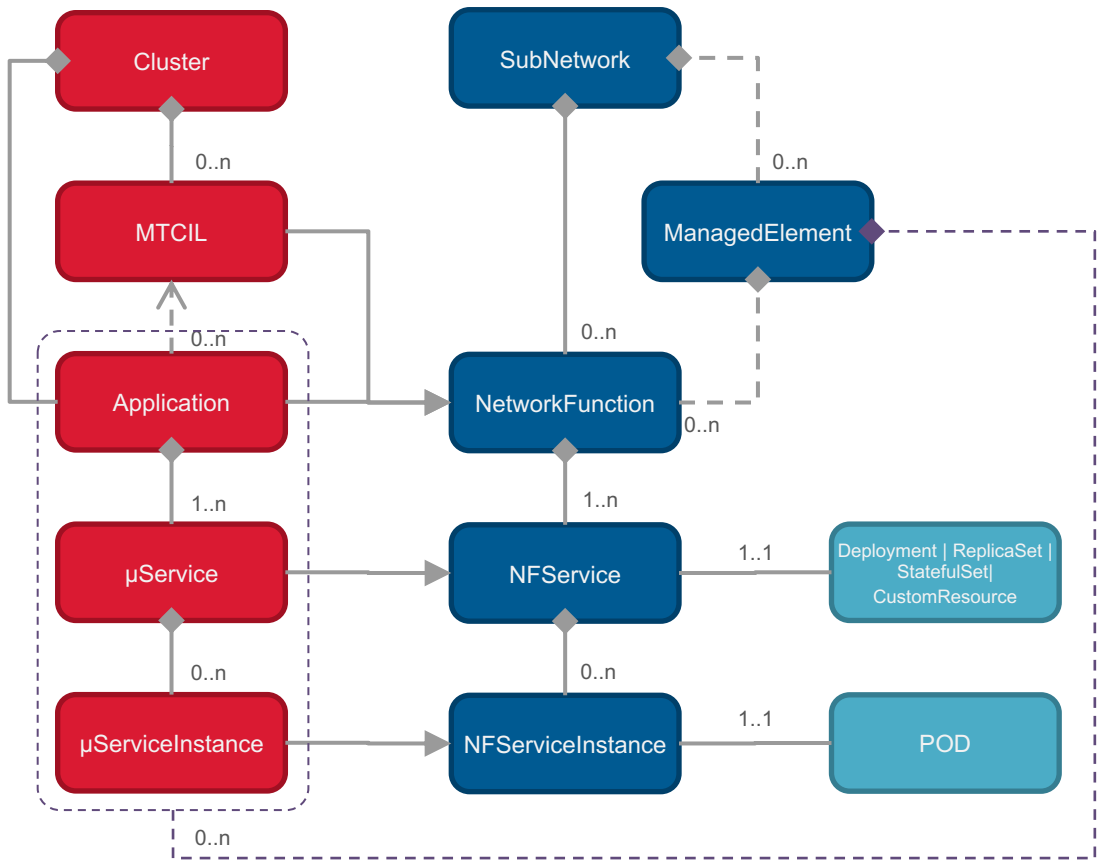


2. Platform addition:

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



Design Overview – Management Model



Modelling and managing PaaS services as VNFs is one of the options discussed under NFV (NFV-IFA029)

7 Potential Architecture Enhancements

7.1 Architectural enhancement on PaaS related use cases

7.1.1 Overall NFV Architecture

7.1.1.1 General

Potential architectural enhancements depend on the design options selected to model PaaS services and thus PaaS services management. Three main design options can be envisioned:

- PaaS services are modelled as VNFs
- PaaS services are modelled as a new type of NFVI resources
- PaaS services are modelled as a new type of object specific to the PaaS layer

ETSI NF States,

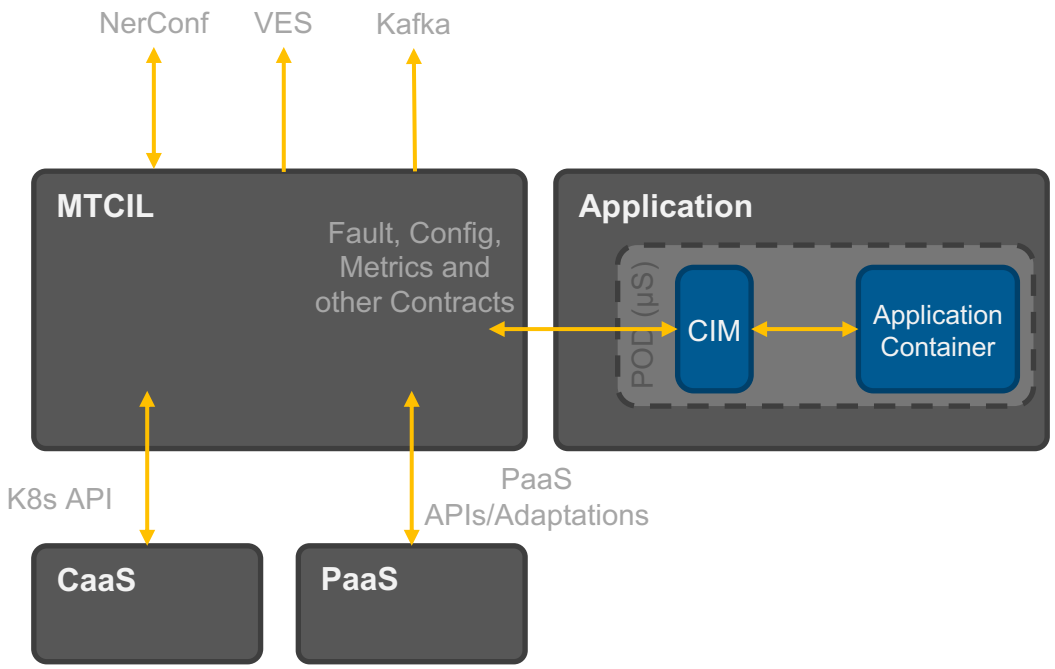
- NULL
- INSTANTIATED_NOT_CONFIGURED
- INSTANTIATED_CONFIGRED_ACTIVE
- INSTANTIATED_CONFIGURED_INACTIVE

Correlated from k8s probes and resource events,

- Startup Probe
- Liveness Probe
- Readiness Probe
- …

MAVENIR

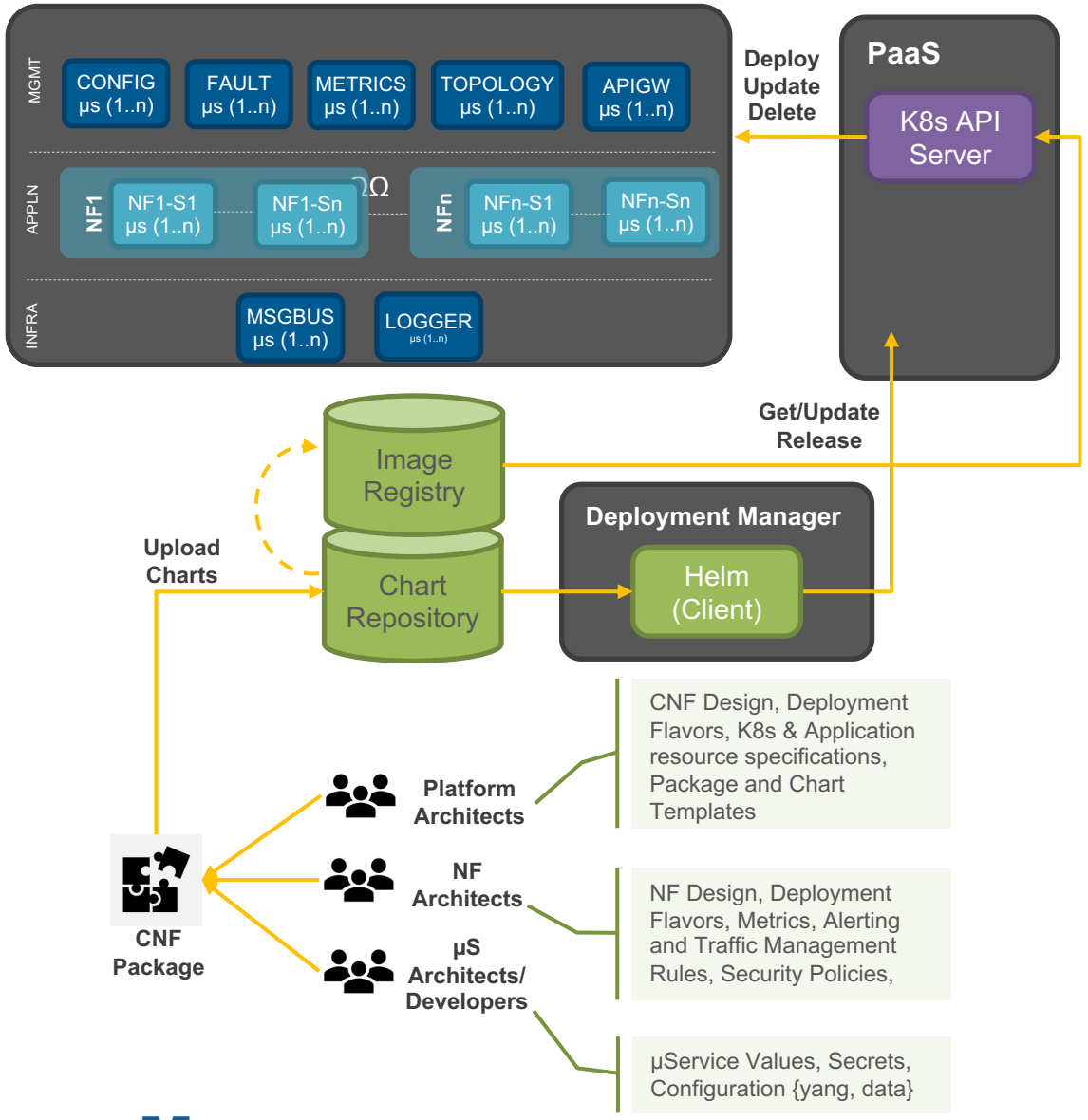
Design Overview – Interfaces



- CIM (CNF interface module, a component of MTCIL) provides single integration point API for Mavenir or 3rd party hosted applications
  - Deployed as a sidecar to application 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.
- MTCIL
  - CMaaS exposes NetConf NBI for configuration and topology management. Interfaces with k8s for config discovery and push. JSON/REST interface to push config to application containers.
  - TMaaS interfaces with k8s for auto-discovery of services, builds 3GPP NF models. Exposes REST and also interfaces with CMaaS to expose Topology data over NetConf.
  - MMaaS is primarily Prometheus at the core.
  - FMaaS interfaces with application via CIM and with Prometheus for TCA. Enriches and exports events VES (version 7.1). Events can also be pushed via Kafka.
  - HAaaS provides a distributed HA model for fast switchover for stateful services. Interfaces with application via CIM.

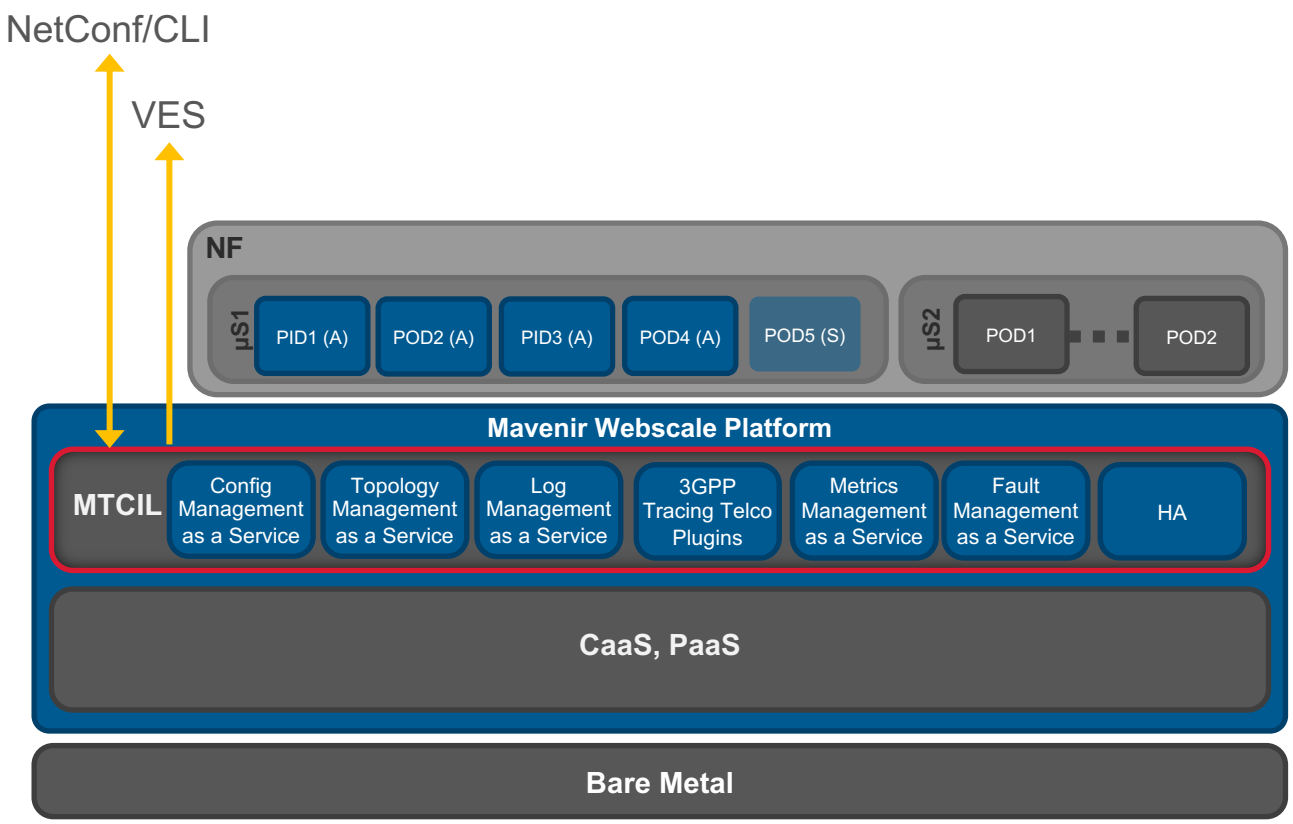
MAVENIR

Design Overview - Packaging



- We take an opinionated (yet flexible) way of packaging CNF's as an essential measure to ensure,
  - Simplicity (by Separation of Concerns): Each team focuses on data and configuration most relevant to them.
    - µService architect/developer only needs to care about µService deployment values (only a subset), configuration schema, data and Secretes.
    - Network Function architect/Developer only needs to care about NF design, deployment flavors, various application specific rules and policies.
    - Platform Architect/Developer provides packaging framework and tools for generating necessary Helm charts, templates and Kubernetes resource specifications, annotation and label injections necessary.
  - Consistency
    - CNF deployment specification and packaging is abstracted and managed through a consistent set of opinionated procedures and design.
    - Packaging framework is not NF specific
    - All services are consistently configured
- CNF Packaging consists of two parts,
  - Package Builder
  - Deployment Manager (It also takes care of C/PaaS deployment along with CNF and is discussed in detail in the next section).
  - Can be encapsulated in a CSAR

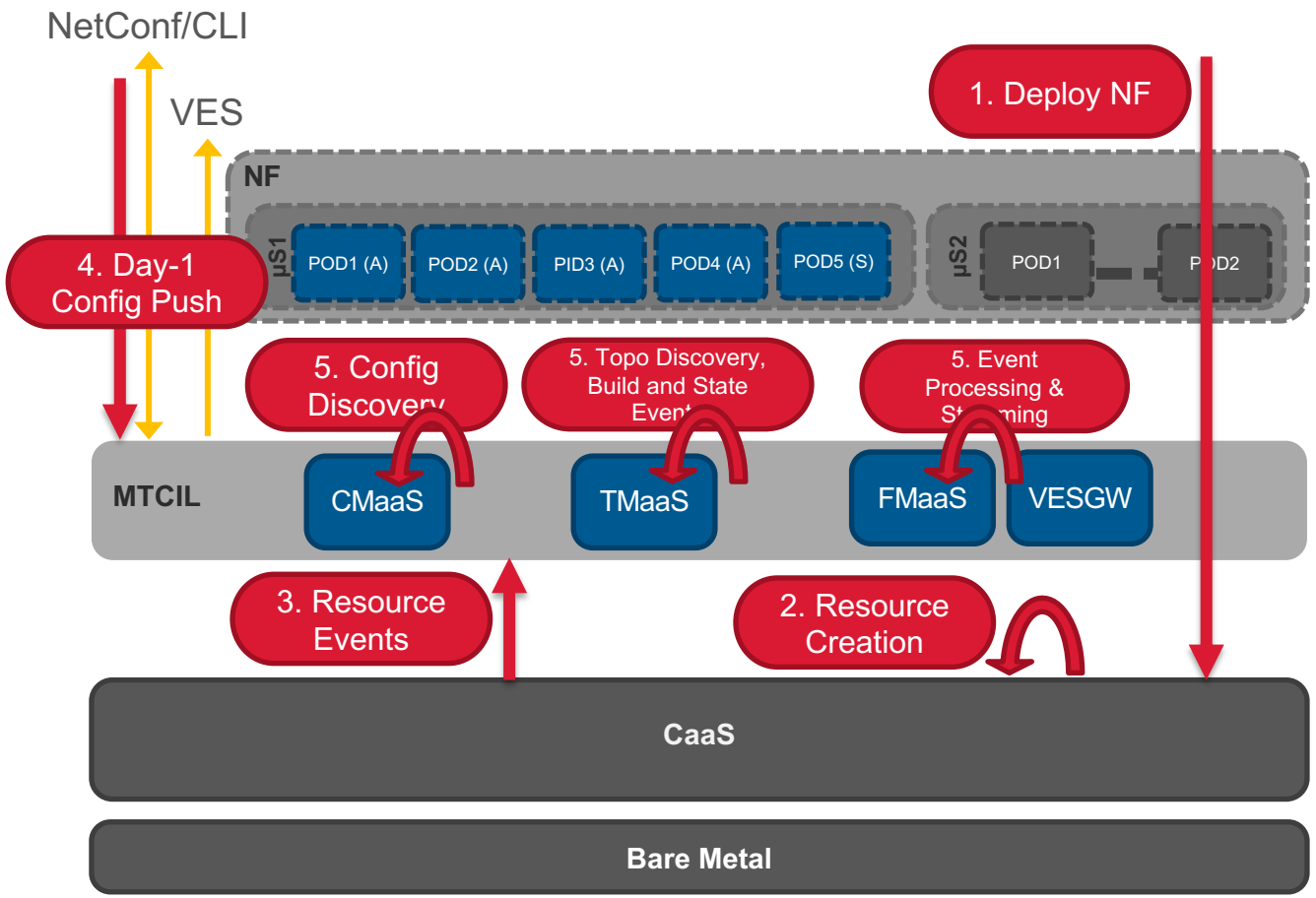
Demo – Setup & Use cases



- 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 with HAaaS preference, 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.
- When an active NFSI/POD fails, active role is switched over to a standby NFSI/POD.

MAVENIR

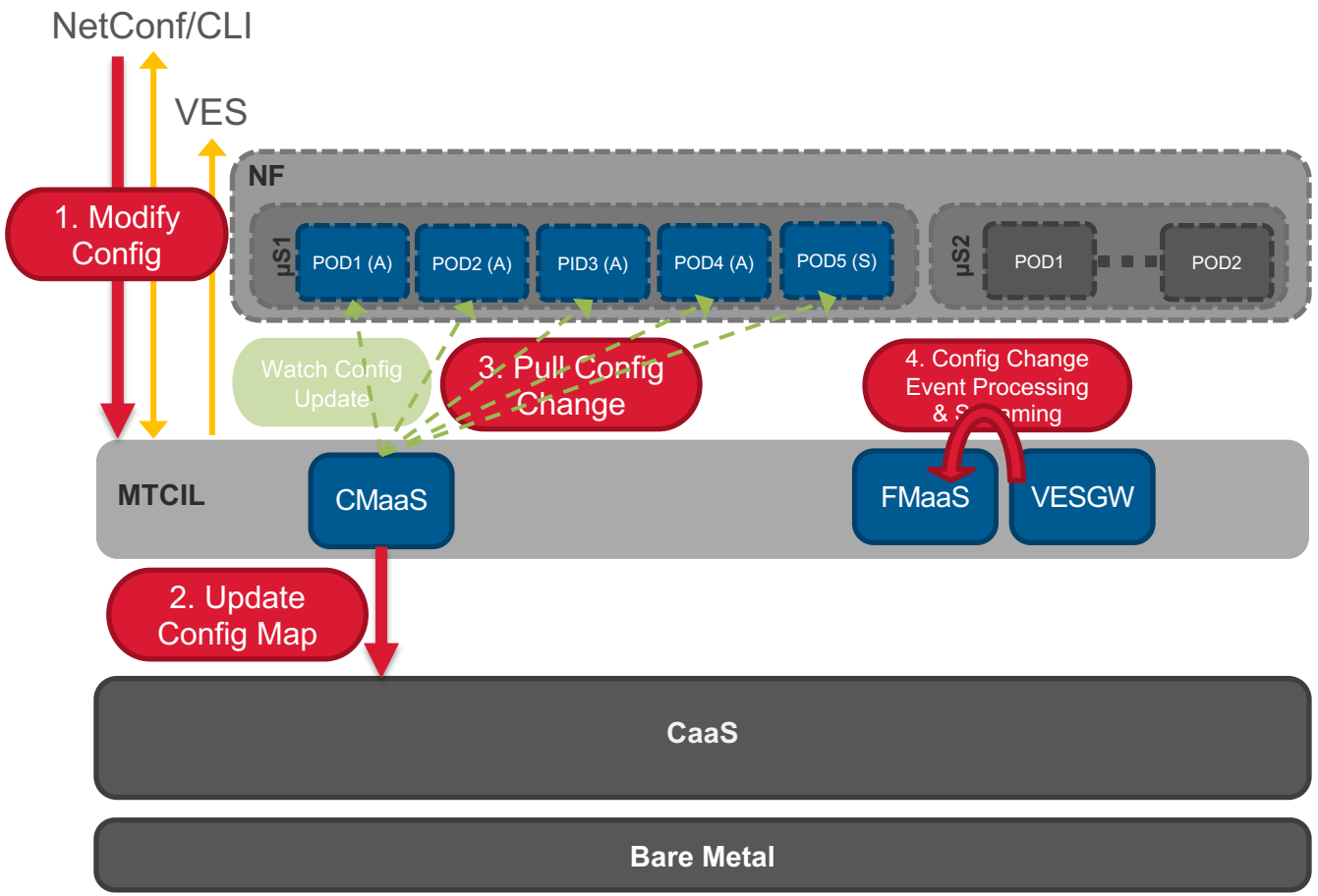
Demo – Use case: Deployment and Discovery



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

MAVENIR

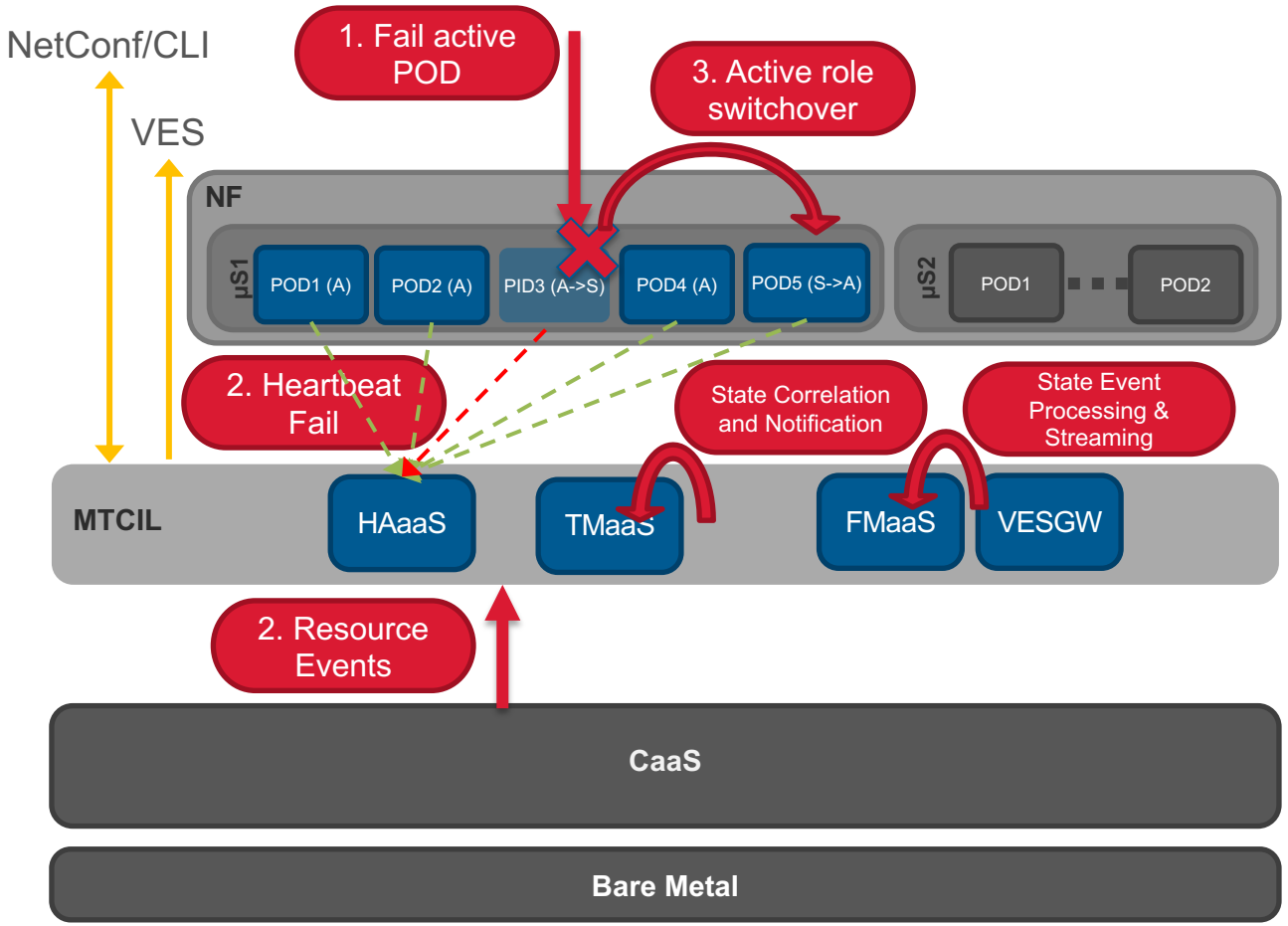
Demo – Use case: Dynamic Config Update



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

MAVENIR

Demo – Use case: HA



- When an active NFSI/POD fails, active role is switched over to a standby NFSI/POD.

THANK YOU