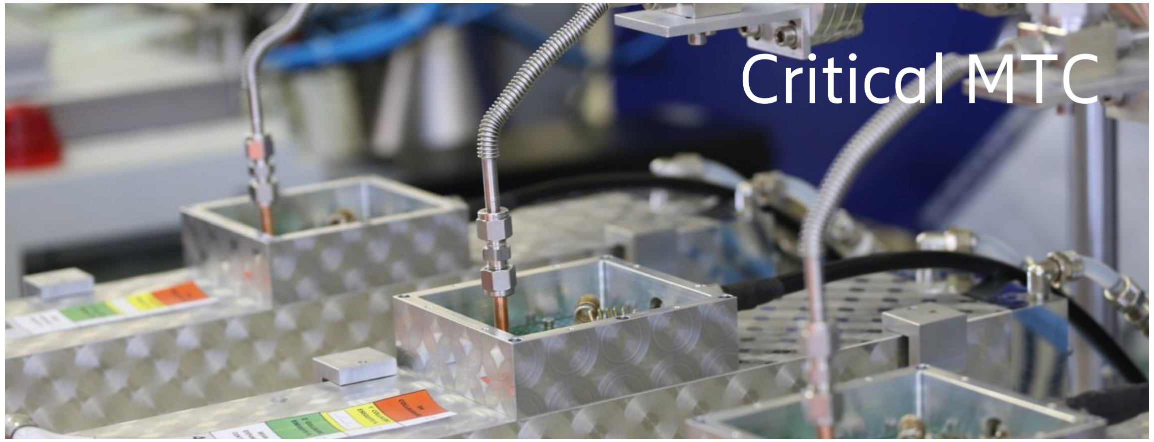
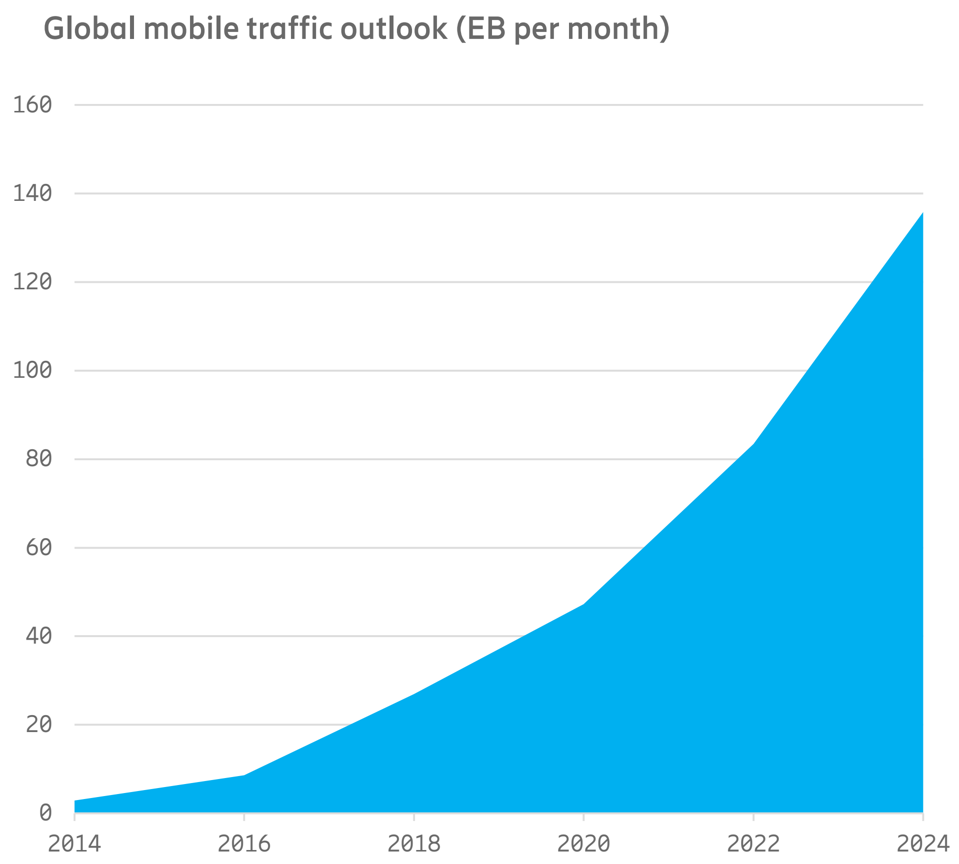
Cloud Native Introduction

Balaji Ethirajulu

5G is use case driven



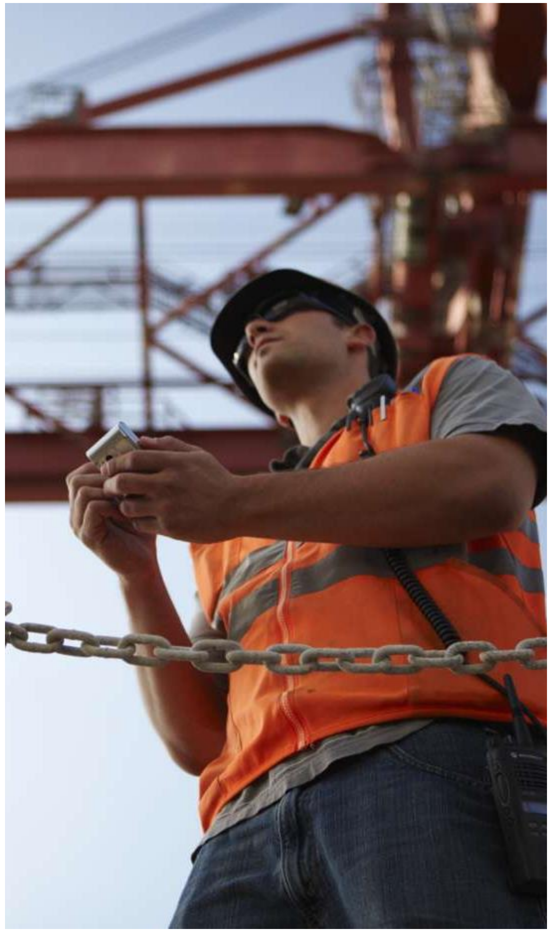
Mobile data traffic outlook



In 2024, 25 percent of mobile data traffic will be carried by 5G networks.

x5

Total mobile data traffic is expected to increase by five times



5G – Cornerstones and role of open source

- Cloud Native / Microservices
- Automation
- Orchestration
- AI and AI driven Policy

Ericsson Demo’s – Booth 401

- Enabling Edge Cloud.
- 5G Automation & Network Slicing
- Cloud native 5G Core & Exposure
- Cognitive OSS & AI/ML Platform

Ericsson – Tutorial Highlights

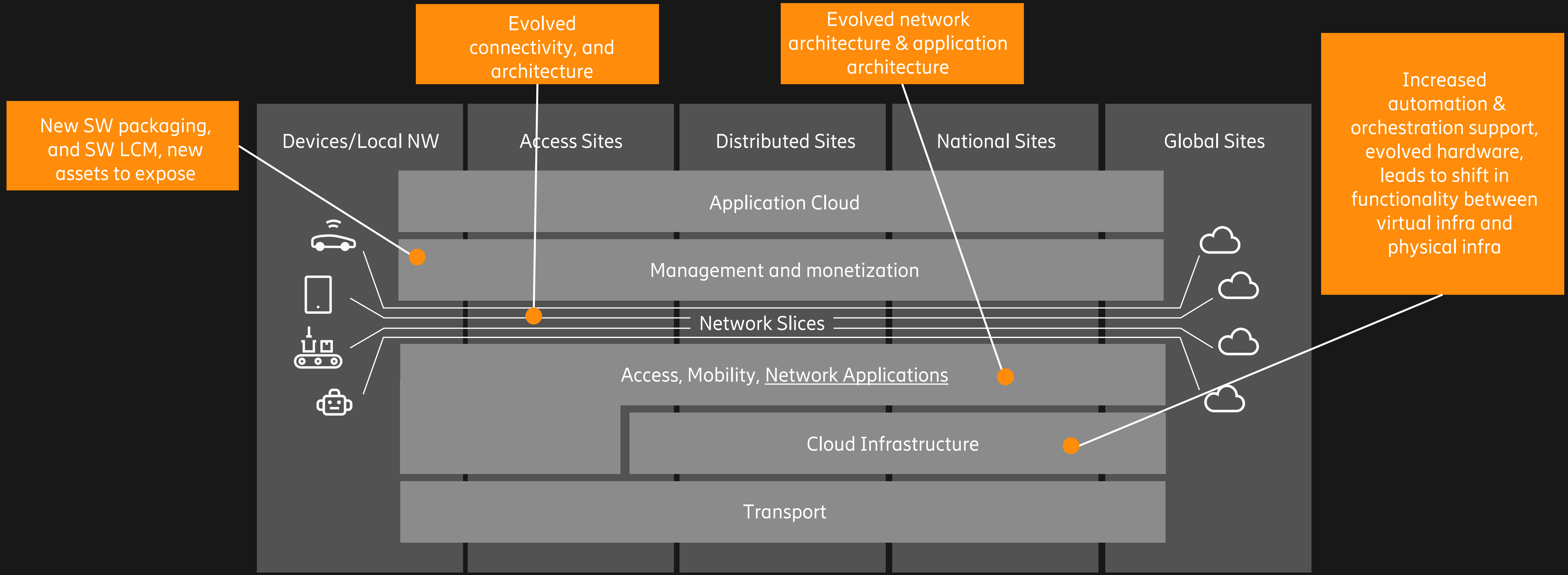
- Cloud native Introduction – Balaji Ethirajulu
- Cloud native design principles / Architecture – Tamas Zsiros
- A Cloud Native Application Design, Opportunities & Challenges – Henrik Saavedra Persson
- Cloud native “Dual-mode 5G cloud core” and Network Exposure (Programmability) – Jitendra Manocha
- Cloud Container distribution /Distributed cloud - Sampathkumar Santhanakrishnan
- Demo – Show the configuration and installation of CCD/k8’S and also show a scaling on a Kubernetes cluster - Sampathkumar Santhanakrishnan and Russ Mahon

ONS2019
Cloud Native Tutorial
Principles & Architecture

Tamas Zsiros

Cloud native values

5G relies on an evolving ecosystem



Cloud native values

Speed

- Fast, low-cost introduction of new services in small scale

Scale

- Scale fast at low cost from hundreds of users to millions

Efficient operations

- Automation, no-touch operations
- Legacy and new services
- Life-cycle independence (services/infrastructure)

Performance and capacity

- Optimized capacity throughput and resource utilization

Software development in fast-moving, independent and empowered teams

Design principles

Cloud native design principles

Agnosticity

A cloud native application has to be agnostic to underlying infrastructure »

- IaaS
- Bare metal
- Kubernetes based CaaS

Decomposed SW

Decomposition enables unified and flexible LCM and better resource utilization »

- Microservice architecture
- Independant lifecycle management per microservice

Application resiliency

Change of metrics with regards to resilliency»:

- Pets vs Cattle
- HA achieved through distribution and fault isolation

State optimized design

Handling state tailored for application context and state/data type »

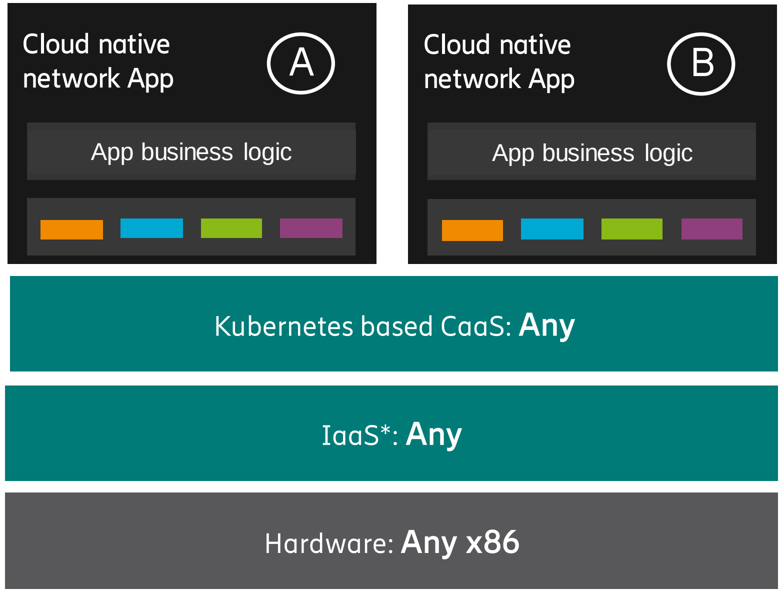
- Separation of application logic and state
- Proper state storage to guarantee performance

Orchestration and automation

Reduce or remove the need for human intervention to decrease OPEX and increase speed»:

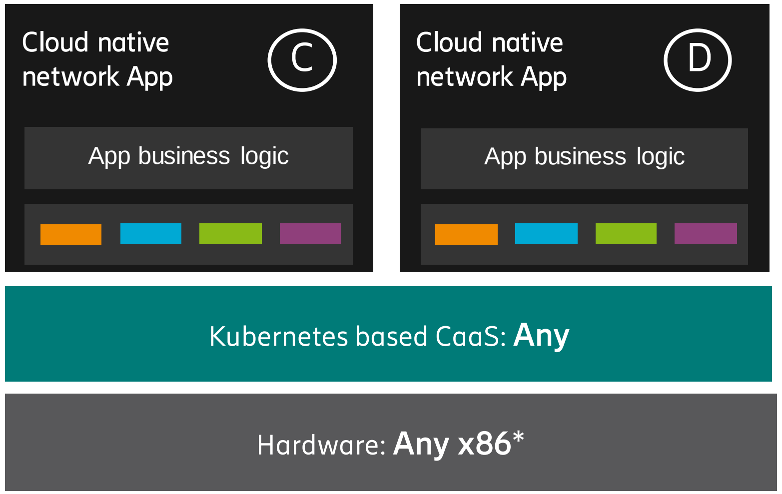
- Internal scaling and load distribution hadleld by CaaS
- Network level automation through model driven management

Cloud native application stacks



CaaS on IaaS
- CaaS LCM: similar to any IAAS workload

- Virtualization overhead (KVM, VMWare)
- Security: harder to get from guest OS to host kernel

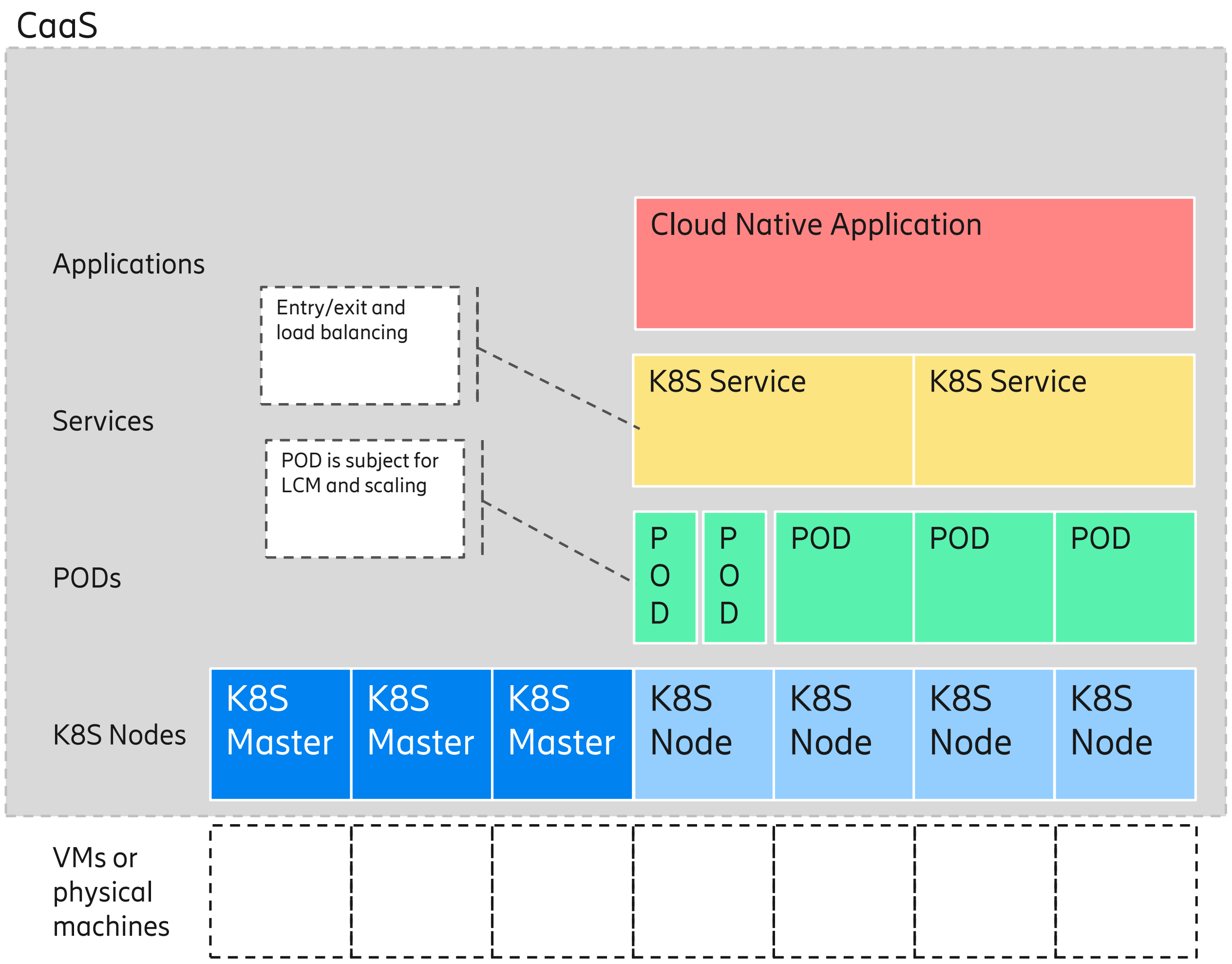


CaaS on bare metal
- CaaS LCM: specialized tools (e.g. MaaS)

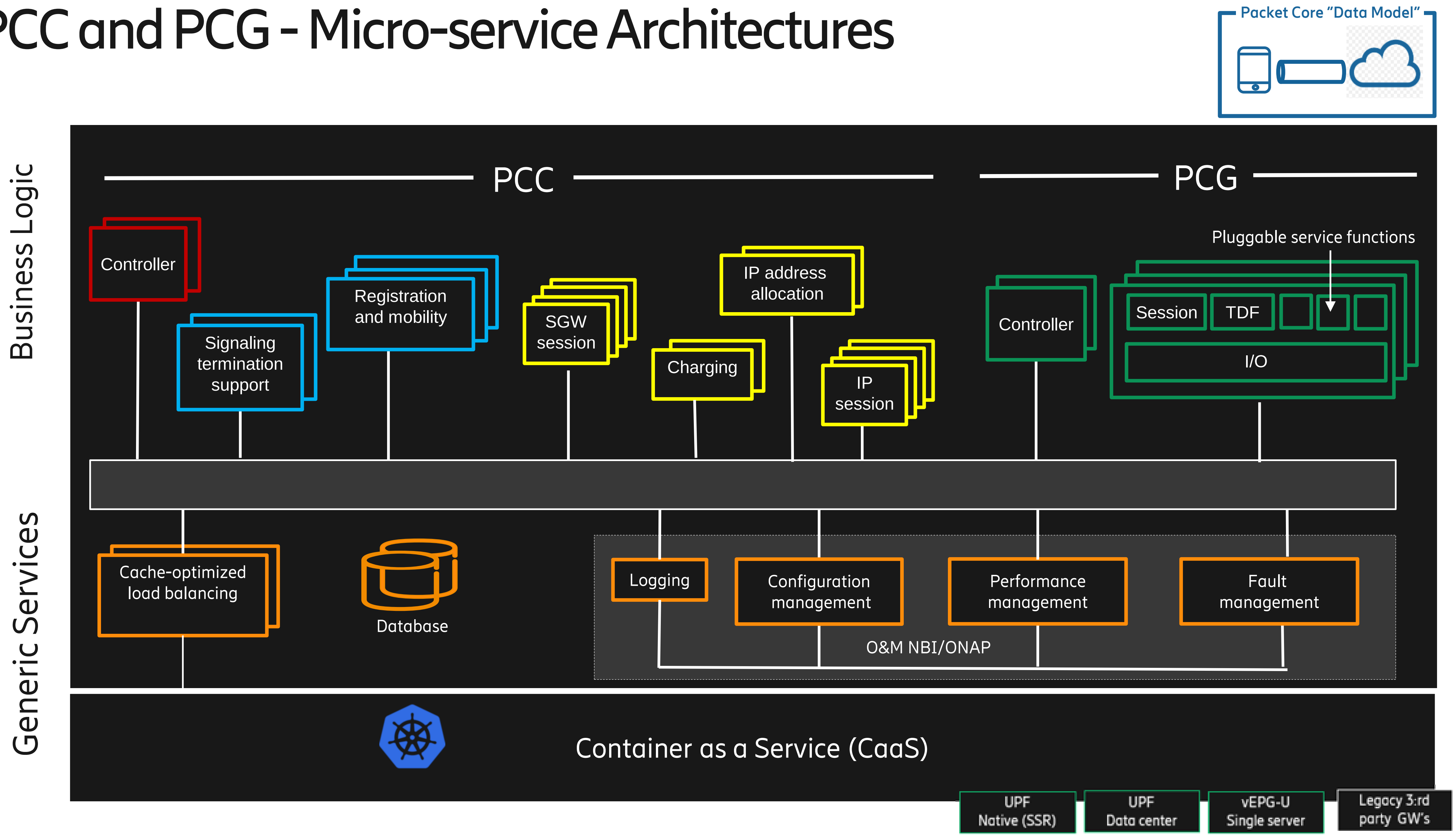
- No virtualization overhead
- Security: easier to get from host user space to host kernel space
  - Advanced container runtimes with improved separation, i.e. katacontainers

Cloud Native Application
using Kubernetes

- Clusters of Nodes, Pods, Services
- The dynamicity applies on many layers
  - Containers, PODs “come and go”
  - Higher layers depend on the state of clusters on lower layers (**downward dependency**)
  - Ideally, every layer should handle its downward dependency on its own without affecting layers above (e.g. the application layer should not care whether the HW is operational or not etc.)
- **Kubernetes** keeps track of (dynamic) cluster members and does general housekeeping around them (starting, stopping, upgrading, health checking, etc...)



PCC and PCG - Micro-service Architectures



Deployment configurations of cloud native applications

VMs vs Containers in Telco space?

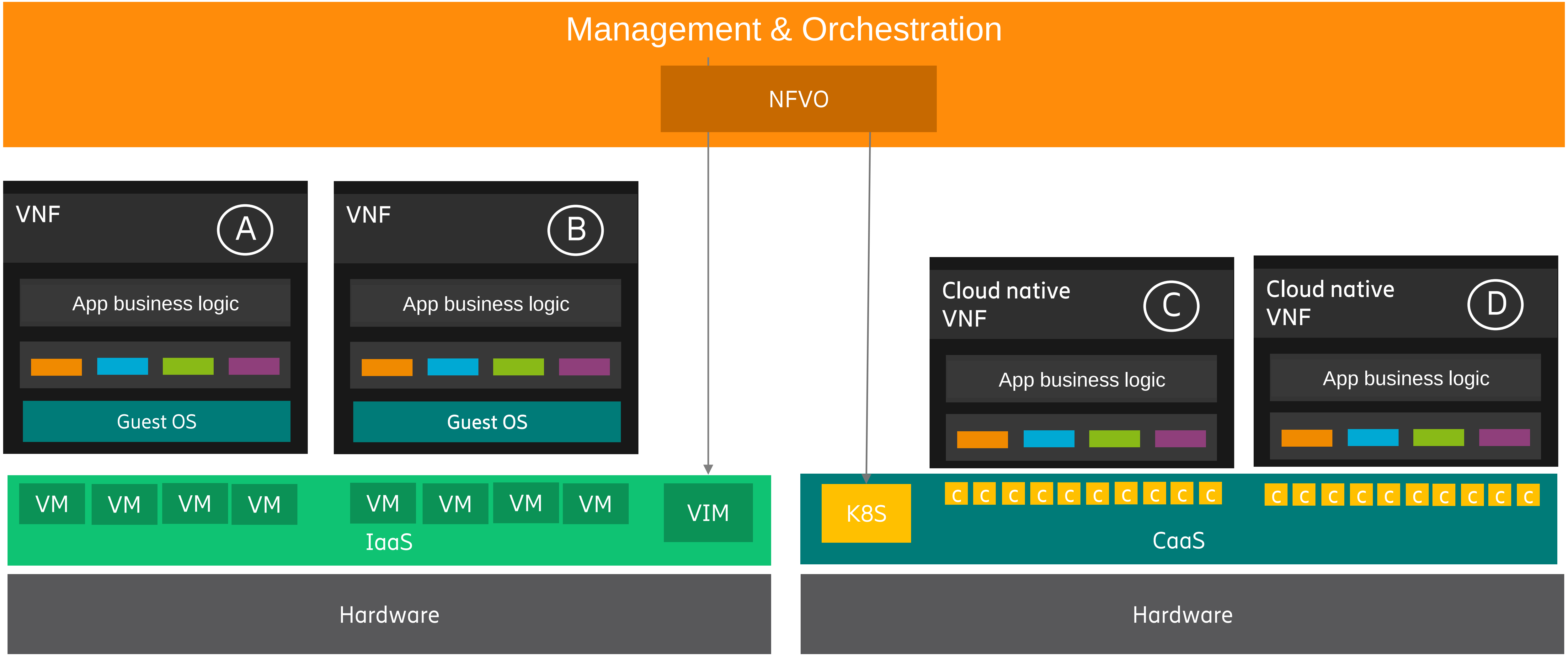
Or rather: IAAS vs CAAS

	VMs & IaaS	Containers & CaaS
Workload isolation	++	o
Compute performance	+	+ (+)
I/O performance	o / ++ (PCI-PT / SR-IOV)	+ / ++ (Bare Metal / SR-IOV)
LCM operation efficiency	o	++
Platform maturity / ecosystem	++	+ (++)
VNF package & deployment size	–	+
Support for cloud native design	o	++ (using Kubernetes)

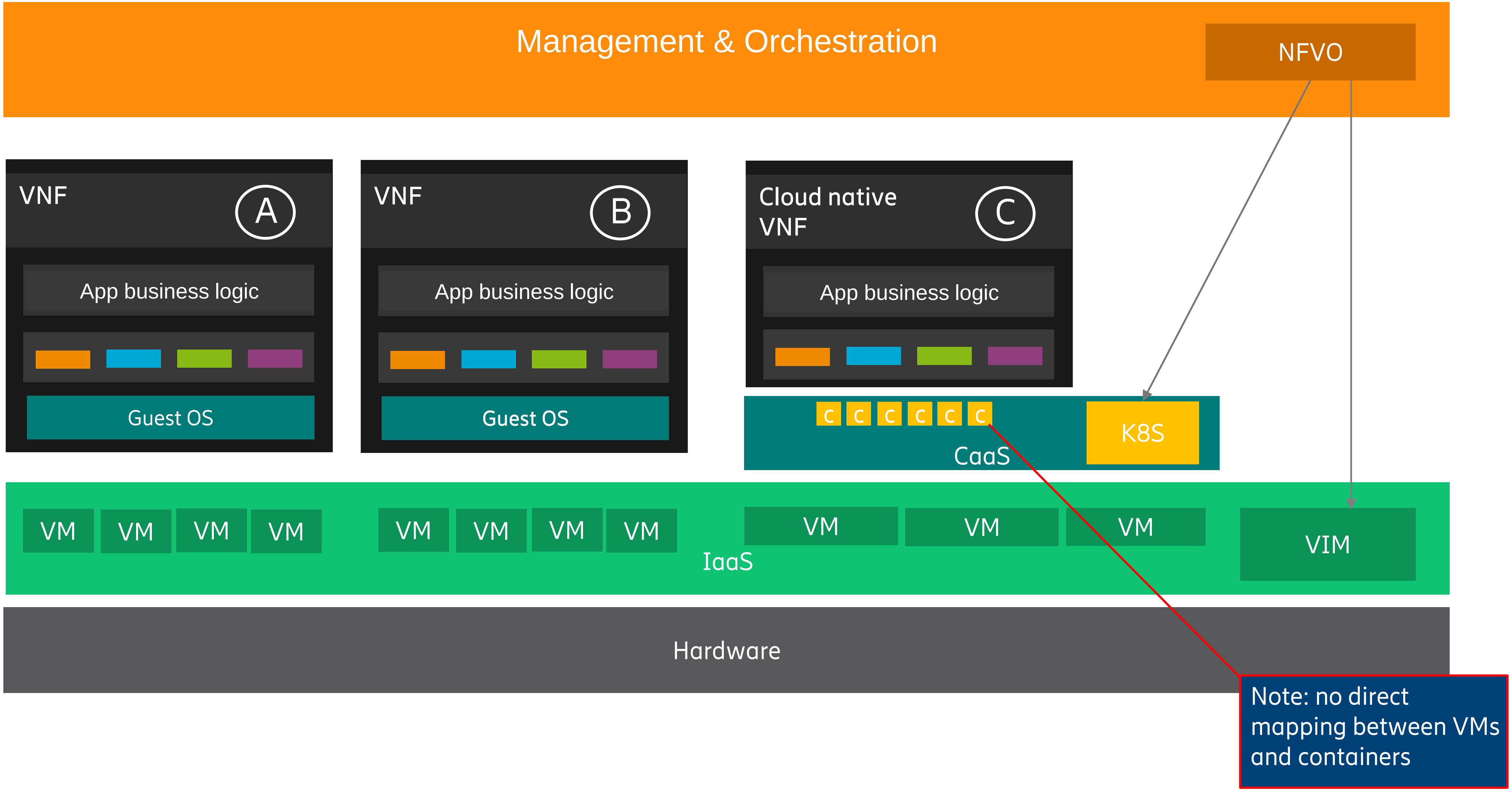
Technologies emerging: Kata Containers, FireCracker, gVisor... (+ virtlet, KubeVirt...)

The Kubernetes ecosystem is evolving rapidly and ETSI and ONAP are adopting it

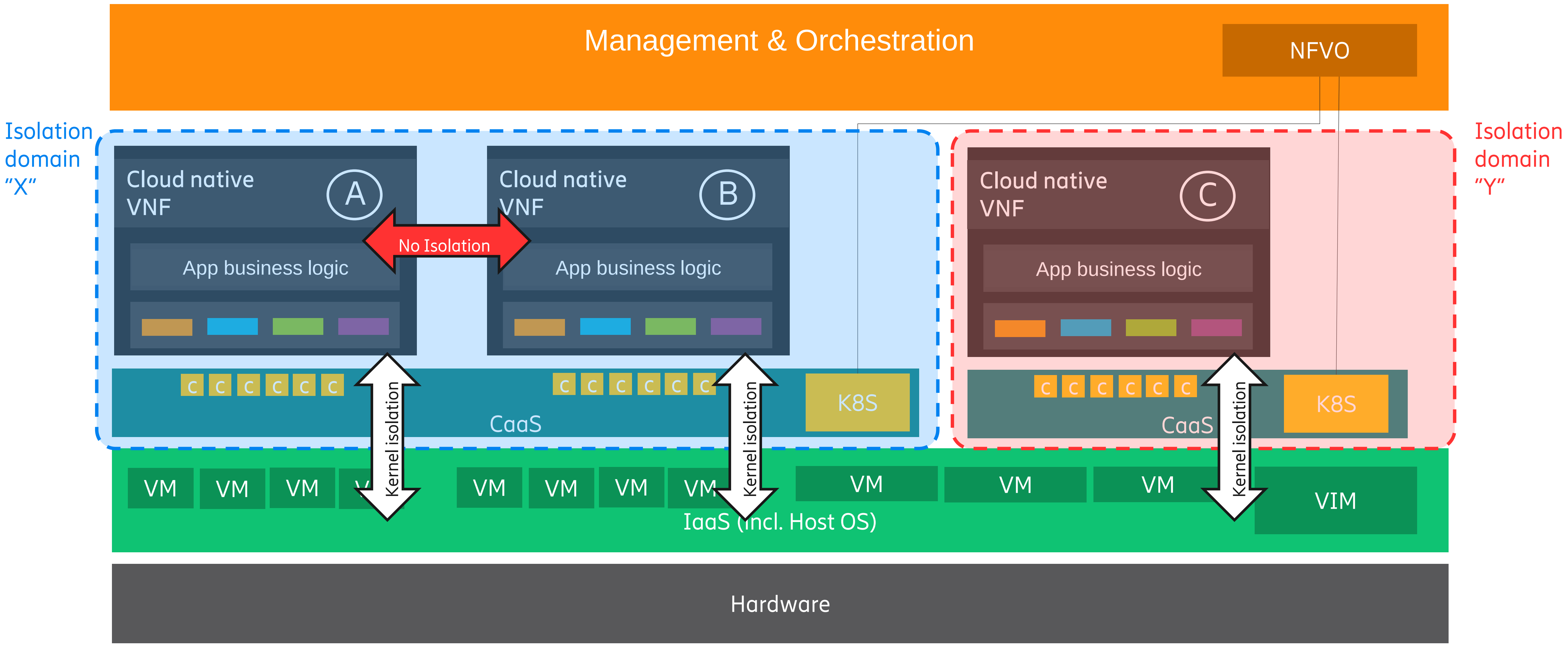
Multi VIM - Carrier supported CaaS



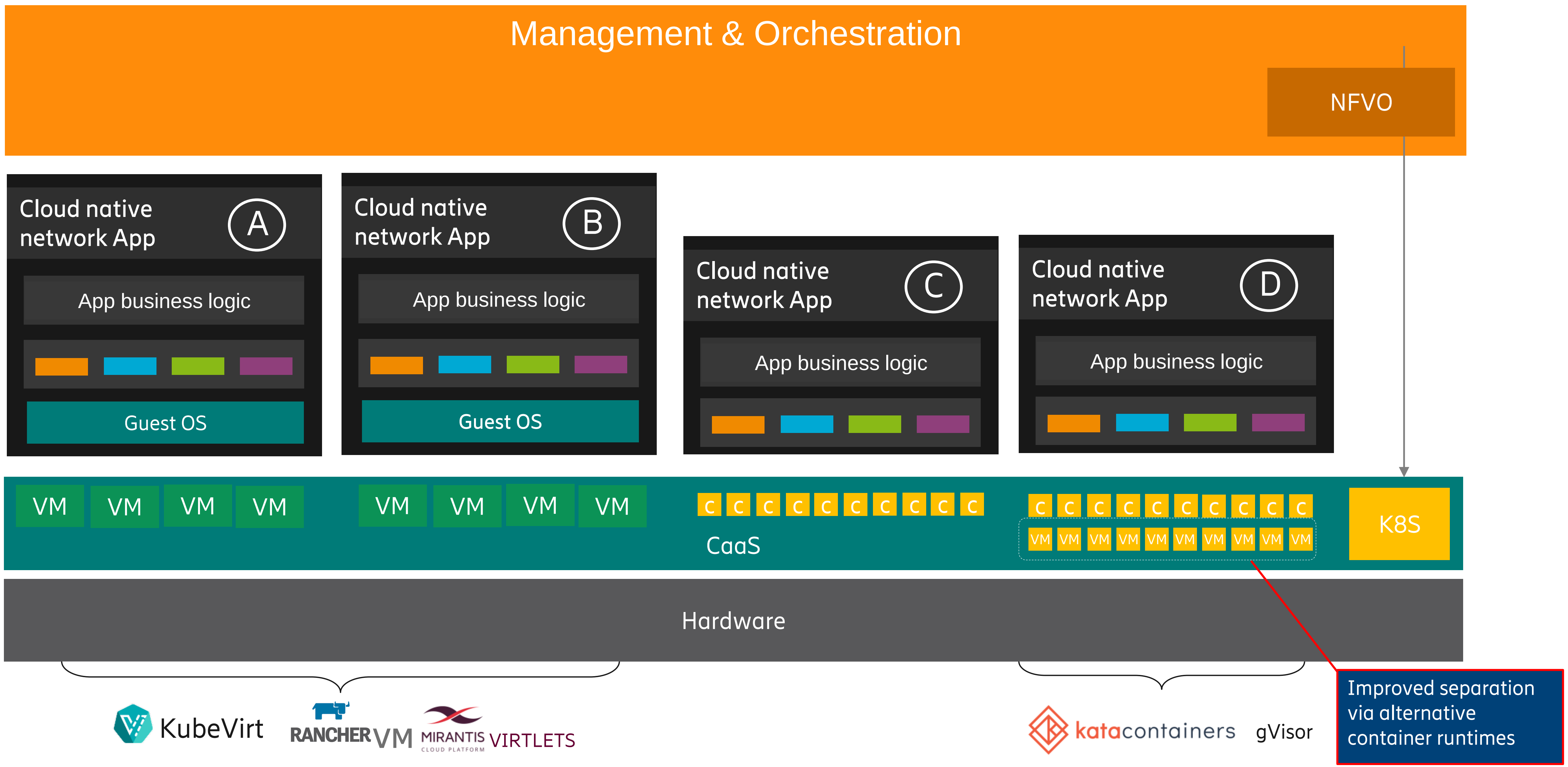
Carrier supported CaaS deployed on top of an IaaS



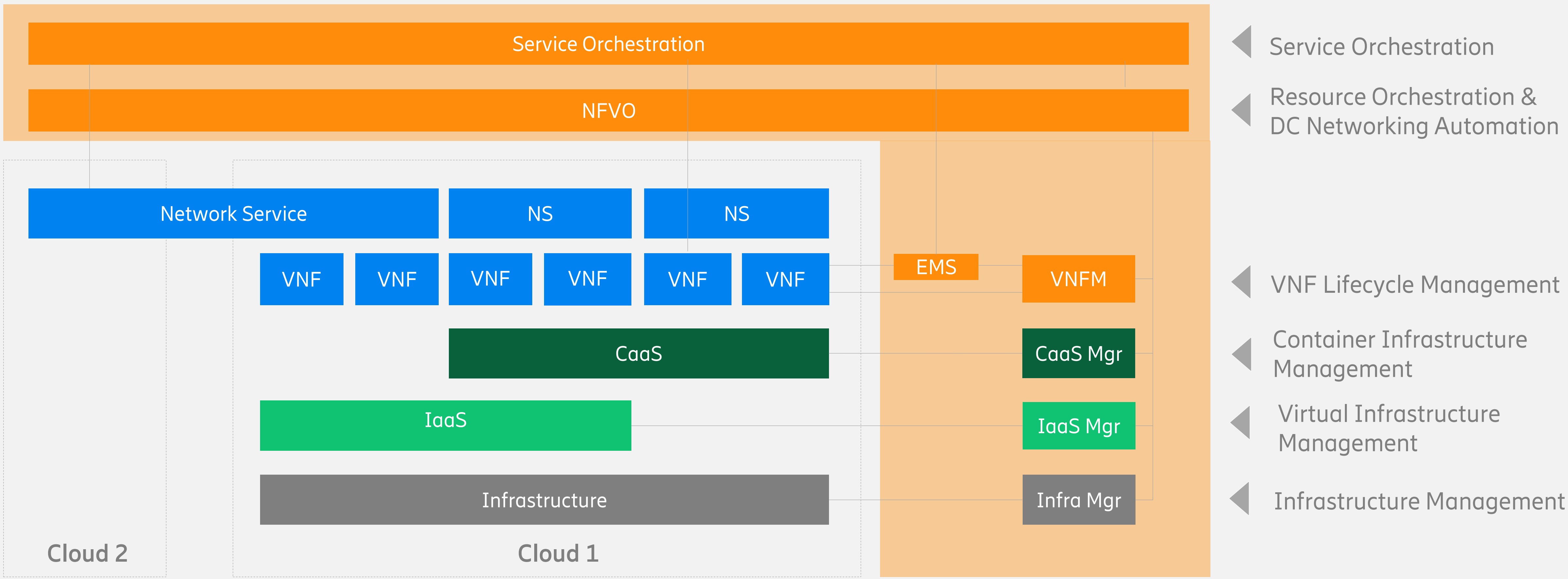
Isolation domains: CaaS deployed on top of an IaaS



CaaS evolution – supporting container and hypervisor workloads



Cloud Native Applications in relation to NFV



Orchestration

Key building blocks to deliver Cloud Native values

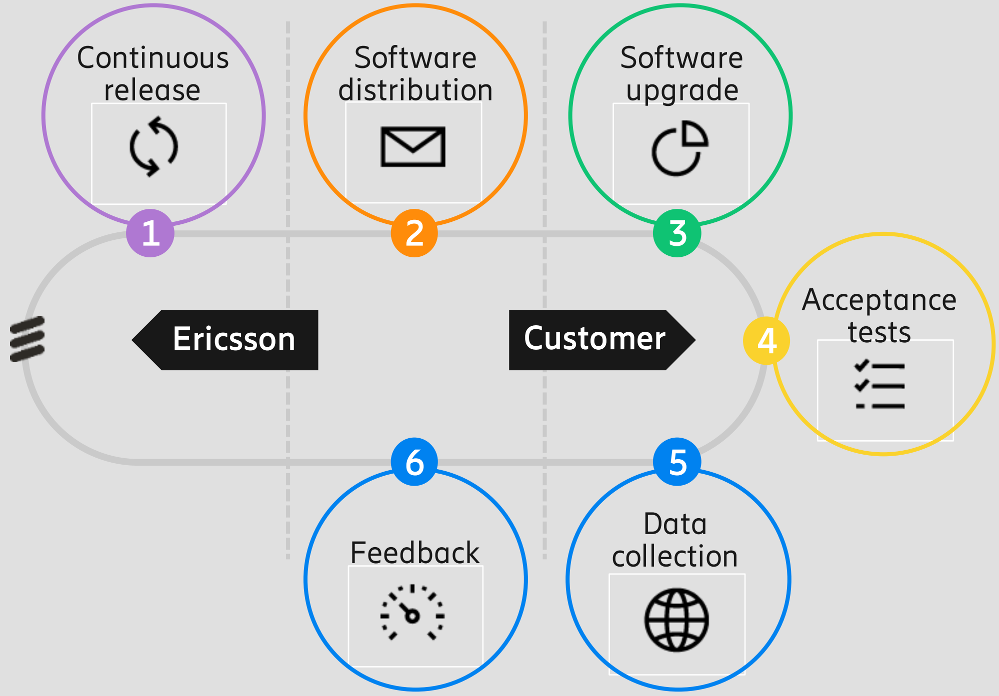
Cloud Native WoW and operations

Orchestration & Automation for K8s and Cloud Native NF´s

Cloud Native NF´s

K8S based platform

Any Infrastructure



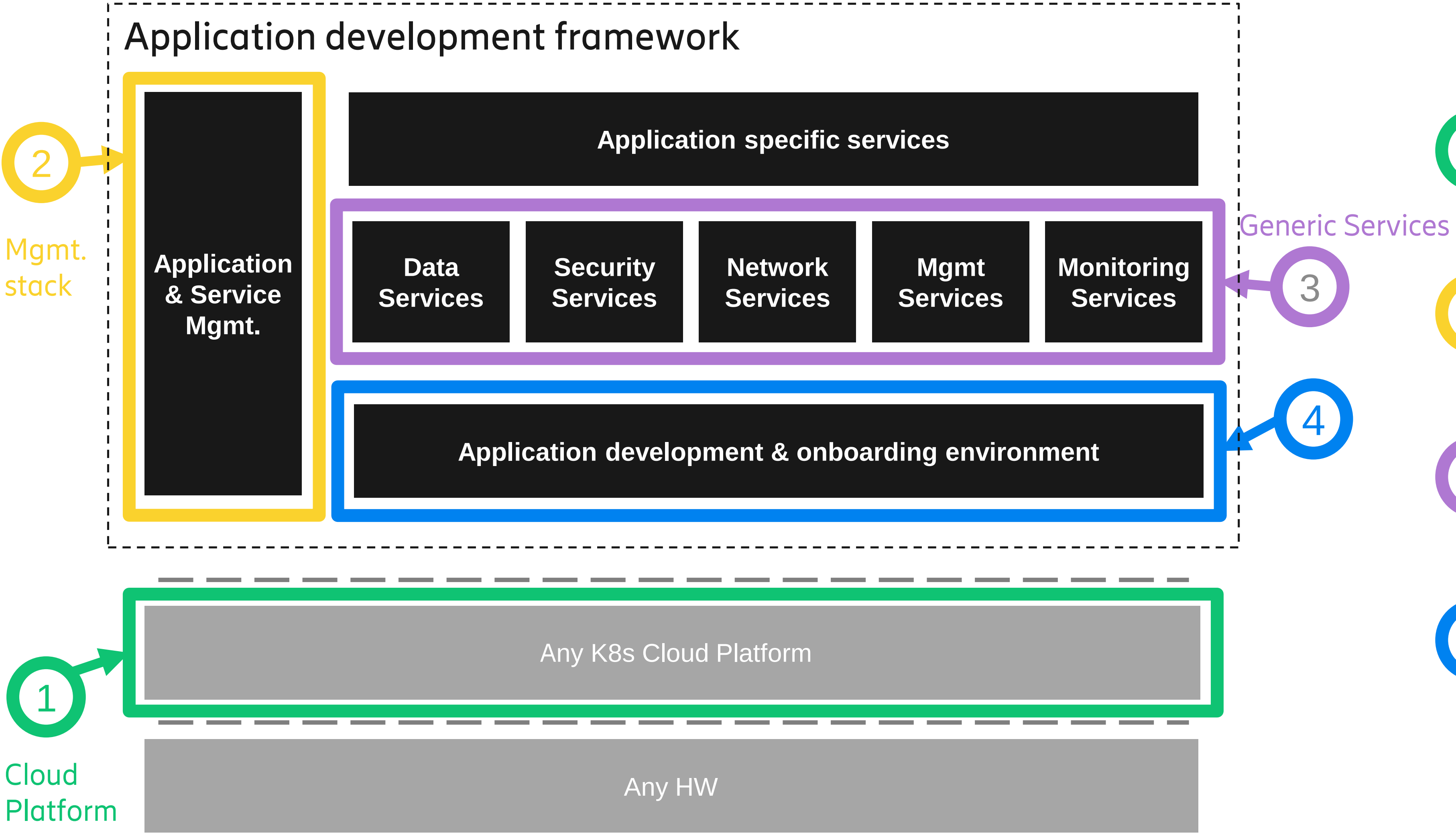
Conclusions

- 5G use cases rely on flexibility of cloud native
- Cloud native transformation is a holistic chalenge
  - **Application** architecture
  - **Cloud** Infrastructure
  - **WoW** & Operations
- Cloud native design principles
  - Agnosticity, disaggregation, resiliency, state optimized, automation
  - NFV ecosystem compatibility is key

Cloud Native Application Design Opportunities & Challenges

Henrik Saavedra Persson

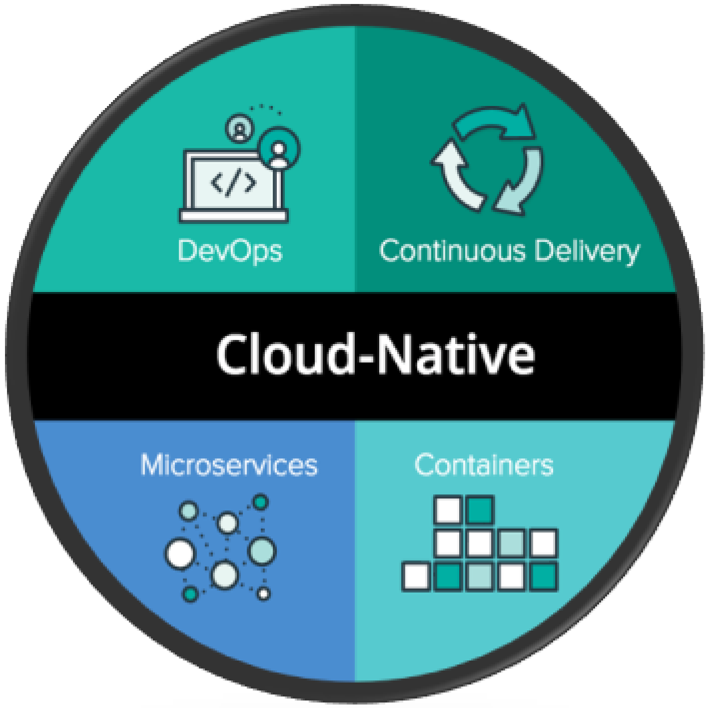
Ericsson application development framework



1. Reference Container platform, K8s based
2. Management & Orchestration functionality for Services and Applications
3. Common (platform type/generic) services for reuse across applications
4. Application & Service development & onboarding environment, tools, DR´s and interface to CI/CD

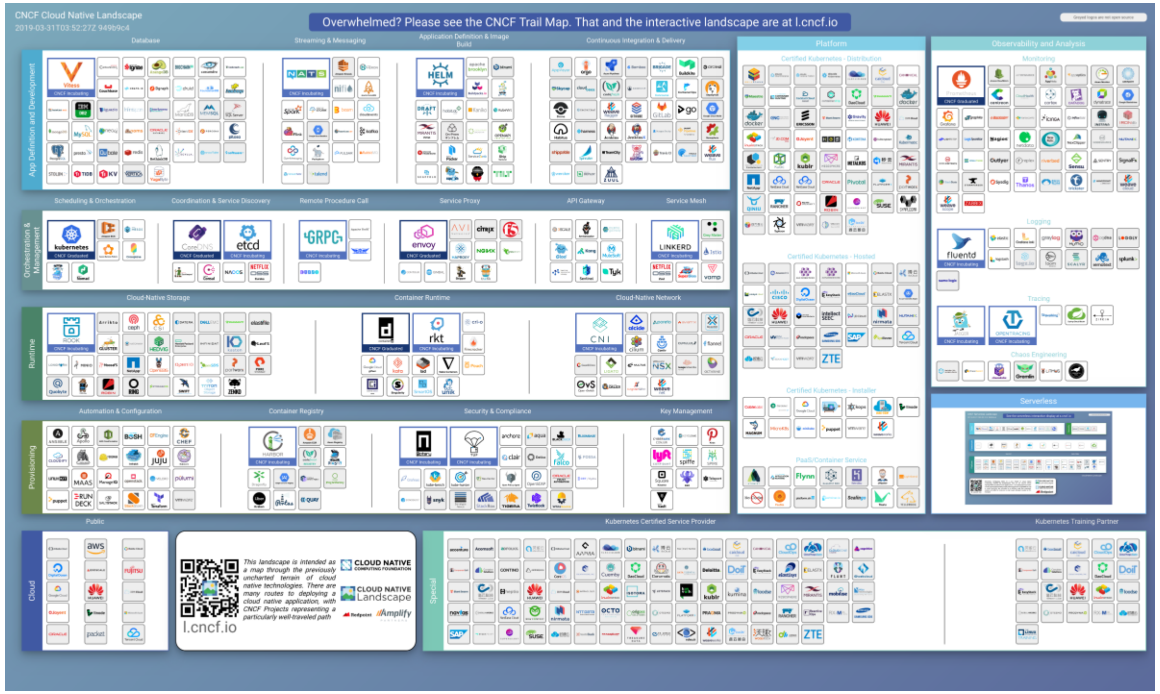
Reuse – what is new?

- Cloud native and MSA with the use of containers
  - High degree of service isolation
- Dependencies are limited to exposed interfaces
- Strict backwards compatibility
  - following semantic versioning



Reuse – CNCF

- Emergence of CNCF has created a new starting point when looking for reuse
- CNCF landscape
  - Structures and provides a map of potential realization
- Transfer of value



CNCF landscape – are we done now?

- Big risk of failure
  - Without a clear view of
    - Context
    - Needed use cases for a selected service
- Architecture even more important
  - Clearly defined architecture principles, design rules and guidelines
  - Guides the different service and functional needs

Architecture first

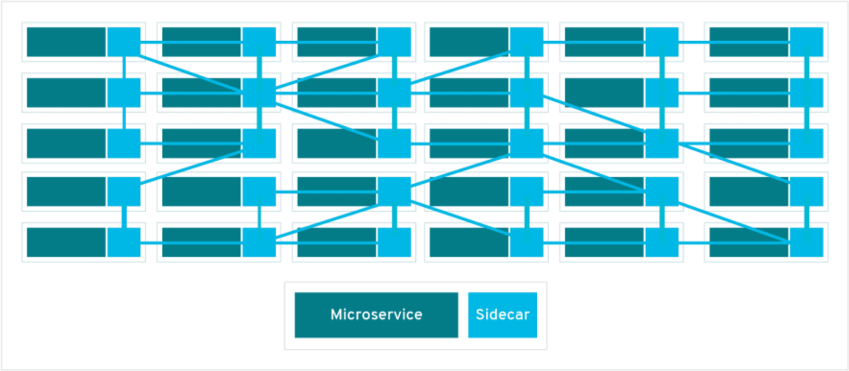
- Identify needed use cases
- How realizations in CNCF fulfill the needs
  - Make the selection
- The architecture need motivates the selected realization
  - Allows reevaluation of the selection
- Continuous evolution of the architecture
  - New use cases
  - Could lead to new realization selection

Architecture first - Service Mesh example

- Define core functional use cases
  - What value a service mesh adds to your architecture for CNAs
  - What principles are important

- Disconnected from a specific realization
  - without being compromised or influenced by support/lack of support from specific realization

- Then make the selection!



Istio

linkerd

Separation of conserns – enabler for telecom use cases

- Rare to find these addressed out of the box
- 3GPP and other telecom standard interfaces
- Minimize the in-house development of the commodity part
- No forking
  - Use of provided backwards compatible and version-controlled interfaces
- Example:
  - Prometheus and building support for exposing 3GPP compliant xml report files

Prometheus + 3GPP A GLOBAL INITIATIVE

- Internal representation should be separated from external representation
- If not applied, possibility of reuse for domain specific use cases will be minimized

How to make the right selection

- Clear definition of needed use cases

- Maturity of the project
  - Fulfillment of use cases
  - Size of the community
  - Backwards compatibility

- How these aspects are handled
  - Gives a snapshot view
  - Longer history gives a better picture

- How to engage in the project
  - Follow
  - Active role
    - Influence backward compatibility
    - Feature evolution

- Not for free
  - Important to build up internal knowledge

Not jeopardizing architecture values

- Created with a specific context in mind
  - Deployment footprint
  - Characteristics aspects

- Typically independent
  - Overlap of functionality
  - Provides value within the scope it was designed for

- Bigger architecture scope
  - Wider end-to-end value
  - Provided without any conflicting aspects

- Best-in-class project
  - Might still fit poorly
  - Selecting a less capable project
    - Better fit given the overall architecture goals

Security – some things cannot be left out

- Expected from open source
  - Securing communication
    - Centered around TLS and OAuth2
    - Evolution of 3GPP moving to HTTP/2
  - A valid evolution of the project

- Source code reuse
  - Security hardening part of the building process
  - Full responsibility and controlled by the in-house team
  - CIS-CAT

Cloud native organization

- An even bigger organizational and cultural change
- Organization is often structured around architecture and processes
- Traditionally bigger teams
  - Larger software projects
  - Responsibility for horizontal tasks

- Much more frequent releases
  - Automate everything!
  - Simplify few tasks not possible to automate
  - Even eliminates some team needs

- Speed is key
  - Any slow down impacts the benefit from cloud native

Organizational acceptance of Open Source

- Build trust and acceptance within the organization
- Bigger issue in areas traditionally solved with proprietary implementations
- Same team structure and the responsibility
  - Constant validation crucial
- Wrongly assumed to be free
- Build competence
  - Dependent on the community makes the organization very vulnerable
  - Understanding the need for evolution
  - Troubleshooting

Having the right in-house talents involved makes use of open source a success

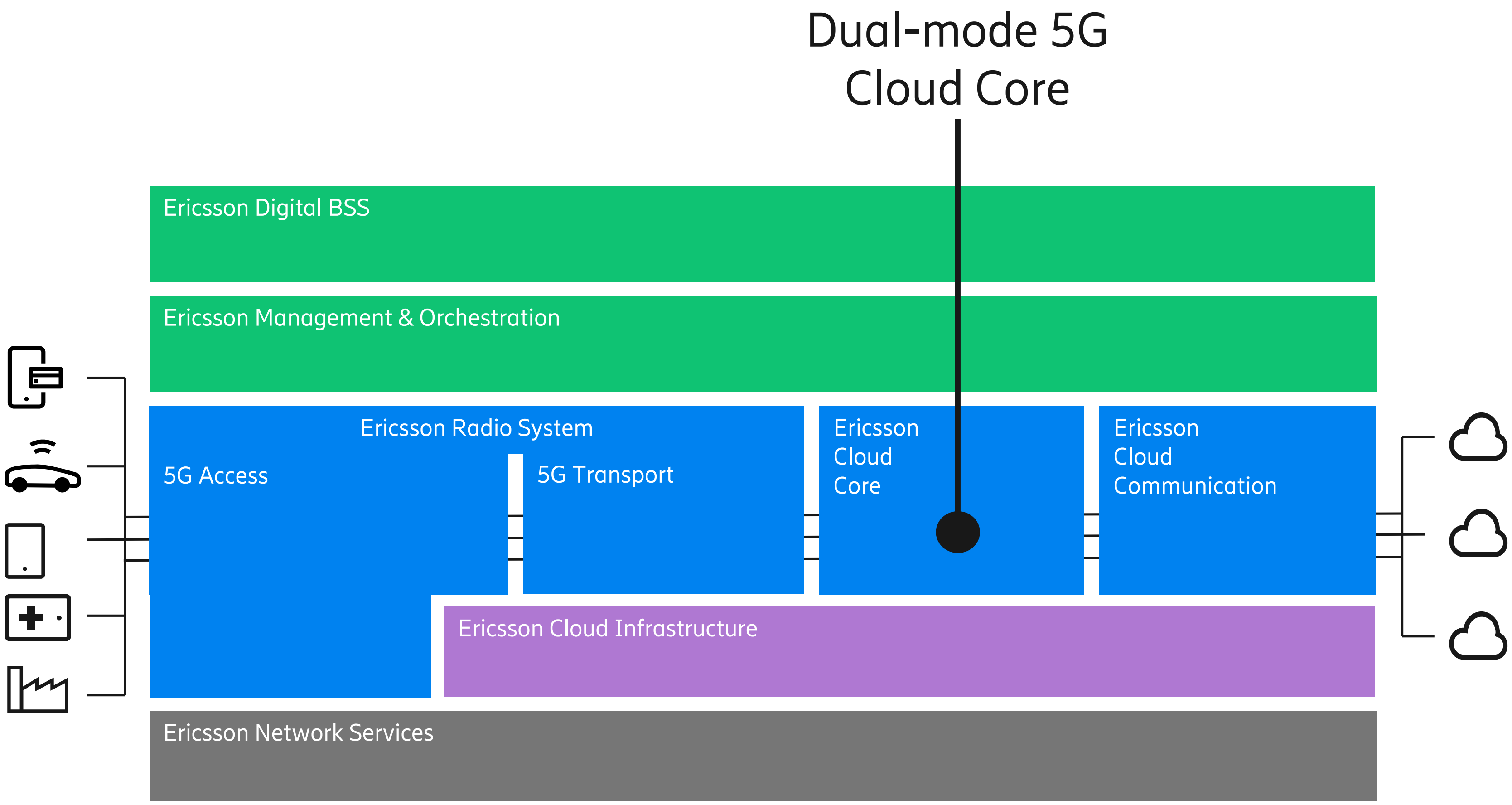
Dual-mode 5G cloud core and network exposure

Open Networking Summit 2019

Jitendra Manocha Strategic Product Manager 5G Core 2019-04-03

ERICSSON

Ericsson 5G platform



Smooth network evolution and deployment at scale

Why 5GC

Optimized interworking with 5G NR

- Reduced signaling
- Lower control-plane latency
- Only standardized solution for NR Stand-alone (SA)

Enhanced service capabilities

- Improved network slice isolation and security (NSSF)
- One Core for all access types (incl fixed)

Service Based Architecture (SBA)

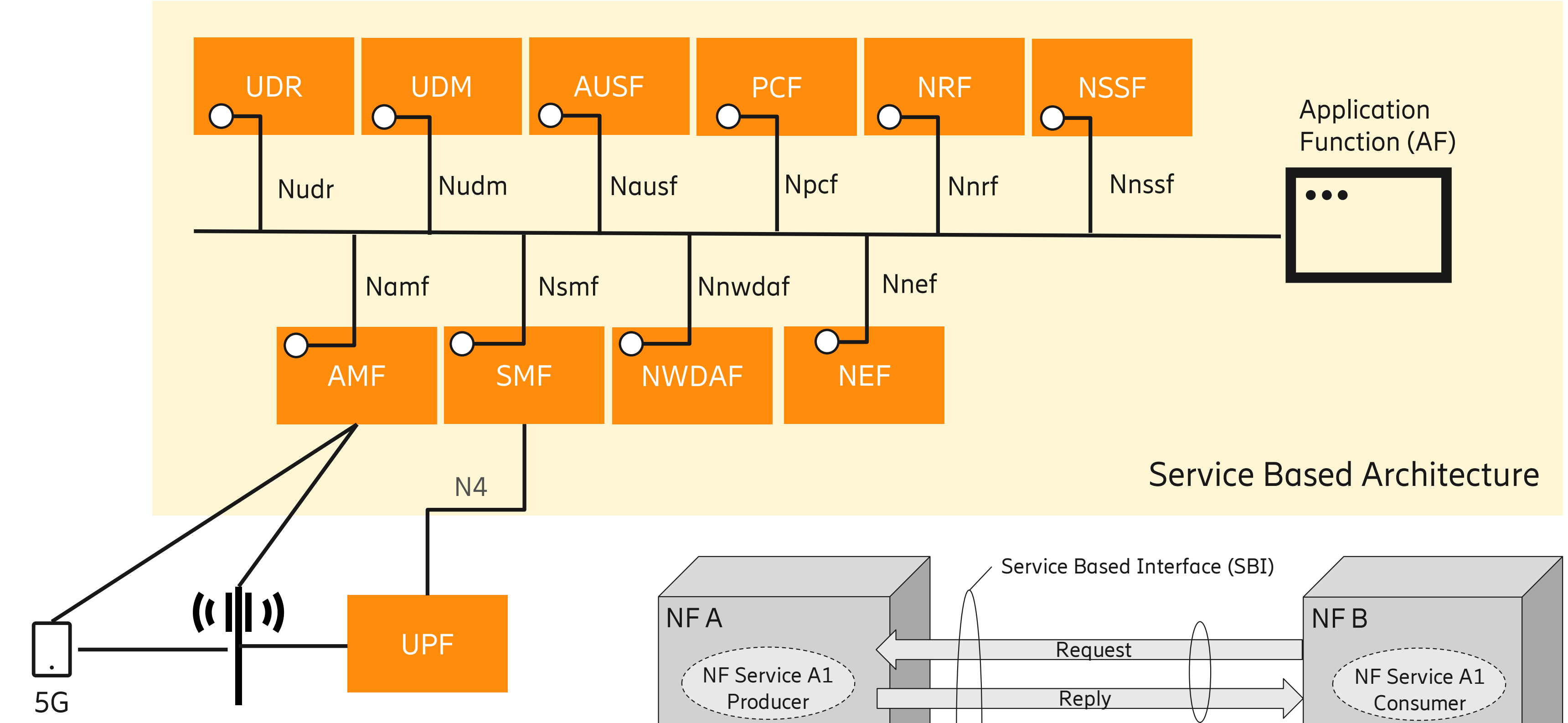
- IT architecture principles
- Fast service creation on 5GC VNFs
- Extensibility and reuse

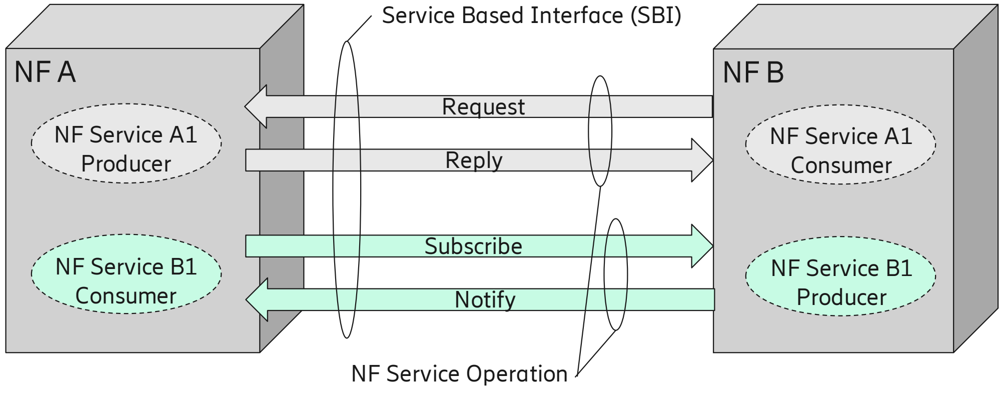
Enable new revenues

- Target architecture for cMTC
- Multiple network slices per device, enabling new use cases
- Support for MTC use cases with non-SIM devices

Service Based Architecture

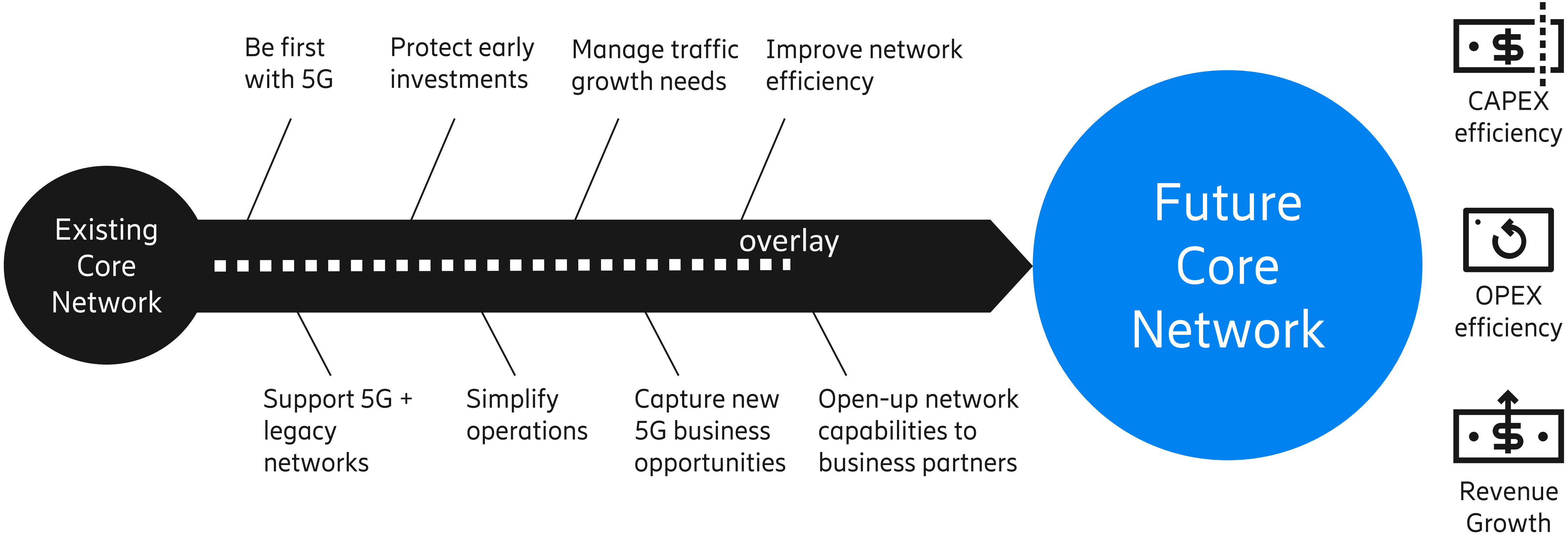
Each 3GPP NF may produce and/or consume one or more services capabilities (3GPP NF Services) through a 3GPP defined Service Based Interface(SBI).



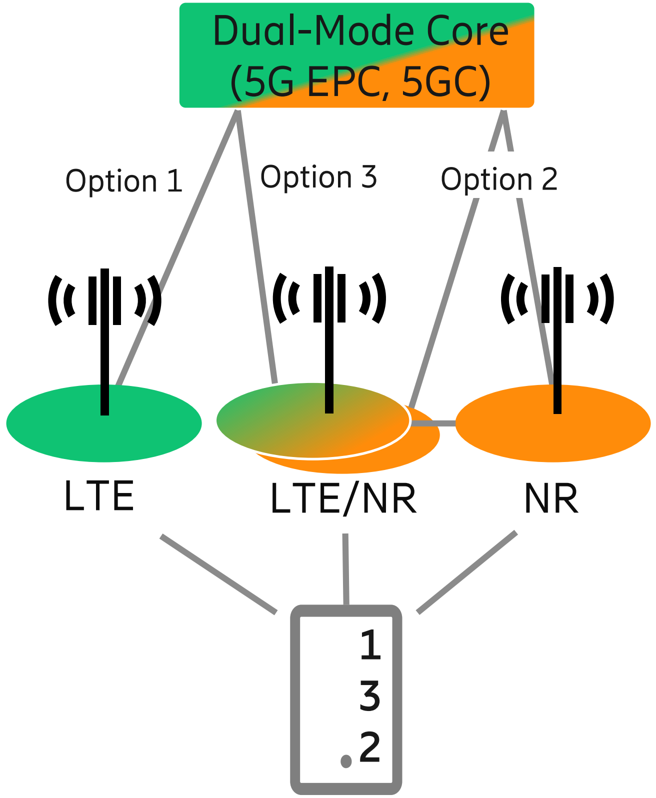


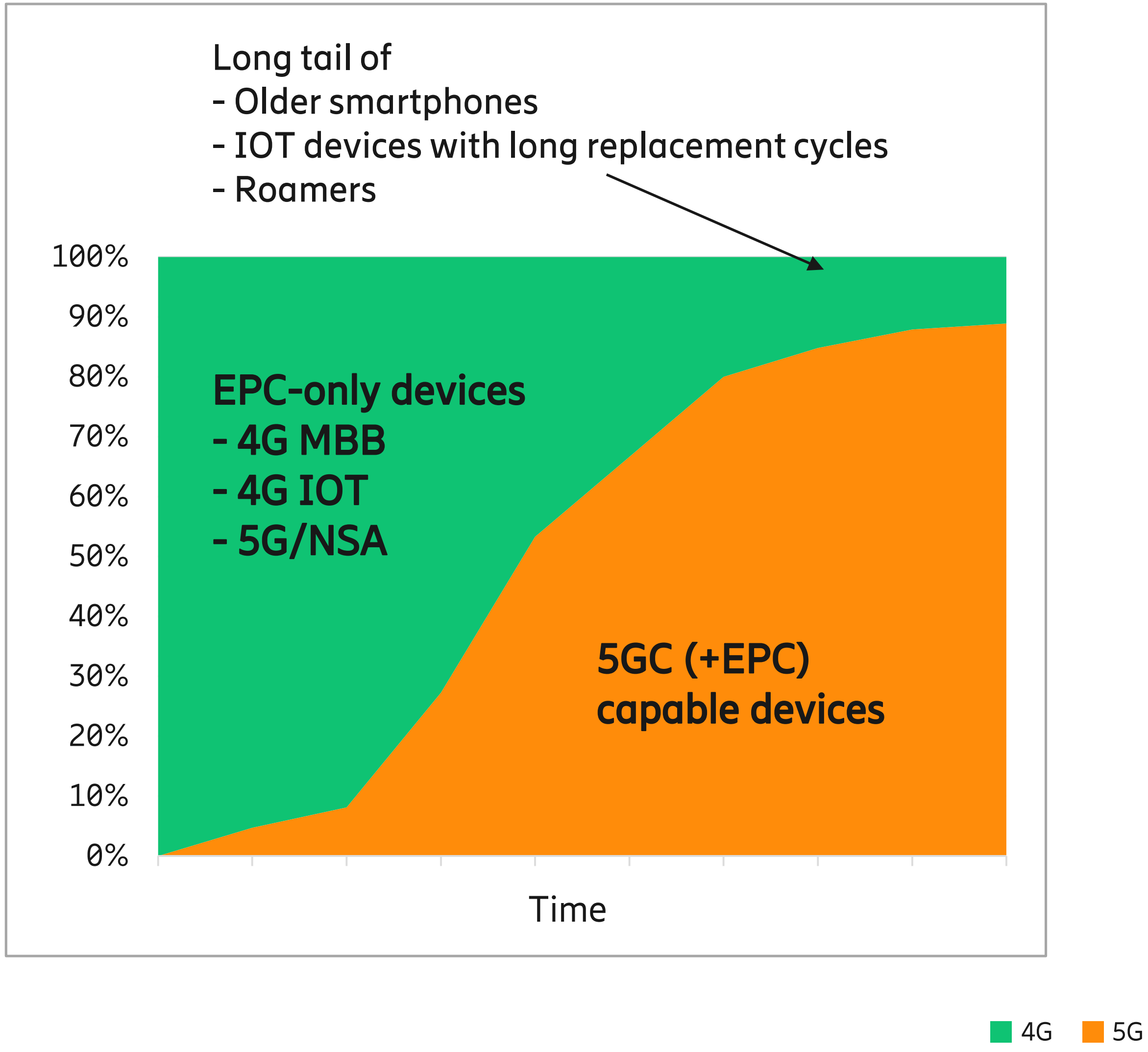
- Implemented using REST architecture style and HTTP/2 over TLS/TCP with JSON bodies.
- Allows fast new service creation
- Extensibility and reuse
- Flexibility, scale & speed of adding new services

Service providers' challenges on the path to 5G



EPC and 5GC will co-exist for many years





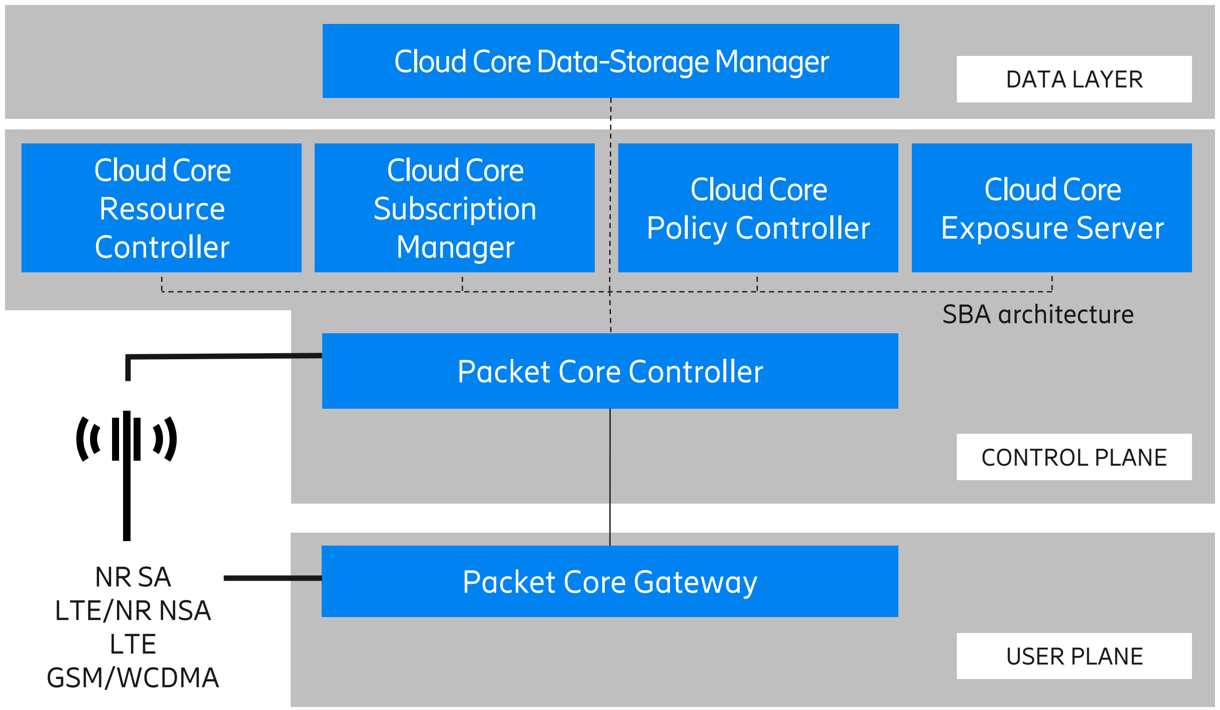
How Dual-mode 5G Cloud Core addresses the challenges

	Challenges	Ericsson's dual-mode 5G Cloud Core proposition
Optimized TCO	Manage growth with **CAPEX** efficiency	**One core network to manage 4G and 5G traffic growth, with investment re-use** – User plane with high capacity, scalability and flexibility to manage 5G high-peak rates – Highly resilient and flexible data storage for network slicing and distributed cloud use cases
Optimized TCO	Reduce **OPEX** and operational inefficiencies	**One cloud native software platform for (5G) EPC and 5GC** – Simplify operations with automation and common O&M across all products – Flexibility to manage 4G and 5G traffic; automated lifecycle and services provisioning
Revenue growth	Grow **revenue** with agility and speed	**Fast time to market for the launch of 5G services** – Network exposure capabilities with open APIs enabling a partner ecosystem for innovation – Network slicing solution customizable per use case

New product family for future networks
- Delivering a dual-mode 5G Cloud Core solution

Solution highlights

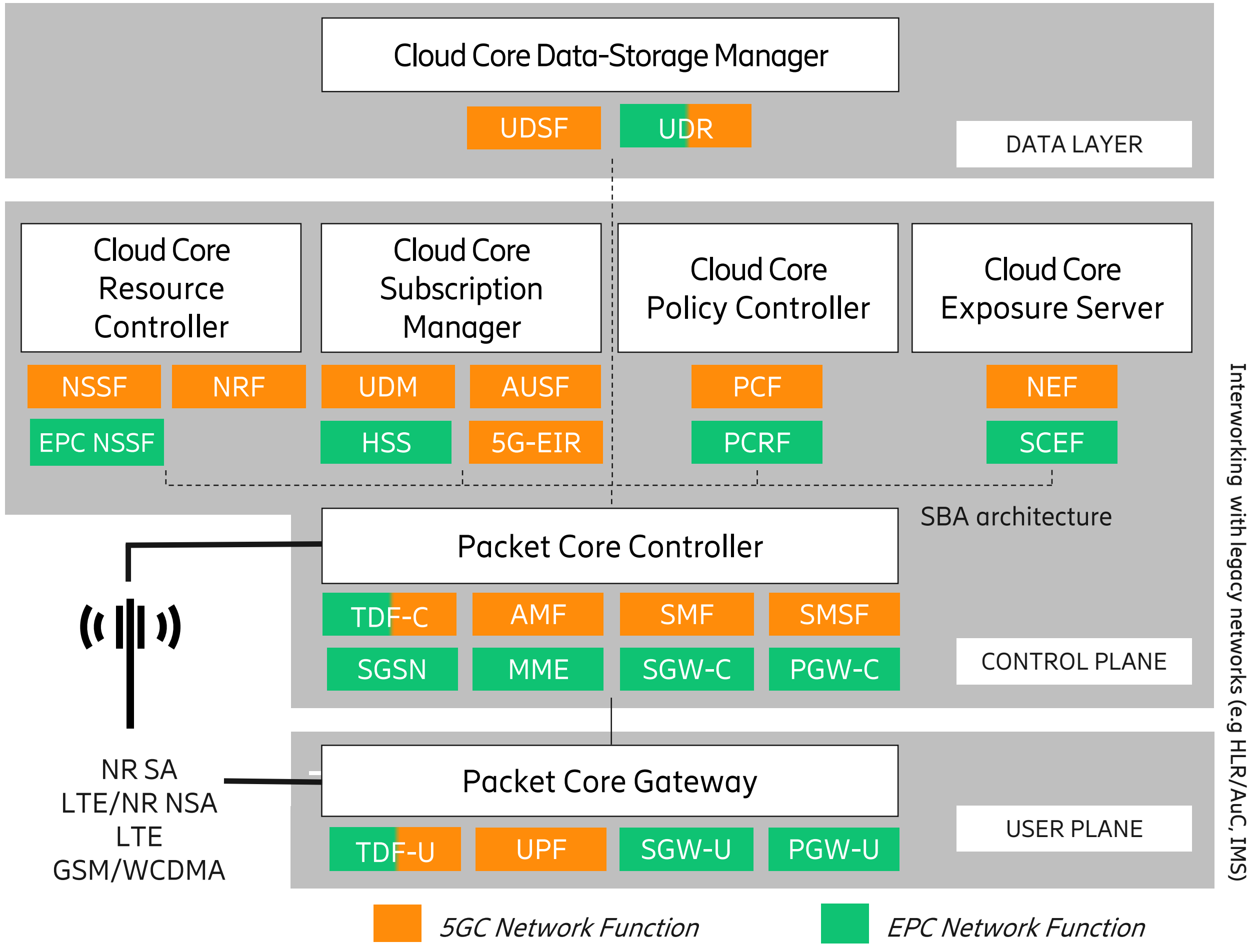
- Single platform for (5G) EPC and 5GC functionalities.
- Cloud native/microservice architecture.
- Automated and simplified O&M.
- High user plane performance and scalability.
- Network exposure capabilities for programmability.



Inside Ericsson’s dual-mode 5G Cloud Core

Network Functions (NF) grouped per network services, allowing deployment flexibility and a smooth evolution to 5GC.

- Cloud native 5GC NFs microservice based.
- EPC NFs re-architected into cloud native with secured feature parity.
- Fully interworking with legacy networks and NFs.

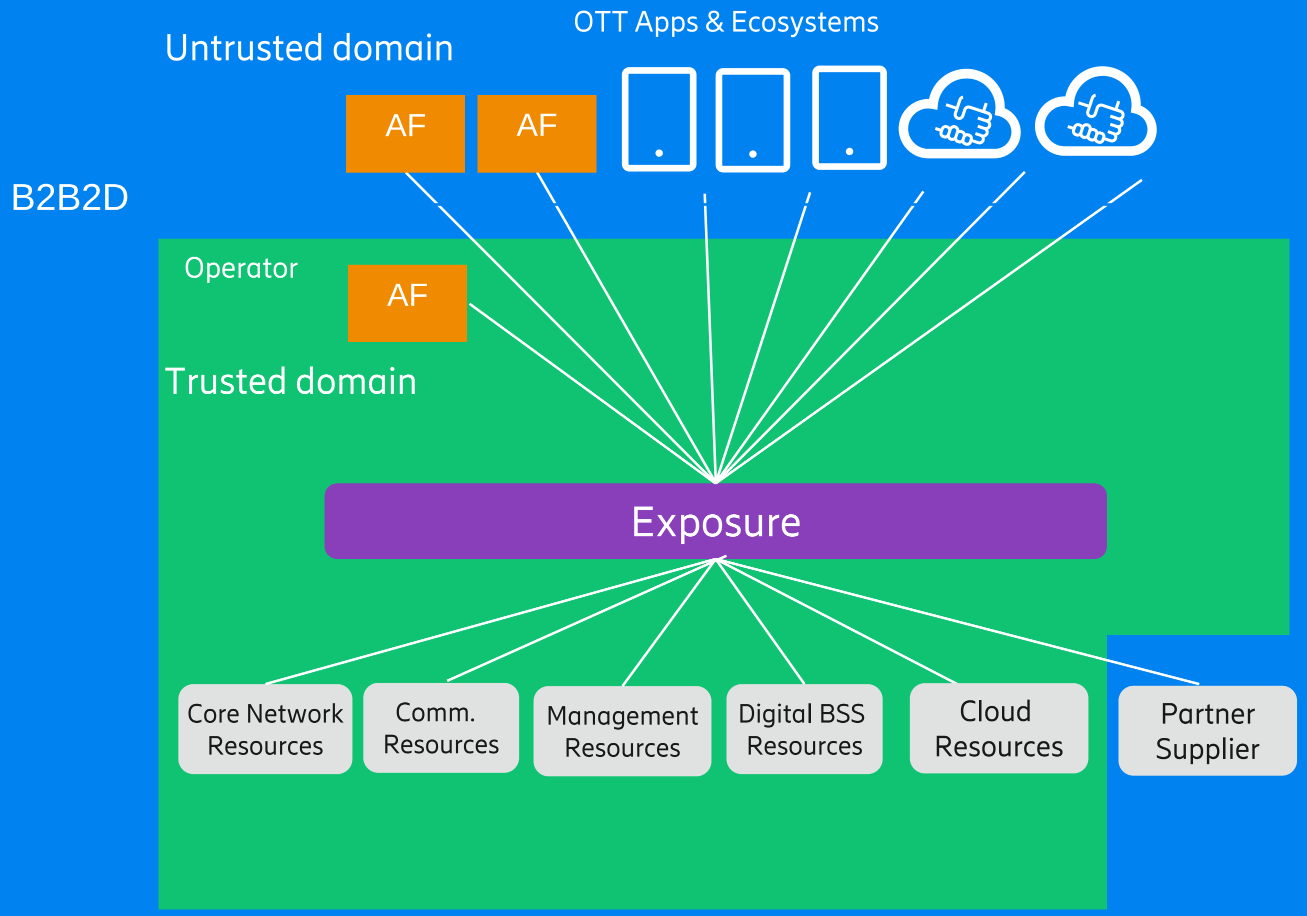


CAPEX
OPEX
REVENUE

Key enablers for 5G programmability

These building blocks fulfill the requirements of programmability, making the network responsive to the future needs of high agility, flexibility and extensibility.

Exposure – Big Picture



- Monetization of the network capabilities
- Hide the complexity of underlying network
- Secure/controlled access of network to external application functions

Exposure evolution

IoT 4G EPC	Massive IoT: NB-IoT/Cat-M1 use cases. Limited mobile broadband capabilities exposure	**Example:** Smart meters, connected water, event monitoring
5G EPC	eMBB and FWA use cases covering NSA-NR	**Example:** Gaming use cases requiring high throughput
5G Core	All types of use cases: mMTC, eMBB, cMTC. Full programmability potential: Exposing capabilities from any NF in 5GC	**Example:** Edge computing use cases (requiring low latency) like immersive video/AR/VR
5G Core >>>	mMTC, eMBB, cMTC use cases assisted with advanced machine learning, network slicing	**Example:** AI/analytics based closed-loop use cases

Cloud Core Exposure Server

Centralized API management and exposure capabilities, enabling:

— Standard and tailored APIs per applications and use cases

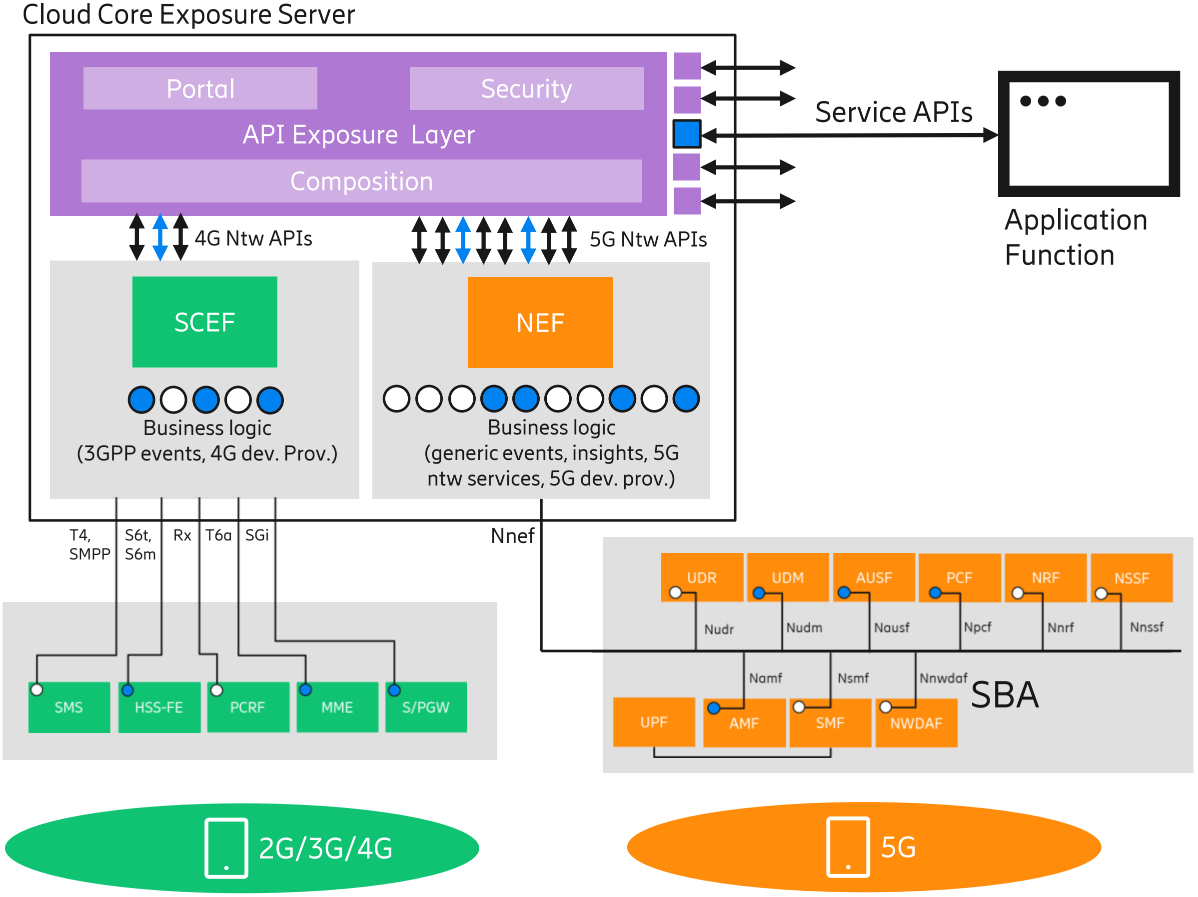
— The ability to open 4G and 5G Core network capabilities to third parties.

— Increased network automation for operational efficiency.

New revenue

Network programmability

Read the Ericsson report: [Programmability in 5G Core networks]



API Overview

Standard technical use cases (3GPP R15)

1. Application Function influence on traffic routing (Nnef_TrafficInfluence)
2. Event exposure* (Nnef_EventExposure)
3. PFD management (Nnef_PFDManagement)
4. Resource Management of Background Data Transfer * (Nnef_BDTPNegotiation)
5. AS Session with required QoS * (Nnef_ChargeableParty)
6. Change the chargeable party* (Nnef_AFsessionWithQoS)
7. Parameter provisioning* (Nnef_ParameterProvision)
8. Application triggering* (Nnef_Trigger)

Green field for innovation, new APIs are possible faster then ever,

*APIs Available in 4G and enhanced in 5G

Optimized data transfer for automotive

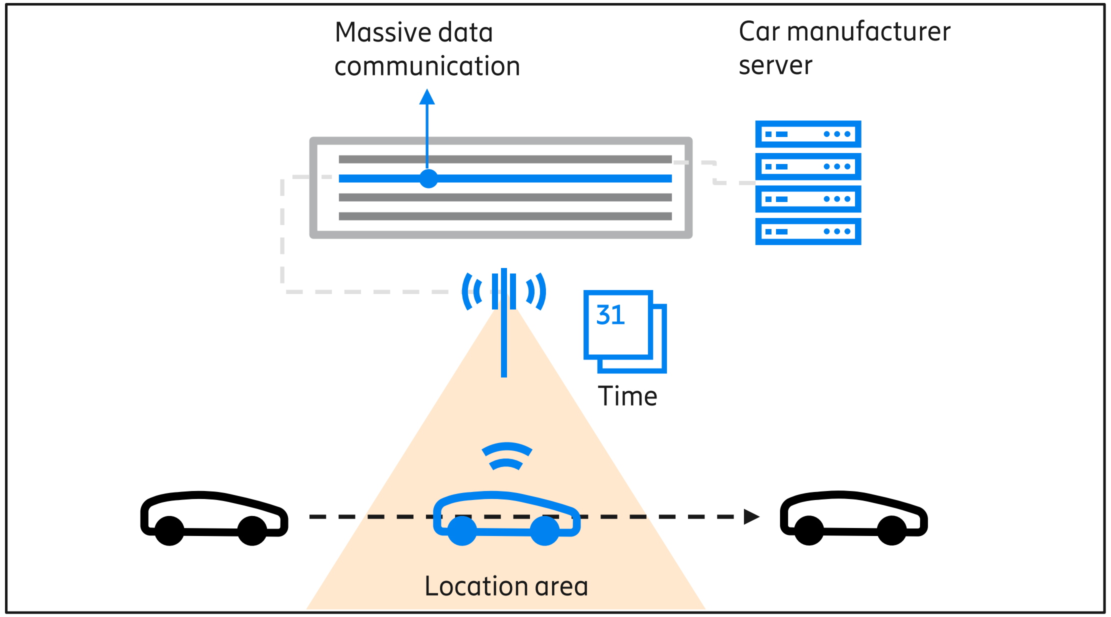
5G segment: Enhanced Mobile Broadband

Business model: B2B

Operator's customer: Car manufacturer

Use Case: Backgorund data transfer of massive data

Description: Group of connected cars can automatically download massive amount of car's maintenance data during a specific time window.



Key Take away

- Service Based Architecture is the change in right direction, specially when implemented using cloud native and micrioservice architecture. As it paves the foundation for the next generation core networks.
- Dual mode core will enable operators and enterprises to save opex, capex and simplify the network capabilities to fuel the eco-system development.
- Open APIs are enabling programmability, new capability from the network can be exposed faster than ever.
- Co-innovation (Open innovation) is key to succeed in this area as many usecases have not been conceptualized yet.

Visit our booth to learn more and share your feedback, insights, ideas!!!!!!!

Cloud Container Distribution

Solution Line Distributed Cloud
Solution Area Cloud and NFV Infrastructure
Business Area Digital Services

Content

- Container Introduction & Customer Challenges
- High-level Product Overview
- Architecture, Networking
- Deployment Models and Packaging
- Summary

Cloud Container Distribution

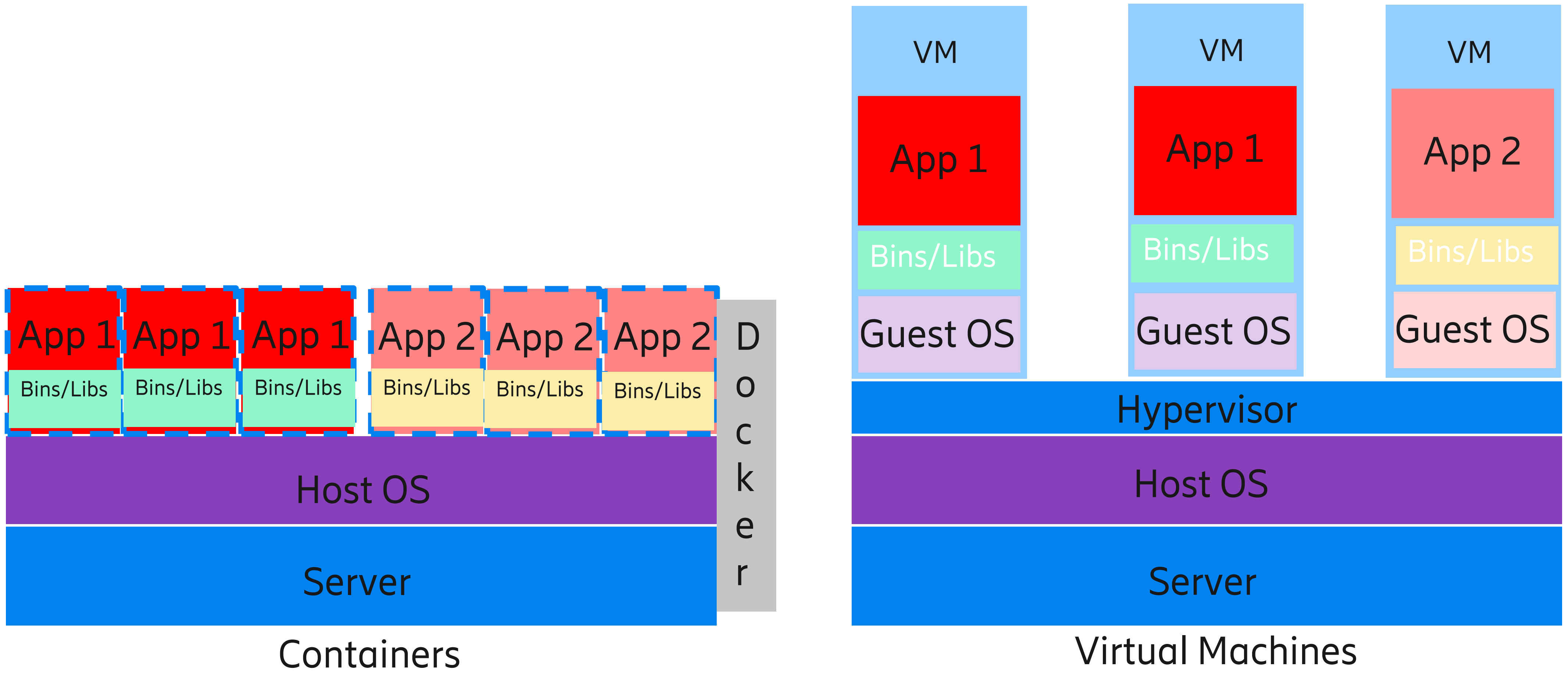
Container Introduction &
Customer Challenges

Containers

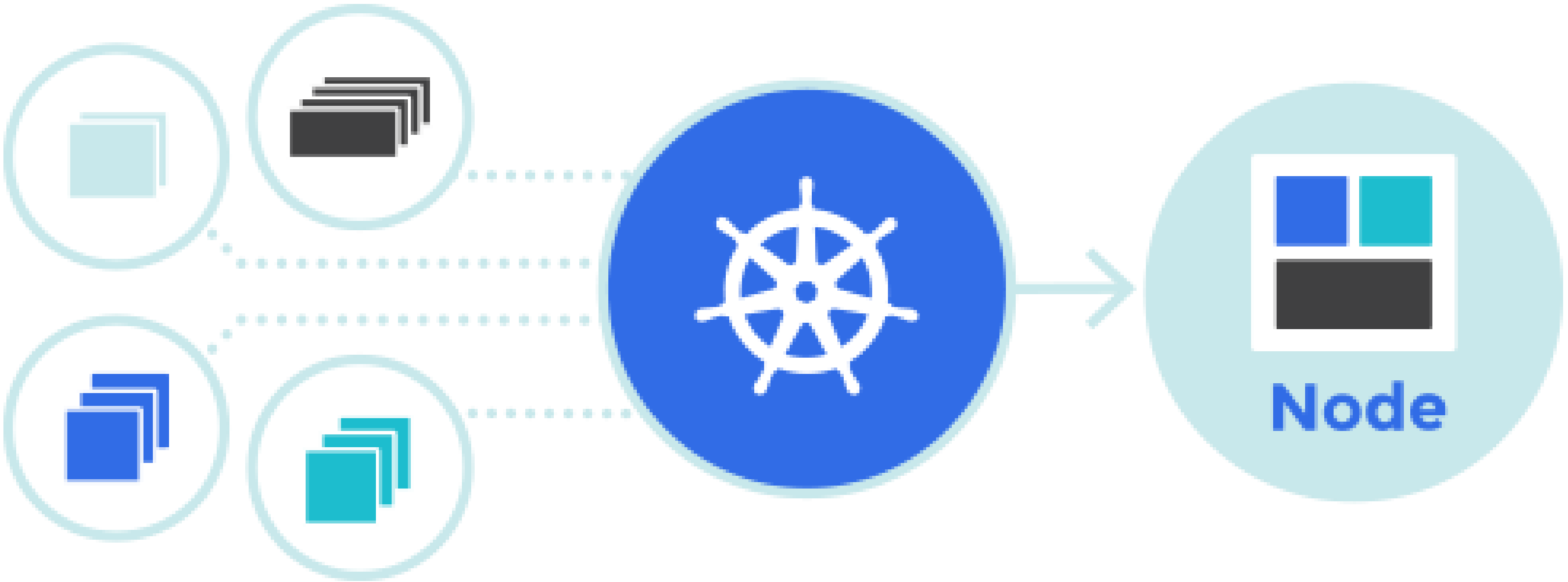
The foundation for the next generation of applications

- Support cloud native applications implemented with a microservices architecture
  - Decoupled, separately scalable services that can have separate life cycle management
  - Faster deployment compared to Virtual Machine based spin-up
- Common standardized application deployment (Docker image format)
  - Provide complete environment needed for application to execute
- Optimize resource usage (CPU, memory, network)
- Integration with CI/CD
- Isolation model using Linux kernel features
- Container runtime & image formats open standards are governed by OCI (Open Container Initiative)

Containers and VMs

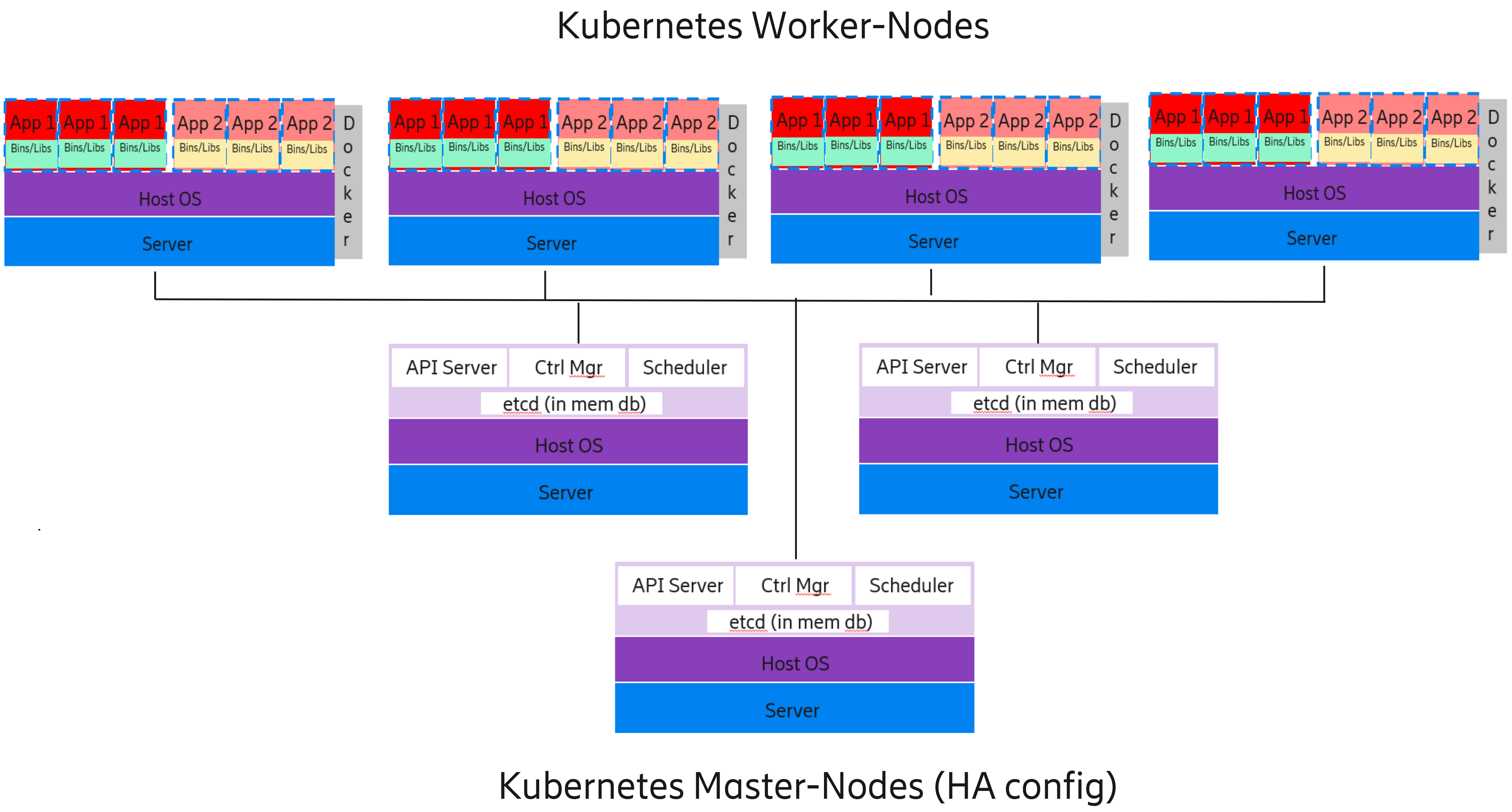


Container Orchestration and Management



- Container Orchestration and Management support using containers at production scale
- Open source Kubernetes is the de facto industry standard
- Rich community of open source value add plug-ins and components
  - Through the governance of the Cloud Native Computing Foundation (CNCF)

Containers at Scale via Orchestration



Customer Challenges

Build Your Own

- Working directly with open source requires resources for building, testing, integration
- Building and Packaging
- CI/CD pipeline
- Versioning and compatibility
- No single standard Kubernetes installer

Vendor Lock-in

- Some commercial vendors require you to use their distributions for technical support, muting the benefits of open source

Navigating the Complexities

- Navigating the complexities of the CNCF landscape
- Selecting the components and plug-ins that support your use cases
- Continuous addition of new components

Security

- Hardening
- Tracking vulnerabilities and patching

Operational

- Lifecycle tasks: backup, restore, upgrades, scaling
- Metrics and Monitoring

Cloud Container Distribution

High-level product overview

Cloud Container Distribution (CCD)

Kubernetes Distribution

Includes Best of breed plug-in

components

Advanced Features

Handpicked CNI plugins

Multiple Network Interfaces,...

Leveraging standard interfaces

Various open source software components

Ansible, Helm, Docker registry, Prometheus,...

Fast Lifecycle Management

Install / Uninstall

Scale in/out

Upgrade

Design and Development Principles

Unmodified open source components

Additional value-add components to complete a

telco-grade offering

CNCF certification

Tested and validated with CNCF

Kubernetes conformance tests

"Upstream first"

Open source community participation:

- Kubernetes Network SIG
- Kubespray
- Multus
- Helm
- And more

Flexible infrastructure deployment options

Image-Based, Pre-provisioned servers

Design and Development Principles

Robust CI/CD

All components bundled as a single package & put through the robust CI/CD system tests.

Every Kubernetes upstream is system verified with the packaged components

Agile Release Cadence

Agile release cadence for internal stakeholder

Explore possibility of such release for customers as well

CNCF E2E & Conformance tests

200+ CNCF Kubernetes E2E tests are verified every release

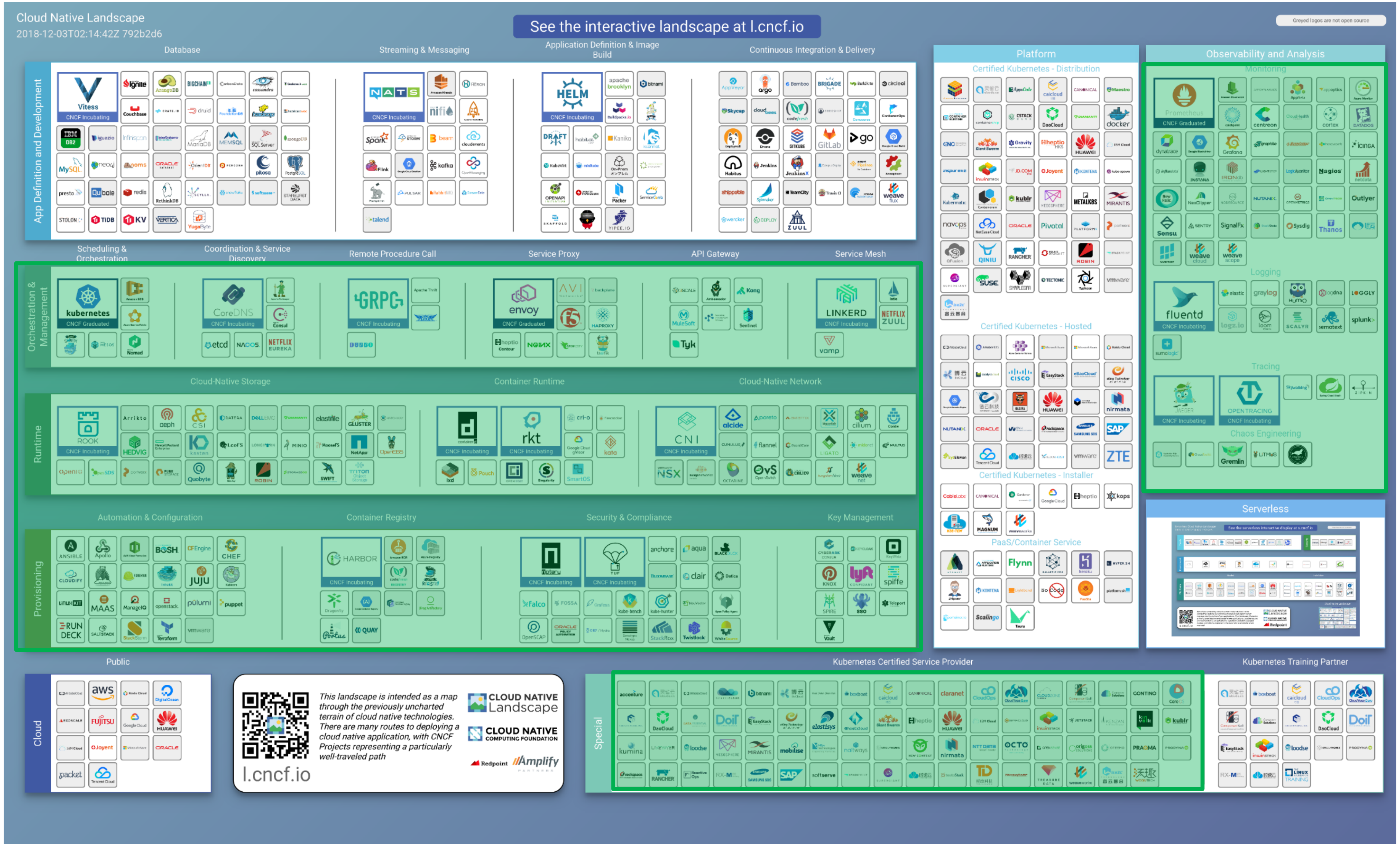
200+ CNCF Conformance tests are verified every release

Security Hardening

Run vulnerability reports

Provide Telco grade security capabilities

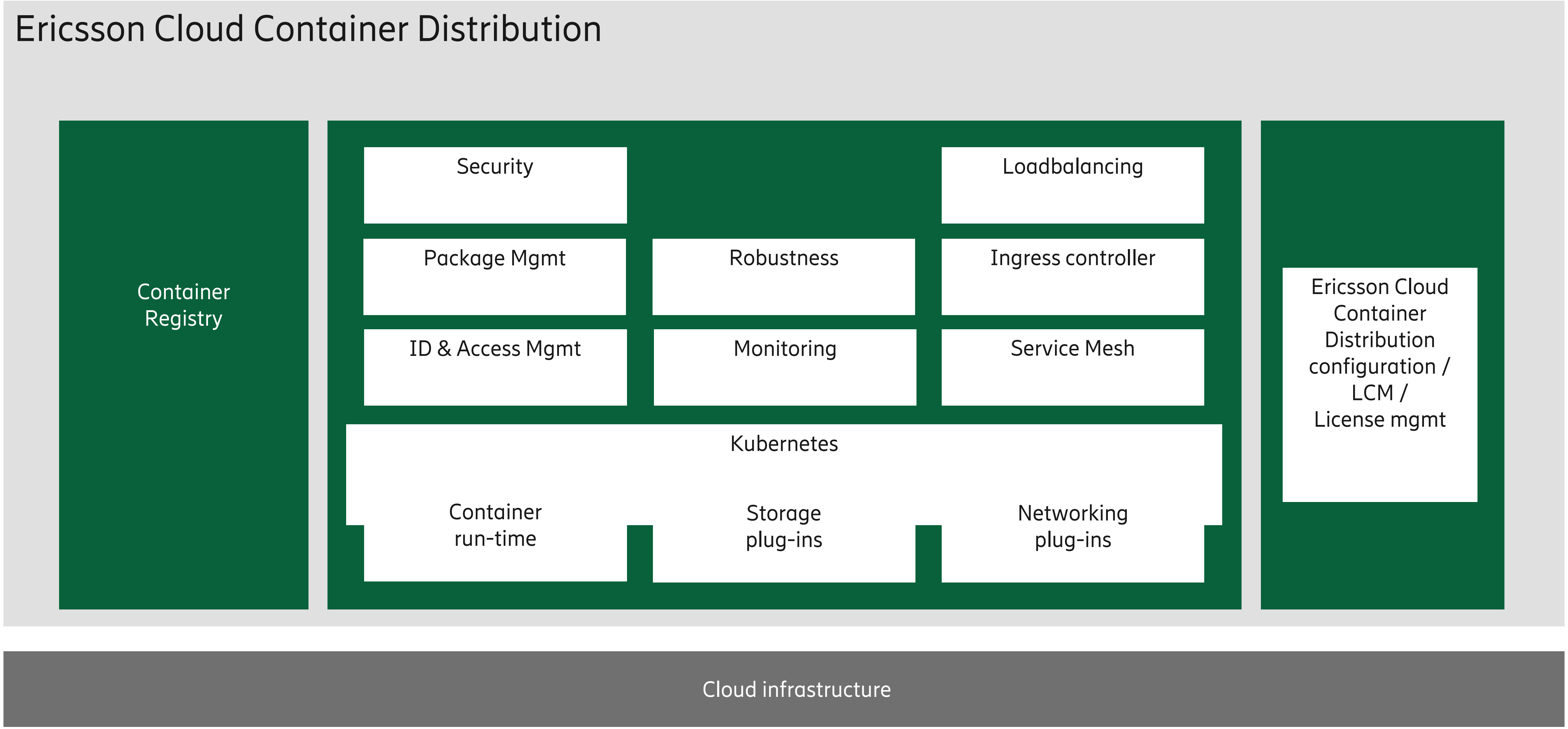
CNCF Cloud Native Landscape: CCD scope



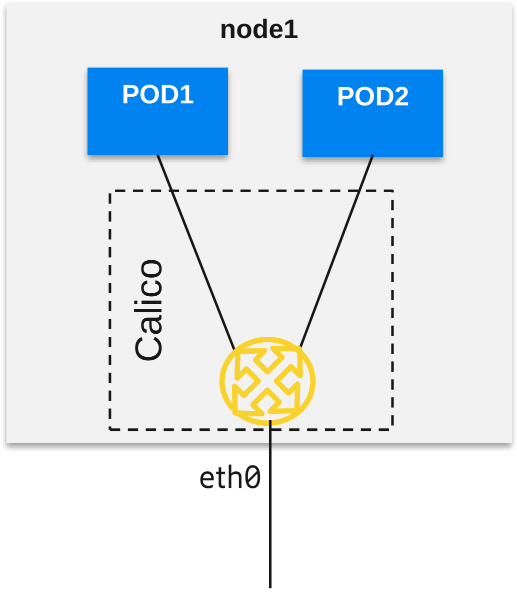
Cloud Container Distribution

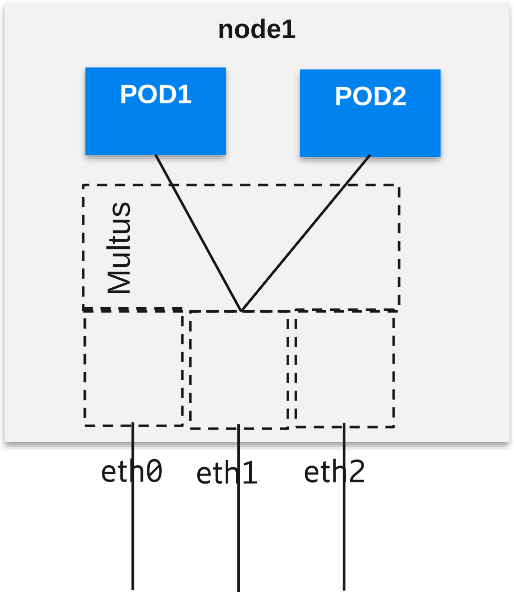
Architecture, networking

Enabling container workloads



Flexible networking with CNI plugins

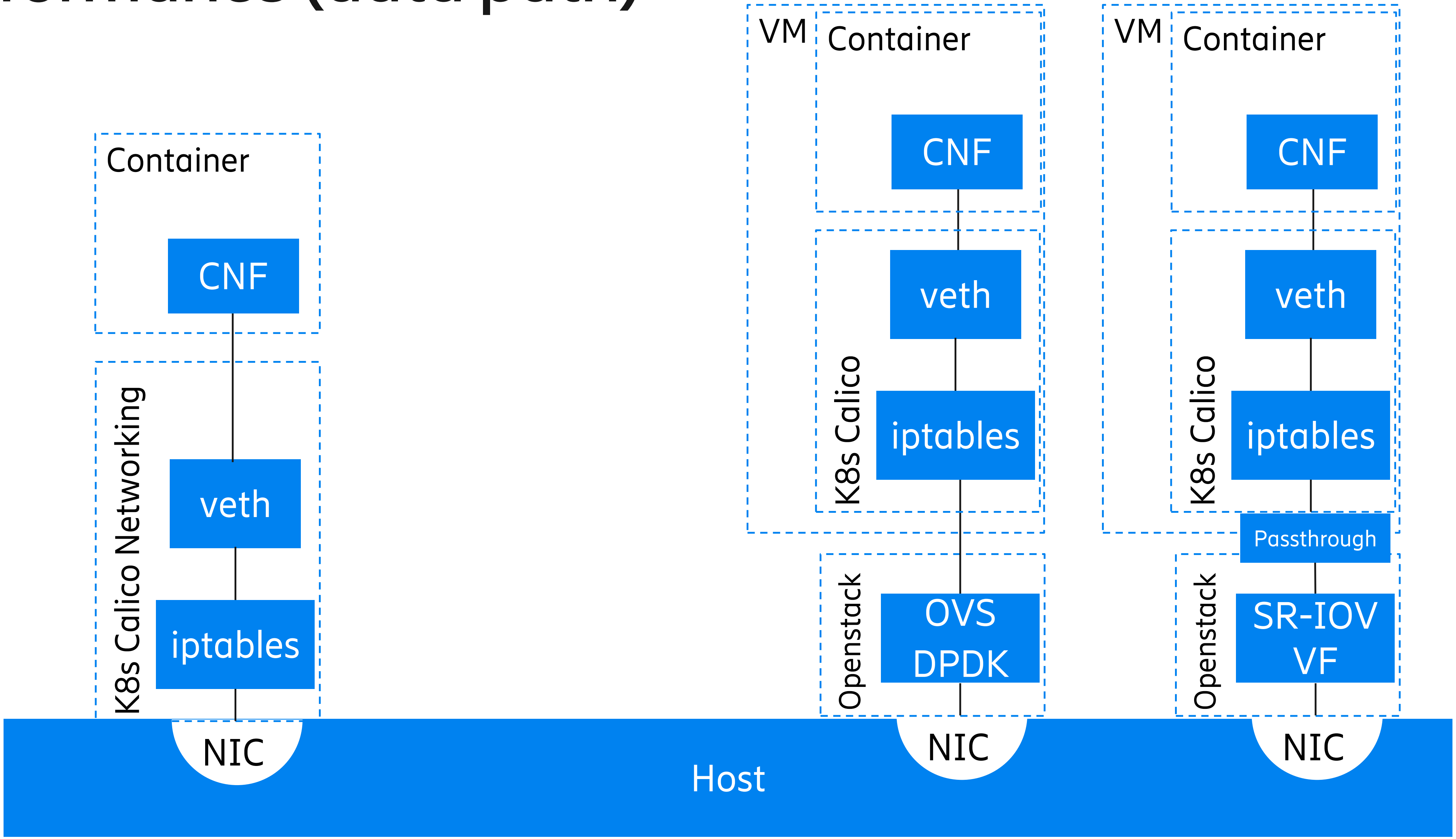




- Primary CNI assumed by Cloud Container Distribution
- Flat routed network using BGP

- Meta CNI enabling several network interfaces towards application workload

Performance (data path)



Components Evaluation

- Industry trends/Telco use cases
- Technological advancements
  - Ex: microVMs, Service Mesh

Solution	Kubernetes network Policies	IPv6	Layers used	Networks	Note
Calico	Yes	Yes	L3 (IPinIP, BGP)	Many networks on cluster. BGP no overlay	Scalable, may integrate with infrastructure.
Multus	Yes	Yes	L2	Many networks on the same cluster	Supports multiple network interfaces
Macvlan	No	Yes	L2	Direct networking	
SRIOV	No	Yes	L2	Direct networking	
Flannel	No	No	L2 (VxLAN)	Many networks on same cluster with multi daemon	

CCD – Kubernetes & IPV6

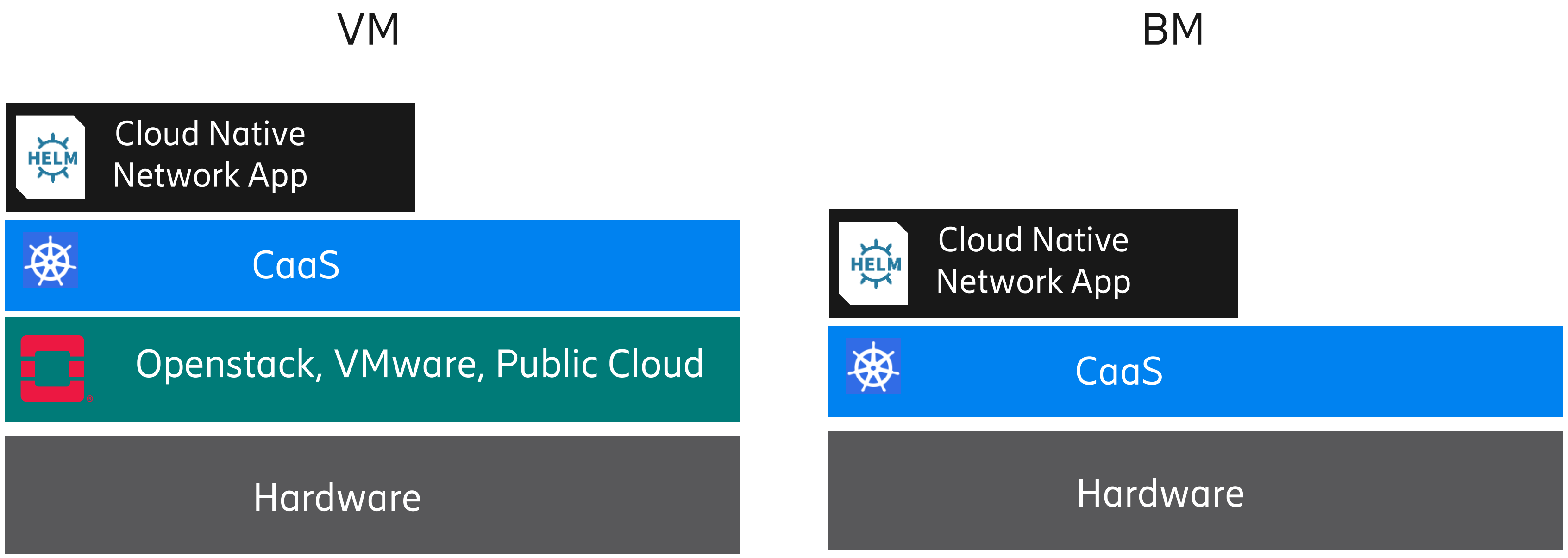
- IPv4 networking supported from the beginning
- Kubernetes added IPv6 (single stack only) option as an alpha feature with the 1.9 release
- CCD team is working within the Kubernetes project, driving the system design and code contribution to bring the dual-stack IPv6 support to Kubernetes in a future release (expect 2020 H2)
- In the interim, IPv6-only clusters are achievable with NAT64.

Cloud Container Distribution

Deployment Options and Packaging

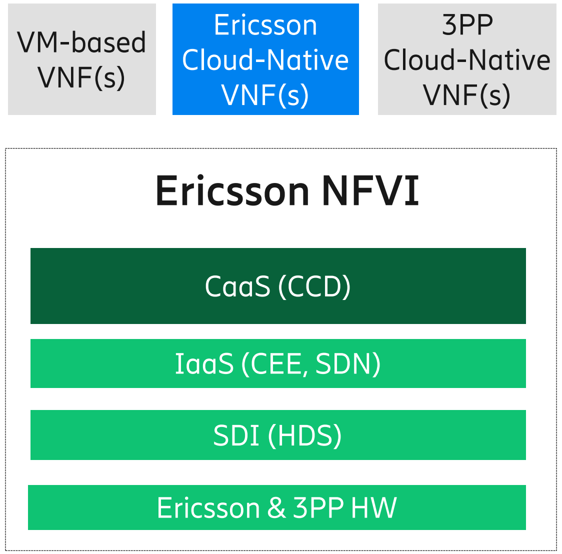
Containers on BM vs in VM

- In the BM case all containers 'share' the same kernel and the same hardware elements (for example, network interfaces)
- In the VM case isolation between different containers is provided by the hypervisor
- Main differences:
  - *LCM*
  - *Security and isolation*
  - *Performance / footprint*
  - *Machine configuration*



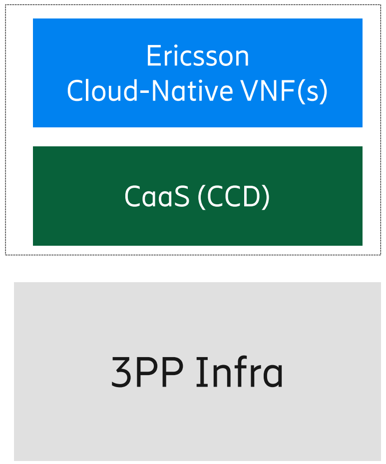
Supported Deployments

Container-enabled NFVi
for **all** Cloud-native applications



- Integrated feature in Ericsson NFVI
- Supports automated deployment onto VMs with image-based deployment (IBD)

Enables cloud-native **introduction**
in 3PP infrastructure



- Bundled with containerized Ericsson Applications
- Supports deployment onto pre-provisioned servers meeting minimum specified system requirements

Cloud Container Distribution

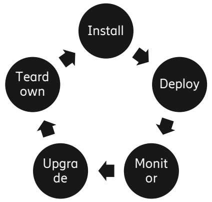
Summary

Value Proposition

Readymade for Telco

"Out-of-the box" Telco-grade container orchestration and management platform ready for production workloads.

Unique full stack cloud portfolio along with Ericsson applications/VNFs portfolio. Ability to provide end-to-end verified solution including container infra and VNFs



Life Cycle Management

Provides tools for installation, upgrade and other LCM operations.

Installation toolkit can be used to configure clusters with the included plug-in components.

Deployment Flexibility

Image Based Deployment - provides flexibility and resource optimization by re-using existing virtualization infrastructure to deploy clusters and orchestrate containerized Network Functions.

Pre-Provisioned Servers Deployment - deploy clusters on any underlying hosts which meets the minimum system requirements. This provides greater degree of flexibility for the customers in choosing the infrastructure for their container solutions.

Forward looking container technology

CCD adds value to Ericsson NFVI solution by providing container orchestration and management capabilities to the existing virtualized solution.

By providing a common container platform that can be leveraged by multiple customer use cases, it provides the means for Ericsson customers to move to the next generation of telco applications.