stl.tech

Getting **BSS-on-Cloud Strategy** Right

A light reading webinar

Know the Speakers

Ajay Iyer

V.P - Platform R&D, Technology Unit Management, STL

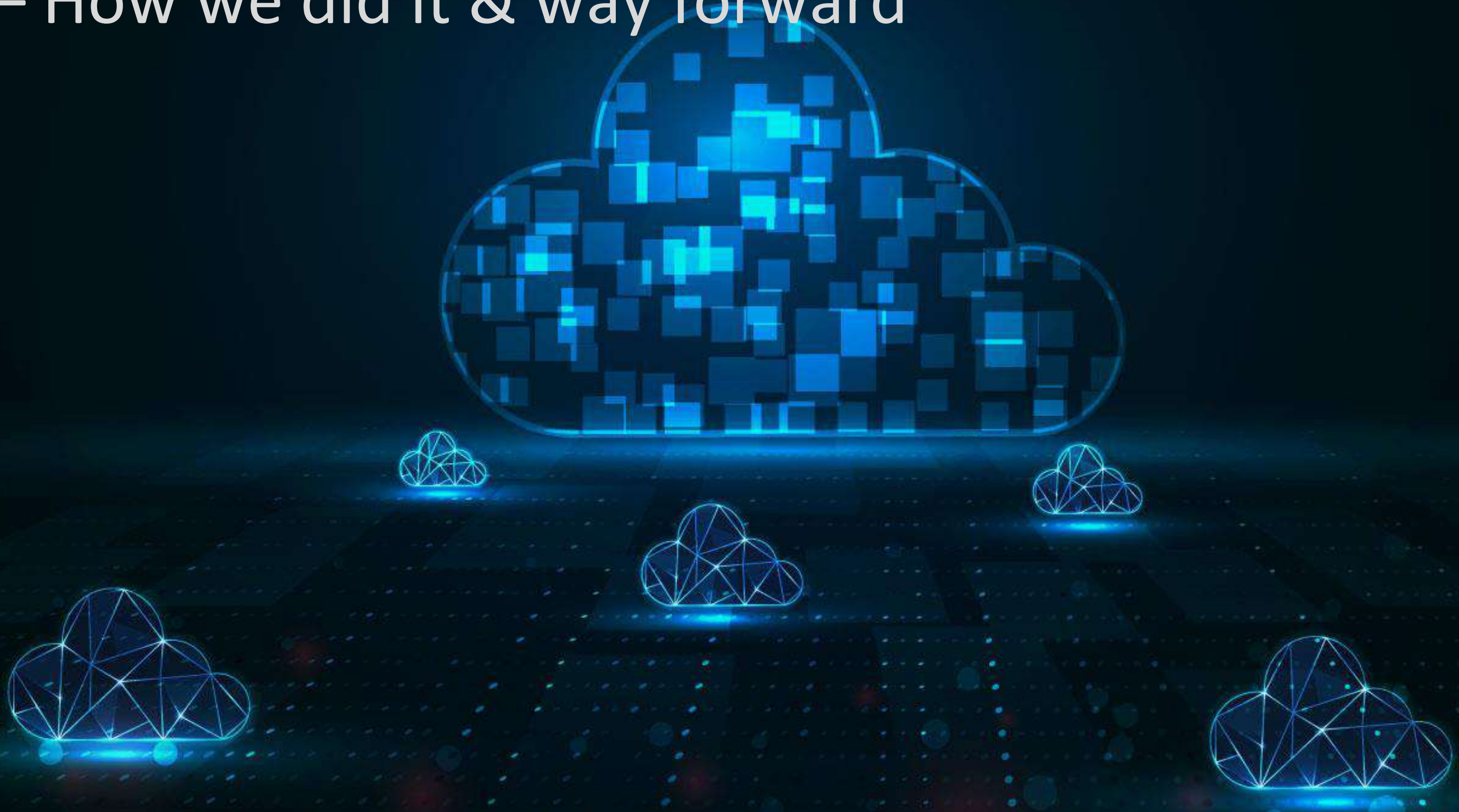
Ajay is passionate about creating innovative products based on open-source technology adoption leading the product development, engineering, delivery, R&D Implementation. Currently, he serves as a VP responsible for overseeing all phases of Product Engineering, Delivery, Platform, Architecture, and Technologies & Modernization of application and ensuring successful rollout of NPD across multiple product lines, selection of open-source technology for enhancing high product growth across the portfolio and rolling out BSS Products across various geographies.

Ravi Shankar is a seasoned professional with customer-first approach & proven track record of product and portfolio management, technology development and digital transformation in enterprise & network software domain. Being a strong advocate of people-first, digital lifestyle and equitable access to resources, Ravi is passionate about using technology to bring extreme agility, data-driven decision making and significant cost savings to the enterprises.

Ravi Shankar

Head - Product Management, STL

1. STL Public Cloud Journey
2. STL Approach for the Right Strategy for BSS on the Cloud
3. STL Story – How we did it & way forward



Company History

1988

Established in Aurangabad

Optical Fibre Cable Plant in Silvassa

1993

1995

Optical Fibre Plant In Aurangabad

Data Cable in Dadra

2004

2012

Manufacturing set up in China

Manufacturing set up in Brazil

2013

2014

Optical Fibre Centre of Excellence

Telecom Software Acquisition

2015

2017

Centre for Smarter Networks

Acquisition of Metallurgica Bresciana s.p.a., Italy

2018

2019

Semiconductor Grade Industry 4.0 Glass Plant

European Data Centre Services IDS Acquisition

Community Memberships for Programmable Networks

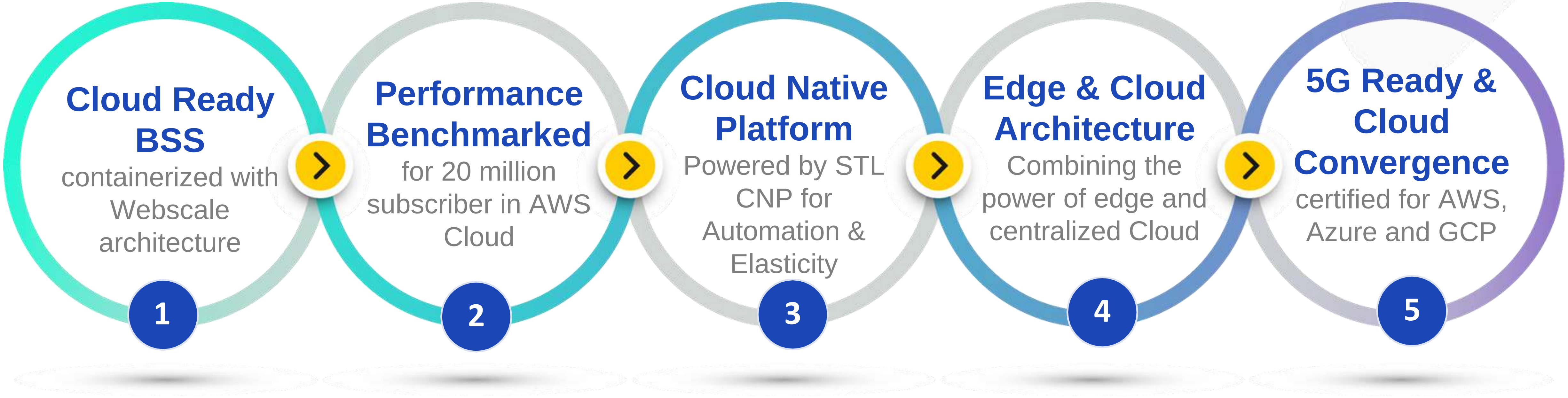
vRAN Investment in ASOCS

5G Virtualization and 5G Radio Partnership

Acquisition of Optical Interconnect Company Optotec S.p.A Italy

2020

STL "Built-for-Cloud" Journey



Cloud Ready BSS containerized with Webscale architecture

1

Early 2018

Performance Benchmarked for 20 million subscriber in AWS Cloud

2

Late 2018

Cloud Native Platform Powered by STL CNP for Automation & Elasticity

3

2019

Edge & Cloud Architecture Combining the power of edge and centralized Cloud

4

2020

5G Ready & Cloud Convergence certified for AWS, Azure and GCP

5

2021

kubernetes docker GitLab Apache maven hazelcast Apache ACTIVEMQ ELK Grafana Istio

aws

CLOUD NATIVE COMPUTING FOUNDATION

Available on ORACLE Cloud Marketplace

Reviewed by

Gartner

BAIN & COMPANY

aws partner network

Reviewed

- Registered in AWS Partner Network on ISV Path
- FTR Completed

Azure

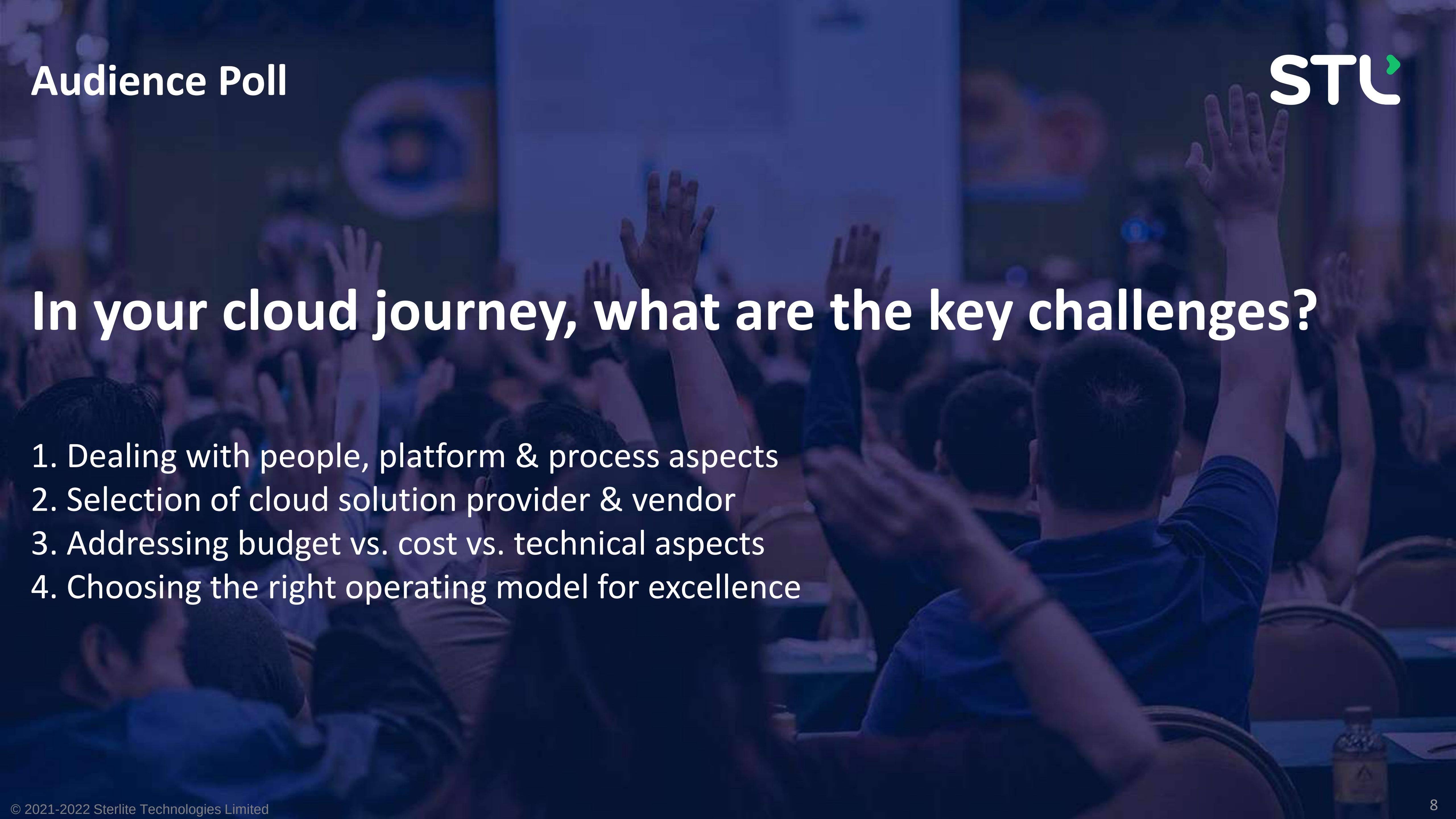
Available on Azure Public Cloud Marketplace

1. STL Public Cloud Journey
2. STL Approach for the Right Strategy for BSS on the Cloud
3. STL Story – How we did it & way forward

Audience Poll

Where are you in cloud journey?

1. Preliminary phase of analysis
2. Advanced stage of analysis/POCs
3. Using Private / Public Cloud Deployment
4. Using Hybrid /Multi Cloud /Multi Edge Deployment (Expert)



Audience Poll

In your cloud journey, what are the key challenges?

1. Dealing with people, platform & process aspects
2. Selection of cloud solution provider & vendor
3. Addressing budget vs. cost vs. technical aspects
4. Choosing the right operating model for excellence

CSPs have to get only 10% wrong to be 100% lost

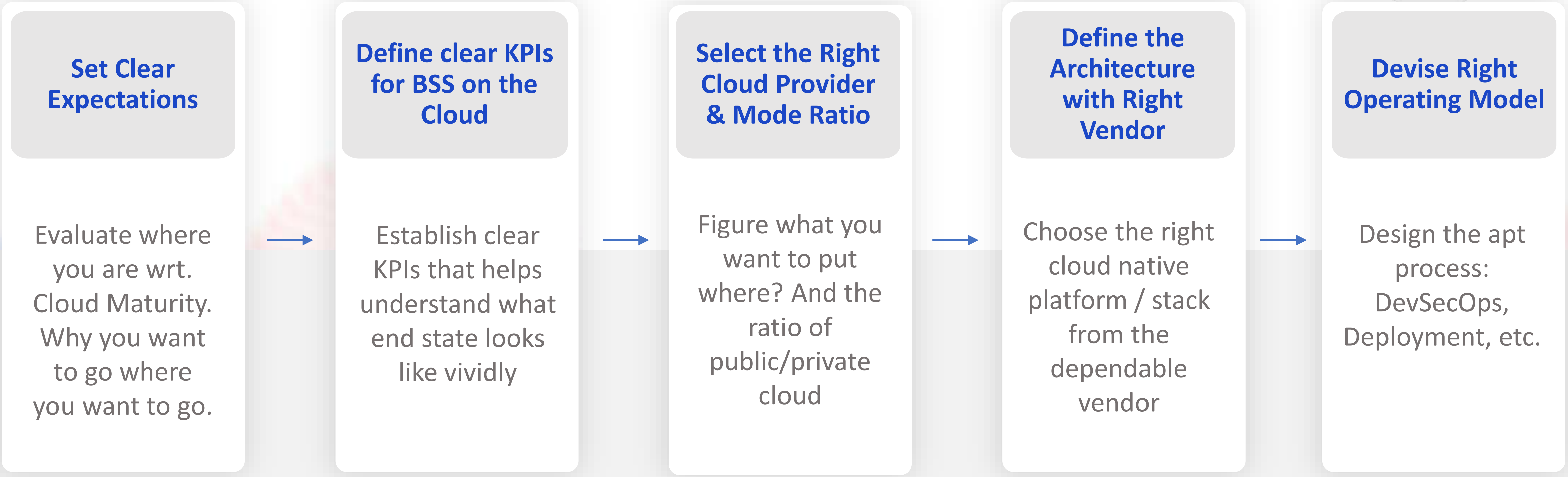
For BSS On the Cloud Journey

Lift-and-shift from legacy to cloud is a perfect recipe for disaster. Why so?

1. **Teams** are not accustomed to Behaviour changes to have expertise on new model
2. **Delivery model** is not designed for cloud infrastructure
3. **Platform's managed services** are not utilized by many applications and hence, development costs and times are always high
4. **Deployment strategy** is not tuned for optimal cloud resource utilization and hence, results into very high TCOs because of over provisioning
5. **Troubleshooting becomes difficult** when standard practices for logging and monitoring for cloud are not implemented

STL Approach for the Right Strategy for BSS on the Cloud

Art of Defining what is "*Right*"



Set Clear Expectations

Evaluate where you are wrt. Cloud Maturity. Why you want to go where you want to go.

Define clear KPIs for BSS on the Cloud

Establish clear KPIs that helps understand what end state looks like vividly

Select the Right Cloud Provider & Mode Ratio

Figure what you want to put where? And the ratio of public/private cloud

Define the Architecture with Right Vendor

Choose the right cloud native platform / stack from the dependable vendor

Devise Right Operating Model

Design the apt process: DevSecOps, Deployment, etc.

Knowing the recipe is not same as cooking it

Most important: Get your basics right and have stage wise evolution of your applications to cloud

Set Clear Expectations for BSS on the Cloud (1/3)

Creating strong foundation
People

Training & Certification & Development Practices

-Setup **Development environment on Kubernetes cluster on** which developers work
- Use of Platform tools for troubleshooting purpose during the Development phase

Cloud optimized
Processes

CI-CD-CA & Zero Touch Deployment

- Create proper **GIT branching strategy**
- **Configuration must be done externally** and does not require any rebuild of Application
- **Implement CI-CD-CA** right from Developer environment to cloud

Cloud Native Transformations of
Applications

12 Factor App & CNCF Certified Components

- Create **independent, lightweight and stateless Micro services**
- **Automatic scaling of application instance** based on CPU, RAM or custom metrics

Use the power of
Platform Services

Optimal Cloud Resource Utilization & Centralized Platform Services

- Platform provided managed services like **Gateway, Service Mesh, Caching, Queue Management, Logging, Monitoring, Identity Management, Storage, Cloud native database management** etc. must be used.

Set Clear Expectations for BSS on the Cloud (2/3)

Security by Design
Security

DevSecOps & Secure APIs

- **Vulnerability scanning of Docker Images** using Harbor, **External Libraries** using OWASP Dependency Track and **Application Source Code** using Sonar
- Store application credentials in a **Secure Vault**

Resiliency by design
Resiliency

Auto-Scaling & Auto-Healing & Chaos Engineering

- Application must provide **health check APIs** to enable auto healing of Application
- Ensure there is Zero Data Loss, Zero Message Loss

Hassle free
Seamless Upgrade

Zero Downtime & Periodic upgrade to N-1 version

- Achieve Zero downtime with the right deployment strategy be it **rolling upgrade**, **blue green deployment** or **canary deployment**
- Periodically upgrade platform components to **the last stable release**

Centralized platform managed
Deep Observability

Centralized Logging & Centralized Monitoring

- Application must do **real time streaming of application logs** to centralized Log Management System like ELK
- Application must push the **application metrics and traces** to the platform monitoring tools like Prometheus and Jaeger

Set Clear Expectations for BSS on the Cloud (3/3)

Micro services interoperability
Service Mesh

Traffic Management & Inter Service Communication

- Implement Service Mesh and use protocols like gRPC for **Inter Micro service communication**
- Visualize the **live service mesh** traffic using Kiali
- Implement routing rules, retries, timeouts, ingress, egress, circuit breaking, rate limiting, canary deployment, shadow testing & dark launches

Any-play platform for
Convergence

Monetization & Policy & Product Agnostic

- **Standard configuration** and productized **template-based approach** for faster time-to-market, **product agnostic capability** with provision for relevant data sets at all touch points.

Close to the Edge
Edge + Cloud

Hybrid & Direct Connect

- Designed to support **hybrid deployments** (public and private cloud) based **on latency and resiliency** requirements. DR and HA at component levels.

Agile, Scalable, Agnostic
Open Architecture

Open Source Technologies & Open API and ODA Aligned

- Use **CNCF certified** Open source tools and technologies
- Have TMF ODA Aligned and Open API based architecture to have **faster integration** with any 3rd party components

Define Clear KPIs for BSS on the Cloud

Development KPIs

1. Build Time < 5 min
2. Zero Sonar blockers / Criticals
3. Code Coverage up to 84%

Security KPIs

Zero vulnerabilities in

- Application source code
- External (3rd party) libraries
- Application image

Upgrade & Release KPIs

1. Zero Downtime
2. Seamless upgrade to N-1 (the last stable version)
3. Automatic rollbacks with backward compatibility
4. Faster release cycles

Development KPIs

1. Image Size < 500 MB for Cloud native application
2. Image Size < 100 MB for CNF
3. Application time <1 min
4. Zero touch deployment

Monitoring & Troubleshooting KPIs

1. Real time system and application metrics availability on central monitoring tool
2. Alerts to be generated upon breach of any threshold (CPU utilization, Auto heal, Auto scale etc.)

Open Source KPIs

1. Zero vendor lock-in
2. 100% CNCF certified components

Platform KPIs

1. Auto Healing
2. Auto Scaling
3. No image rebuild with configuration change
4. Zero data loss with pod crash

Set Clear Expectations | Define Clear KPIs for BSS on the Cloud | Select the Right Cloud Provider & Mode Ratio | Define the Architecture with Right Vendor | Devise Right Operating Model

Parameters to evaluate private/ public / hybrid /edge cloud

Private Clouds are better suited for applications requiring

- Critical real-time sessions (Packet Gateway)
- Low-latency transactions (Online charging)

Hybrid Clouds are better suited for applications

- That can run stable on private cloud and need scaling only on special occasions for e.g. promotions, back up, DR

Public Clouds are better suited for Applications requiring

- Opex based investment
- Facing high demand fluctuation
- Redundancy for Critical Services (DRM)
- Optimized Storage Footprint for higher economic benefits
- Trials / Pre-Production, UAT environments

Edge Clouds are better suited for applications

- That require ultra low latency connectivity needs proximity to customer is super critical

Set Clear Expectations

Define Clear KPIs for BSS on the Cloud

Select the Right Cloud Provider & Mode Ratio

Define the Architecture with Right Vendor

Devise Right Operating Model

Parameters to evaluate which cloud provider to choose

Local presence of cloud provider in respective geography

Investment / acquisition to do multi cloud management

Public Cloud Location Selection based on

- Application Latency requirements
- Data Privacy Requirements
- Multi-Tenancy Requirements

Platform maturity to support your use case

- Edge based solution
- Bundle support
- Free Tier support, Pricing flexibility to support your needs

Partner Maturity

- Partner programs & support for sale
- Partner Support SLAs
- Partner Joint Marketing

Set Clear Expectations

Define Clear KPIs for BSS on the Cloud

Select the Right Cloud Provider & Mode Ratio

Define the Architecture with Right Vendor

Devise Right Operating Model

Select the Right Cloud Provider & Mode Ratio

STL

aws

Digital BSS on AWS

Large Addressable Market

Google Cloud Platform

Digital BSS on GCP

Aggressive Pricing and Opensource Ecosystem Advantage

Microsoft Azure

Digital BSS on Azure

Telco Focus & Partnership Advantage

CRM as a Service, Order Management as a Service, Self Service / Engagement Platform as a Service, Billing as a Service, Offline Charging, Provisioning, Policy Control, Network Monitoring System, Analytics as a Service, Notification Management as a Service

Private Cloud / On premise

Large Addressable Market

kubernetes

openstack.

vmware®

Network Functions, Large data lake, Online charging , applications with steady traffic/high data volume for e.g. IPTV

Set Clear Expectations | Define Clear KPIs for BSS on the Cloud | **Select the Right Cloud Provider & Mode Ratio** | Define the Architecture with Right Vendor | Devise Right Operating Model

Define the Architecture with Right Vendor

Based on following Architecture Principles

Enabling Consistent User experience across

- Configuration
- Provisioning
- Operations
- Administration &
- Management

Business Agility - minimization of TTM and risk of launching new product / service

Engage Right Design - exploiting flexible features by the cloud such as Elastic Scaling

Security by Design - Per-Component, Per-Service and System Wide

Resiliency by Design - System Auto-Heal, Auto-Scale, peak /off-peak loads

Privacy by Design - Data protection across design, implementation & ongoing Ops

Operations Cost Minimization -through Policy based Deployment Stack enabling

Agile Governance - Centralized Mgmt. of Products/Processes /Teams across multi-markets

Set Clear Expectations

Define Clear KPIs for BSS on the Cloud

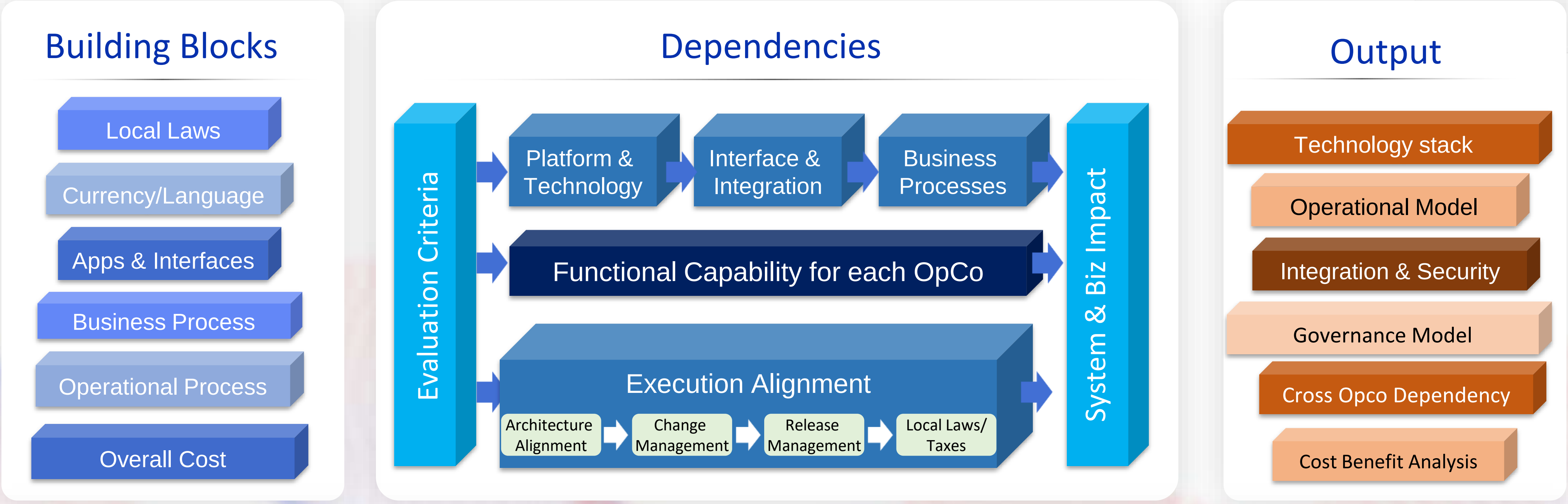
Select the Right Cloud Provider & Mode Ratio

Define the Architecture with Right Vendor

Devise Right Operating Model

Define the Architecture with Right Vendor

To Run Multi-Geography Operations



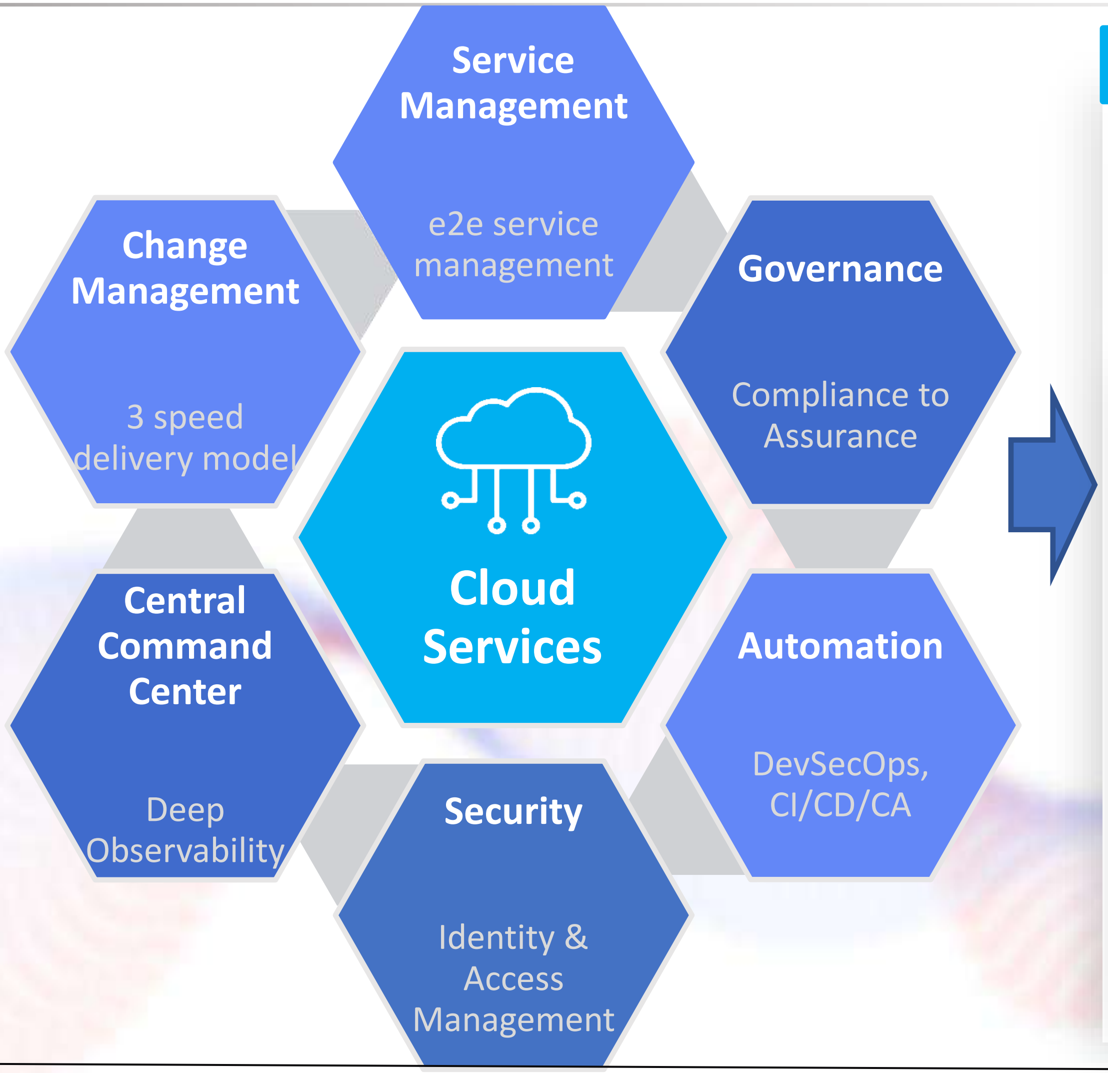
Create Common Business team level Integration for Planning, Priority, Change Management and Execution

- Focus on Business Process dependency and Long-term operational efficiency (both Cost & TTM)
- Harmonization of Platform should bring efficiency and not a business bottleneck
- Have right balance between Local vs Global

Set Clear Expectations | Define Clear KPIs for BSS on the Cloud | Select the Right Cloud Provider & Mode Ratio | **Define the Architecture with Right Vendor** | Devise Right Operating Model

Device Right Operating Model

Cloud Managed Services & Operations



Hybrid of Consultative and Disruptive Modes

Best Practices that suits your customer needs

Tailor-made approach for efficient operations

Strategic Consulting

Ongoing Technology Adoption

Address Innovations for operational efficiency

Preventive & Cooperative Approach

Drive communication, collaboration & Do preventive maintenance

Special attention to critical sites

Set Clear Expectations

Define Clear KPIs for BSS on the Cloud

Select the Right Cloud Provider & Mode Ratio

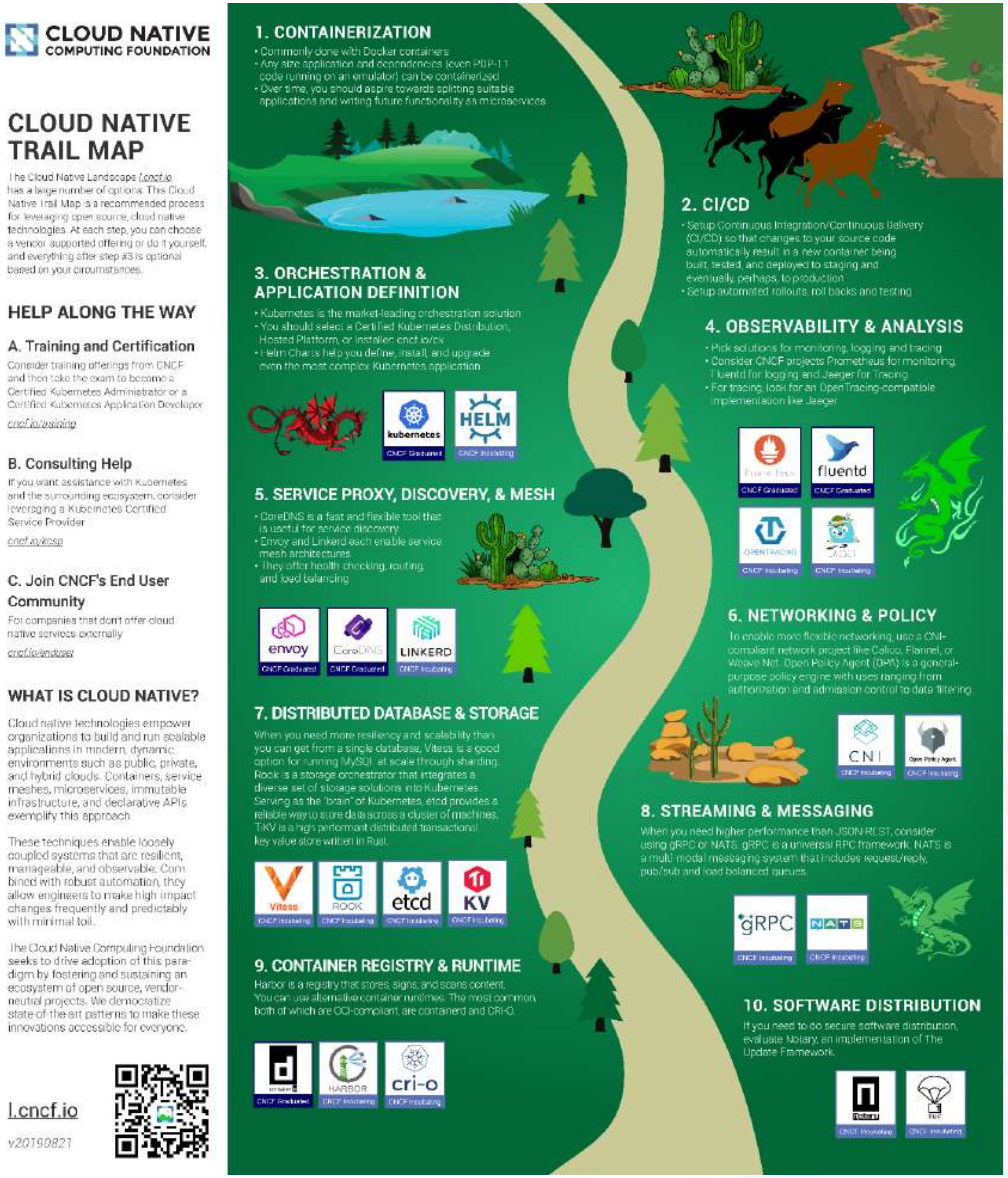
Define the Architecture with Right Vendor

Devise Right Operating Model

1. STL Public Cloud Journey
2. STL Approach for the Right Strategy for BSS on the Cloud
3. STL Story – How we did it & way forward

CNCF Trail Map Adoption

Sr. No.	CNCF Trail map Milestone	Status
1	Containerization	Completed
2	CI/CD	Completed
3	Orchestration & Application Definition - Kubernetes - Helm	Completed
4	Observability & Analysis – ELK - Prometheus & Grafana - Jeager & Kiali	Completed
5	Service Proxy, Discovery & Mesh	Completed
6	Networking & Policy (Advance Use Cases)	Completed Planned
7	Distributed Database & Storage	Completed
8	Streaming & Messaging	Completed
9	Container Registry & Runtime	Completed
10	Software Distribution	WIP



CLOUD NATIVE COMPUTING FOUNDATION

CLOUD NATIVE TRAIL MAP

The Cloud Native Landscape l.cncf.io has a large number of options. This Cloud Native Trail Map is a recommended process for leveraging open source, cloud native technologies. At each step, you can choose a vendor supported offering or do it yourself, and everything after step #3 is optional based on your circumstances.

HELP ALONG THE WAY

A. Training and Certification

Consider training offerings from CNCF and then take the exam to become a Certified Kubernetes Administrator or a Certified Kubernetes Application Developer

cncf.io/training

B. Consulting Help

If you want assistance with Kubernetes and the surrounding ecosystem, consider leveraging a Kubernetes Certified Service Provider

cncf.io/kcsp

C. Join CNCF's End User Community

For companies that don't offer cloud native services externally

cncf.io/enduser

WHAT IS CLOUD NATIVE?

Cloud native technologies empower organizations to build and run scalable applications in modern, dynamic environments such as public, private, and hybrid clouds. Containers, service meshes, microservices, immutable infrastructure, and declarative APIs exemplify this approach.

These techniques enable loosely coupled systems that are resilient, manageable, and observable. Combined with robust automation, they allow engineers to make high impact changes frequently and predictably with minimal toil.

The Cloud Native Computing Foundation seeks to drive adoption of this paradigm by fostering and sustaining an ecosystem of open source, vendor-neutral projects. We democratize state-of-the-art patterns to make these innovations accessible for everyone.

l.cncf.io

v20190821

1. CONTAINERIZATION

- Commonly done with Docker containers
- Any size application and dependencies (even PDP-11 code running on an emulator) can be containerized
- Over time, you should aspire towards splitting suitable applications and writing future functionality as microservices

2. CI/CD

- Setup Continuous Integration/Continuous Delivery (CI/CD) so that changes to your source code automatically result in a new container being built, tested, and deployed to staging and eventually, perhaps, to production
- Setup automated rollouts, roll backs and testing

3. ORCHESTRATION & APPLICATION DEFINITION

- Kubernetes is the market-leading orchestration solution
- You should select a Certified Kubernetes Distribution, Hosted Platform, or Installer: cncf.io/ck
- Helm Charts help you define, install, and upgrade even the most complex Kubernetes application

4. OBSERVABILITY & ANALYSIS

- Pick solutions for monitoring, logging and tracing
- Consider CNCF projects Prometheus for monitoring, Fluentd for logging and Jaeger for Tracing
- For tracing, look for an OpenTracing-compatible implementation like Jaeger

5. SERVICE PROXY, DISCOVERY, & MESH

- CoreDNS is a fast and flexible tool that is useful for service discovery
- Envoy and Linkerd each enable service mesh architectures
- They offer health checking, routing, and load balancing

6. NETWORKING & POLICY

To enable more flexible networking, use a CNI-compliant network project like Calico, Flannel, or Weave Net. Open Policy Agent (OPA) is a general-purpose policy engine with uses ranging from authorization and admission control to data filtering

7. DISTRIBUTED DATABASE & STORAGE

When you need more resiliency and scalability than you can get from a single database, Vitess is a good option for running MySQL at scale through sharding. Rook is a storage orchestrator that integrates a diverse set of storage solutions into Kubernetes. Serving as the "brain" of Kubernetes, etcd provides a reliable way to store data across a cluster of machines. TiKV is a high performant distributed transactional key value store written in Rust.

8. STREAMING & MESSAGING

When you need higher performance than JSON-REST, consider using gRPC or NATS. gRPC is a universal RPC framework. NATS is a multi-modal messaging system that includes request/reply, pub/sub and load balanced queues.

9. CONTAINER REGISTRY & RUNTIME

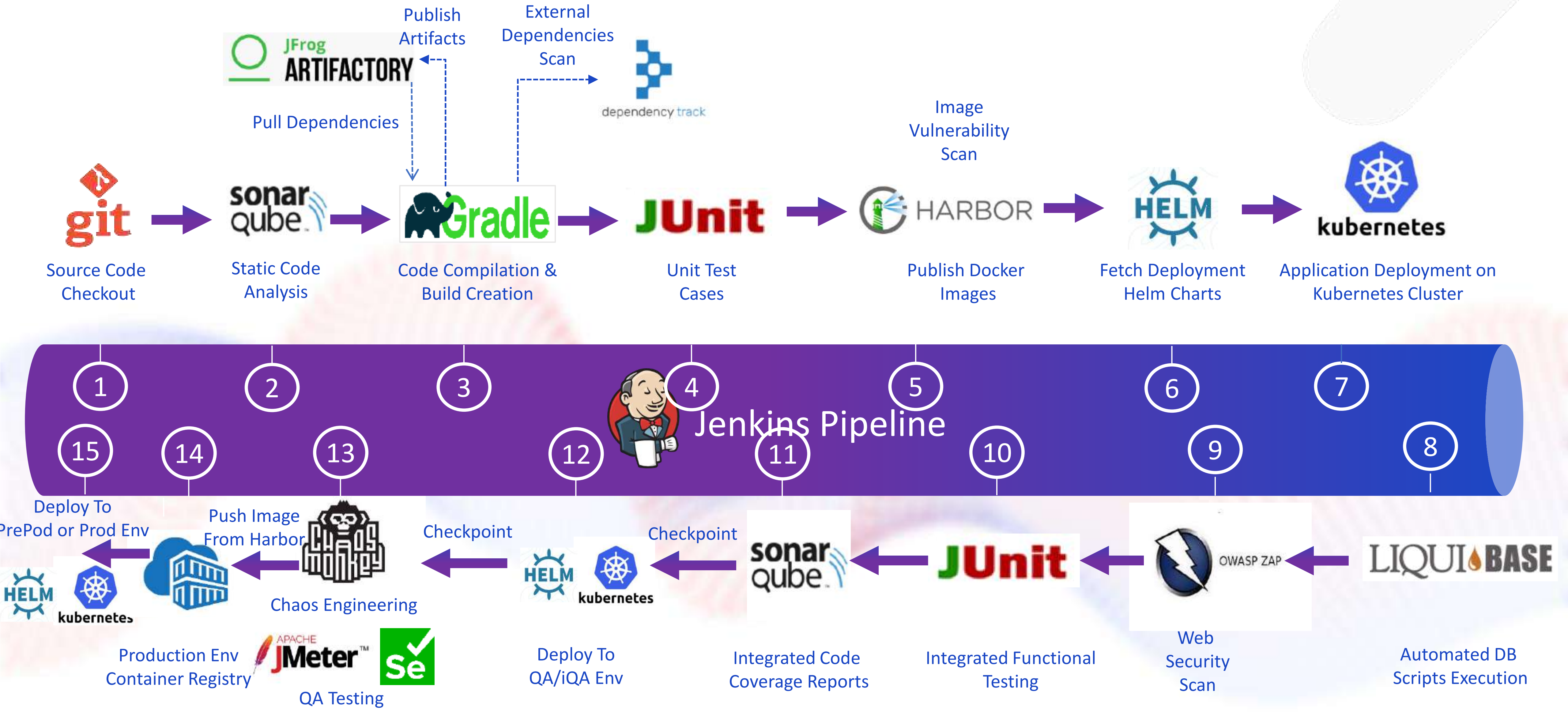
Harbor is a registry that stores, signs, and scans content. You can use alternative container runtimes. The most common, both of which are OCI-compliant, are containerd and CRI-O.

10. SOFTWARE DISTRIBUTION

If you need to do secure software distribution, evaluate Notary, an implementation of The Update Framework.

Source: https://github.com/cncf/trailmap/blob/master/CNCF_TrailMap_latest.png

CI/CD/CA Automation for Cloud native BSS



Pull Dependencies

Publish Artifacts

External Dependencies Scan

Image Vulnerability Scan

Jenkins Pipeline

1 Source Code Checkout

2 Static Code Analysis

3 Code Compilation & Build Creation

4 Unit Test Cases

5 Publish Docker Images

6 Fetch Deployment Helm Charts

7 Application Deployment on Kubernetes Cluster

8 Automated DB Scripts Execution

9 Web Security Scan

10 Integrated Functional Testing

11 Integrated Code Coverage Reports

Checkpoint

12 Deploy To QA/iQA Env

Checkpoint

13 QA Testing

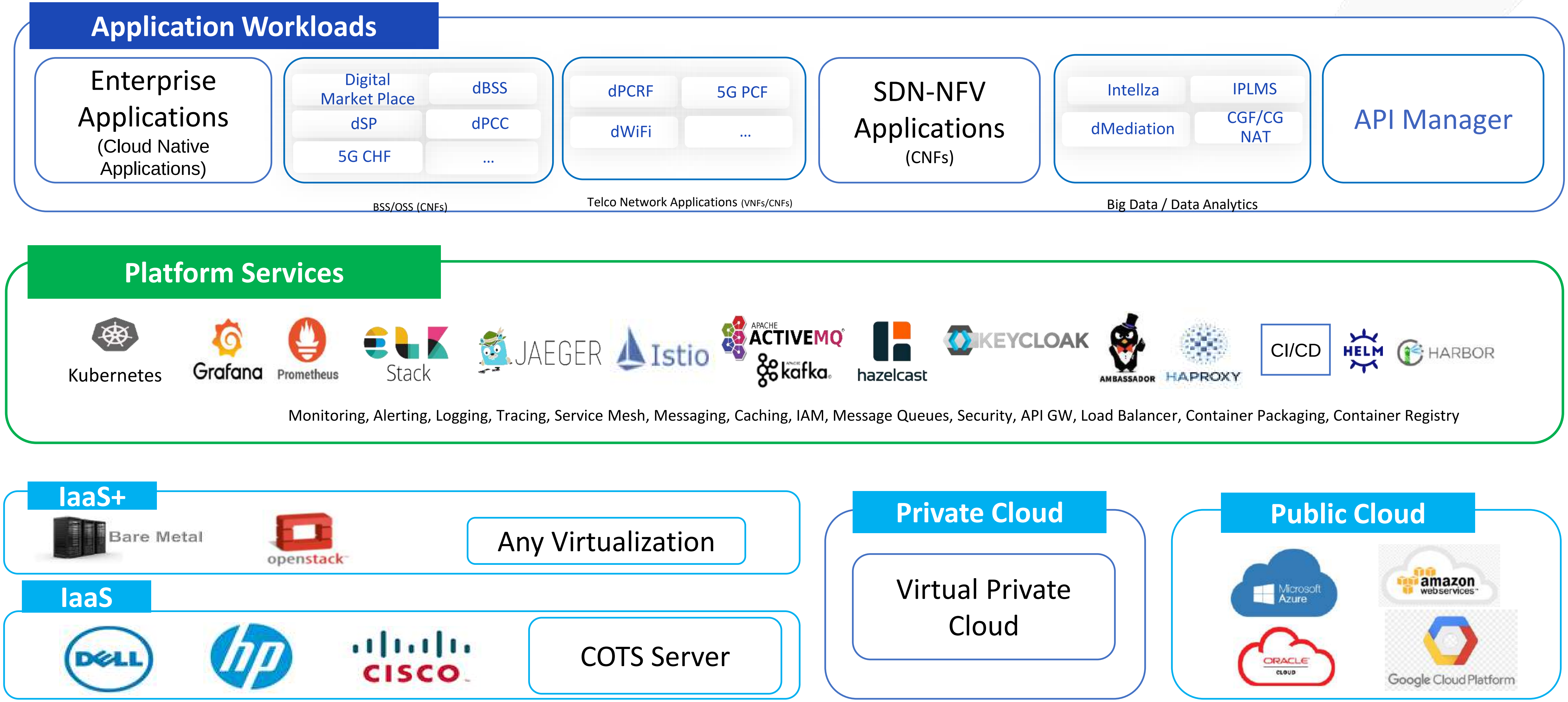
Chaos Engineering

14 Production Env Container Registry

Push Image From Harbor

15 Deploy To PrePod or Prod Env

STL Cloud Native Platform for Cloud native BSS



Application Workloads

Enterprise Applications (Cloud Native Applications)

Digital Market Place | dBSS | dSP | dPCC | 5G CHF | …

BSS/OSS (CNFs)

dPCRF | 5G PCF | dWiFi | …

Telco Network Applications (VNFs/CNFs)

SDN-NFV Applications (CNFs)

Intellza | IPLMS | dMediation | CGF/CG NAT

Big Data / Data Analytics

API Manager

Platform Services

Monitoring, Alerting, Logging, Tracing, Service Mesh, Messaging, Caching, IAM, Message Queues, Security, API GW, Load Balancer, Container Packaging, Container Registry

IaaS+

Any Virtualization

IaaS

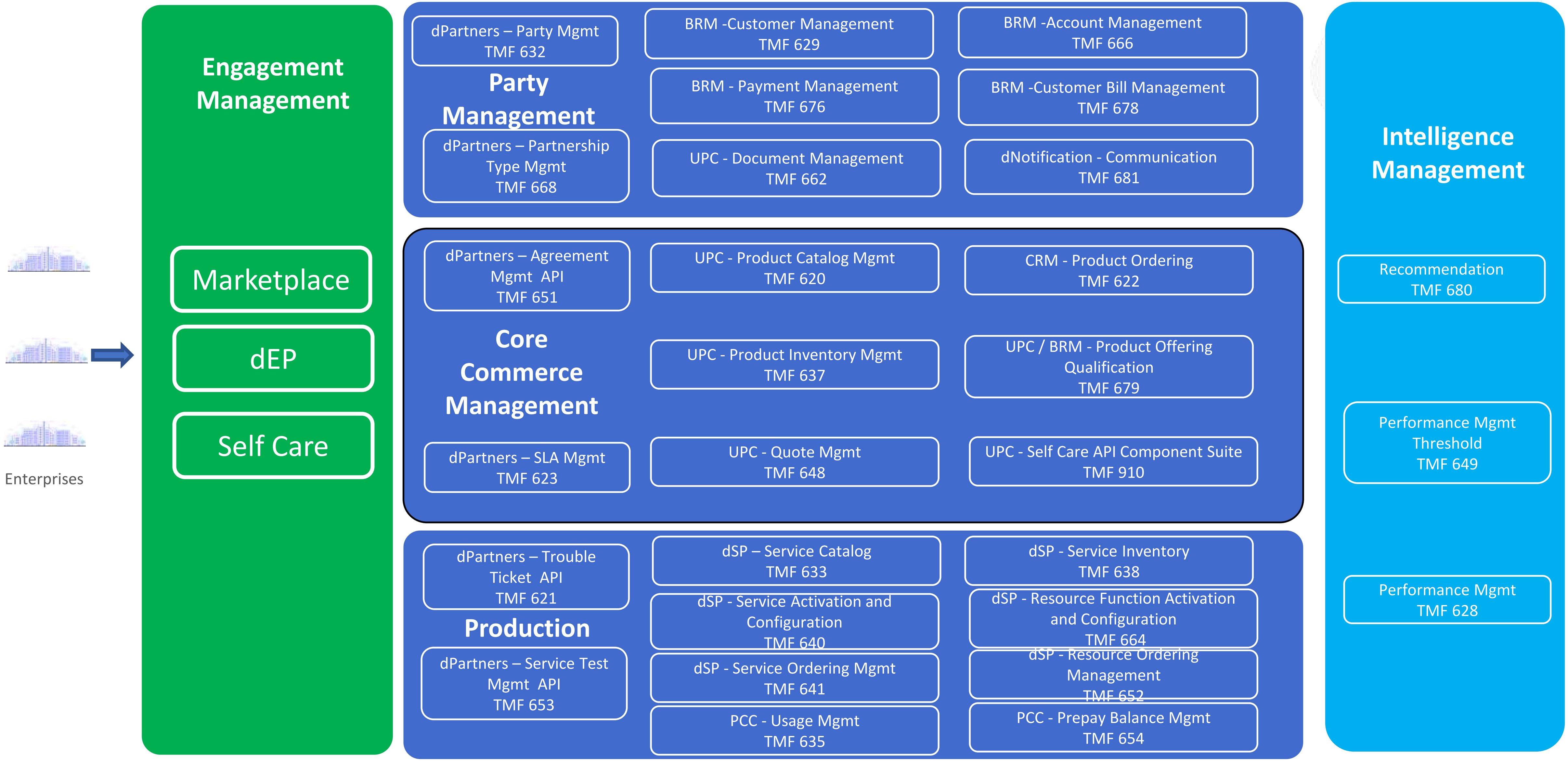
COTS Server

Private Cloud

Virtual Private Cloud

Public Cloud

Open Digital Architecture with Open APIs



Enterprises

Engagement Management

Marketplace

dEP

Self Care

Party Management

dPartners – Party Mgmt TMF 632

dPartners – Partnership Type Mgmt TMF 668

BRM -Customer Management TMF 629

BRM - Payment Management TMF 676

UPC - Document Management TMF 662

BRM -Account Management TMF 666

BRM -Customer Bill Management TMF 678

dNotification - Communication TMF 681

Core Commerce Management

dPartners – Agreement Mgmt API TMF 651

dPartners – SLA Mgmt TMF 623

UPC - Product Catalog Mgmt TMF 620

UPC - Product Inventory Mgmt TMF 637

UPC - Quote Mgmt TMF 648

CRM - Product Ordering TMF 622

UPC / BRM - Product Offering Qualification TMF 679

UPC - Self Care API Component Suite TMF 910

Production

dPartners – Trouble Ticket API TMF 621

dPartners – Service Test Mgmt API TMF 653

dSP – Service Catalog TMF 633

dSP - Service Activation and Configuration TMF 640

dSP - Service Ordering Mgmt TMF 641

PCC - Usage Mgmt TMF 635

dSP - Service Inventory TMF 638

dSP - Resource Function Activation and Configuration TMF 664

dSP - Resource Ordering Management TMF 652

PCC - Prepay Balance Mgmt TMF 654

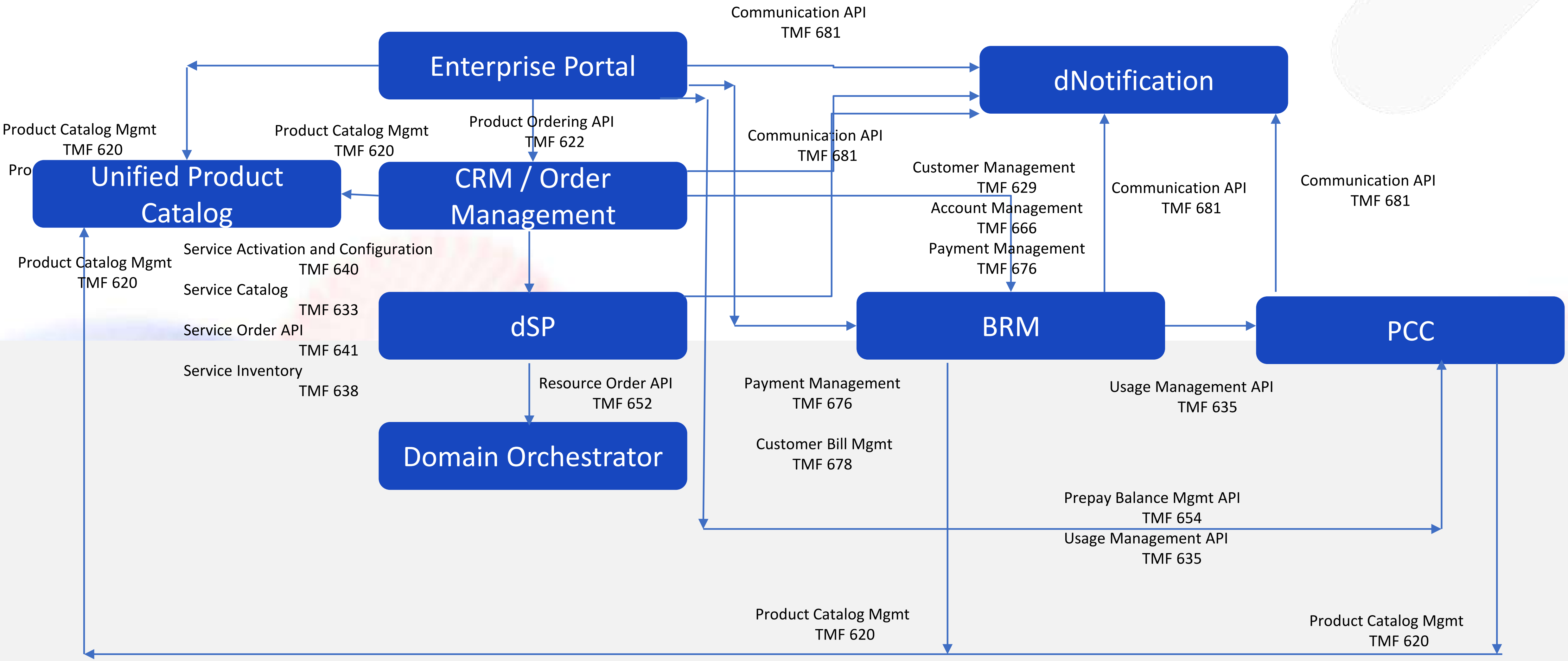
Intelligence Management

Recommendation TMF 680

Performance Mgmt Threshold TMF 649

Performance Mgmt TMF 628

Open API communication across BSS components



Way Forward

- Cloud native storage integrated with application using ROBIN.io, ROOK
- Cloud native networking integrated with application using FD.io
- Cloud native run time security integrated with application using FALCON
- Implementing Federation of Clusters using KubeFed
- Deploying Application on Edge using Kube Edge
- Deploying Application Automation using GitOps with Server less Architecture
- Cloud native AI ML - Deploying machine learning workflows on the cloud using Kube Flow
- Multi Data Centre Automation – using ROBIN.io
- Automated Cloud Infrastructure Management – using ROBIN.io

STL

beyond tomorrow