CABLE-TEC EXPO® 2017

SCTE · ISBE

THE NEXT BIG...

DEAL
CONNECTION
INNOVATION
TECHNOLOGY
LEADER
NETWORK

SCTE·ISBE CABLE-TEC EXPO 2017®

DENVER, CO
OCTOBER 17-20

2017 Fall Technical Forum
SCTE·ISBE • NCTA • CABLELABS

VIRTUALIZATION – STACKS AND SCHEDULERS

SCTE · ISBE

Bridging the Gap Between ETSI-NFV and Cloud Native Architecture

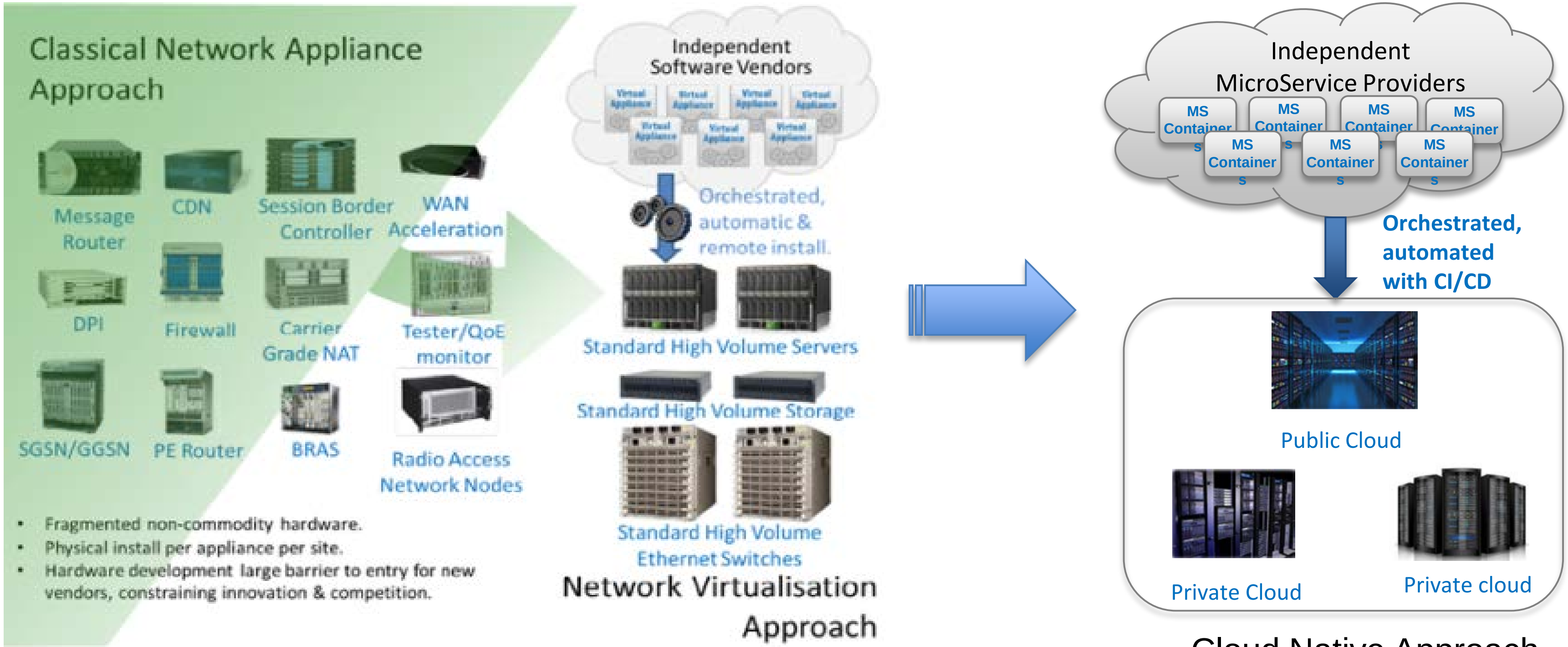
YuLing Chen
Senior Technical Leader
Cisco Systems Inc.

Alon Bernstein
Distinguished Engineer
Cisco Systems Inc.

SCTE · ISBE CABLE-TEC EXPO 2017

DENVER, CO
OCTOBER 17-20

Network Functions moving to the Cloud Native Environment



Classical Network Appliance Approach

Message Router
CDN
Session Border Controller
WAN Acceleration
DPI
Firewall
Carrier Grade NAT
Tester/QoE monitor
SGSN/GGSN
PE Router
BRAS
Radio Access Network Nodes
Independent Software Vendors
Virtual Appliance
Orchestrated, automatic & remote install.
Standard High Volume Servers
Standard High Volume Storage
Standard High Volume Ethernet Switches
Independent MicroService Providers
MS Containers
Orchestrated, automated with CI/CD
Public Cloud
Private Cloud
Private cloud

- Fragmented non-commodity hardware.
- Physical install per appliance per site.
- Hardware development large barrier to entry for new vendors, constraining innovation & competition.

Network Virtualisation Approach

Cloud Native Approach

Cloud Native and ETSI NFV

- Current industry reference for NFV is ETSI-NFV, focusing on VM based solutions
- Cloud native was originated from web scale providers in eCommerce and distribution
- There is a gap that needs to be bridged between ETSI-NFV and Cloud Native solutions

ETSI MANO architecture framework

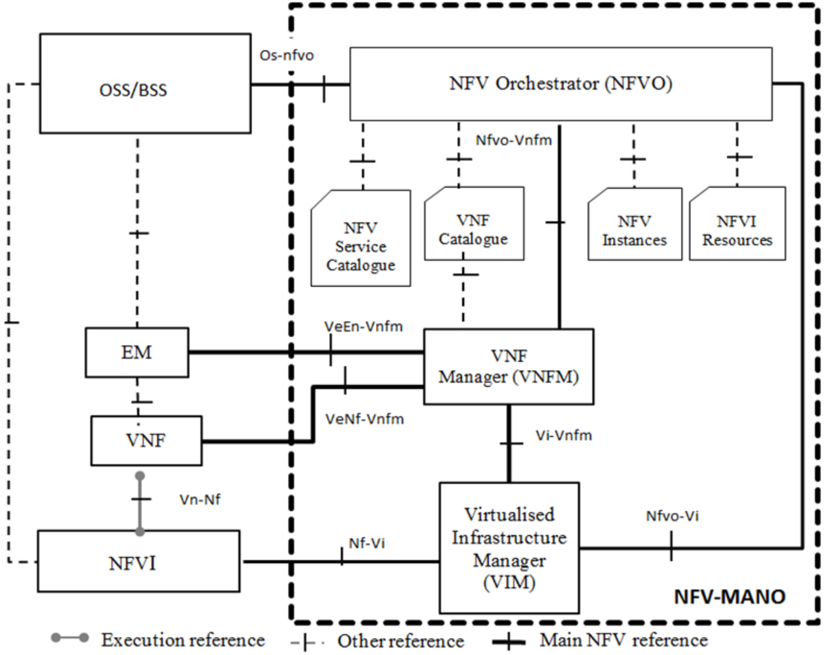
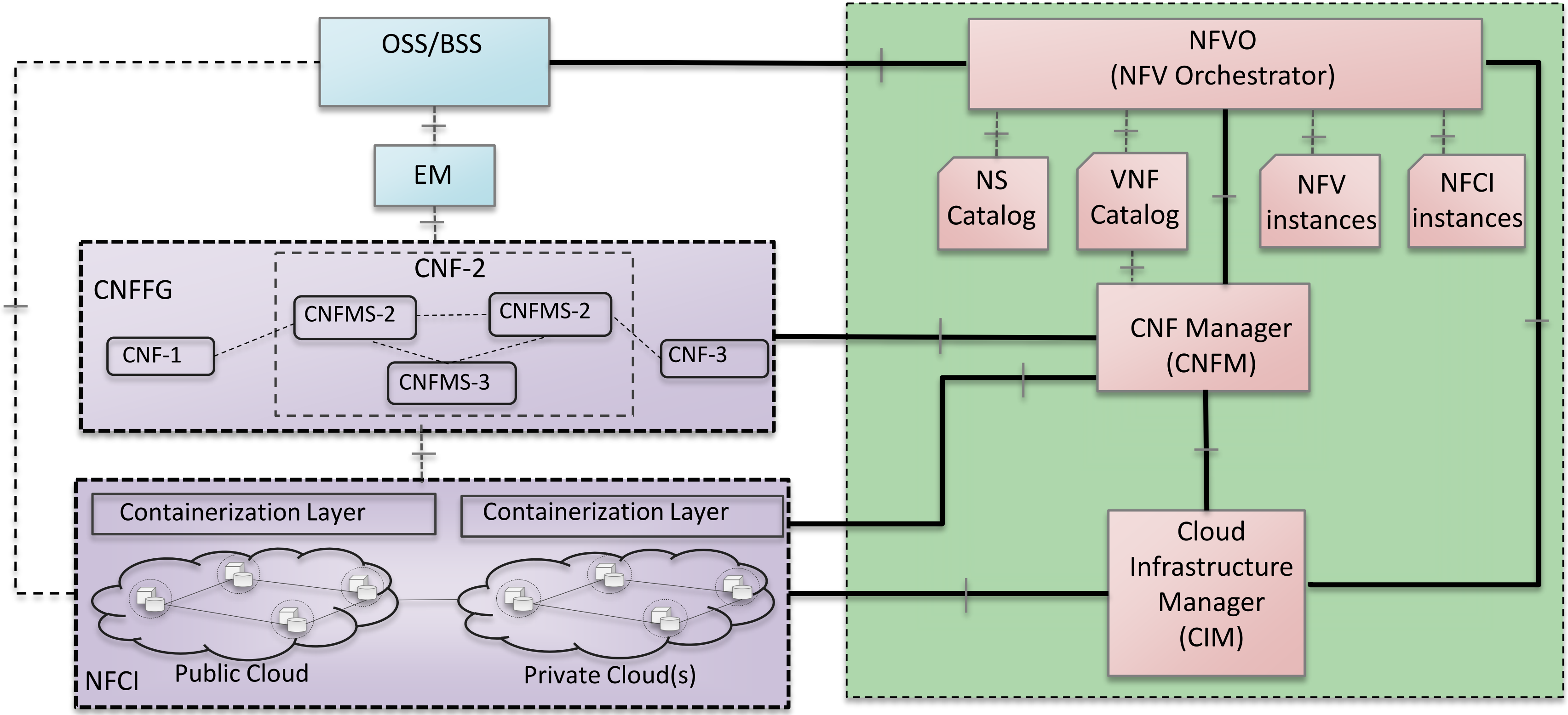


Fig. 1. ETSI NFV-MANO architectural framework with reference points [3]

- NFV Orchestrator (NFVO)
  - ➢ Network Service orchestration
  - ➢ Security validation and authorization
  - ➢ Global Resource Management
  - ➢ Policy Management
- VNF Manager (VNFM)
  - ➢ VNF Lifecycle management
  - ➢ Adaptation, configuration, and coordination for event reporting among NFVI and EMS
- Virtualized Infrastructure Manager (VIM)
  - ➢ NFVI resource management
  - ➢ Performance and event collection and forwarding

Proposed Augmentation to ETSI MANO Reference Architecture in Cloud Native NFV

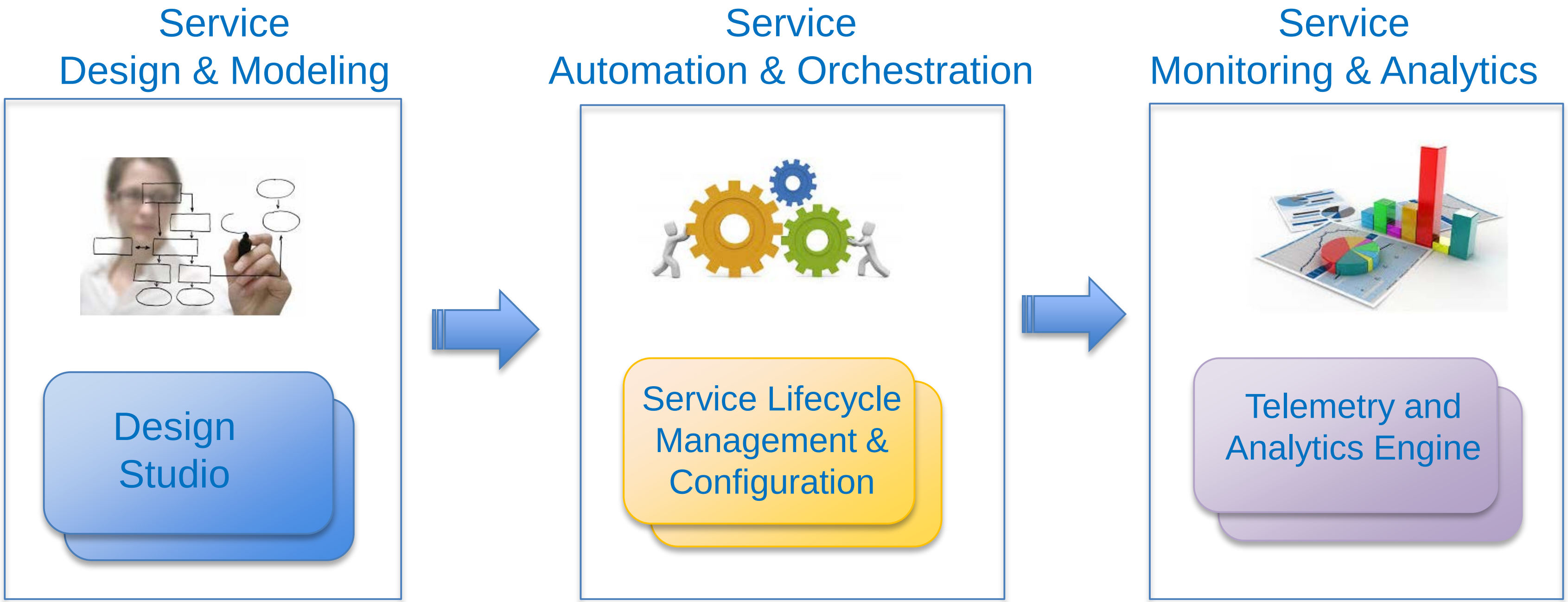


|

Proposed Augmentation to ETSI MANO Terminology

- CNFMS (Cloud Network Function MicroService)
  - ➢ The microservice that belongs to the CNF (Cloud Network Function)
- CNF (Cloud Network Function)
  - ➢ The network functions deployed in the cloud as a form of microservices containers
- NFCI (Network Function Cloud Infrastructure)
  - ➢ Provides the underline physical infrastructure for the network functions
- CIM (Cloud Infrastructure Manager)
  - ➢ Control and manage NFCI with the capability to schedule containers in the cloud
- CNFM (CNF Manager)
  - ➢ Control the lifecycle of the CNFs
- CNFFG (CNF Forwarding Graph)
  - ➢ The list of the CNFs and the virtual links among the CNFs and physical endpoints

A pragmatic NFV MANO system



Service
Design & Modeling

Design Studio

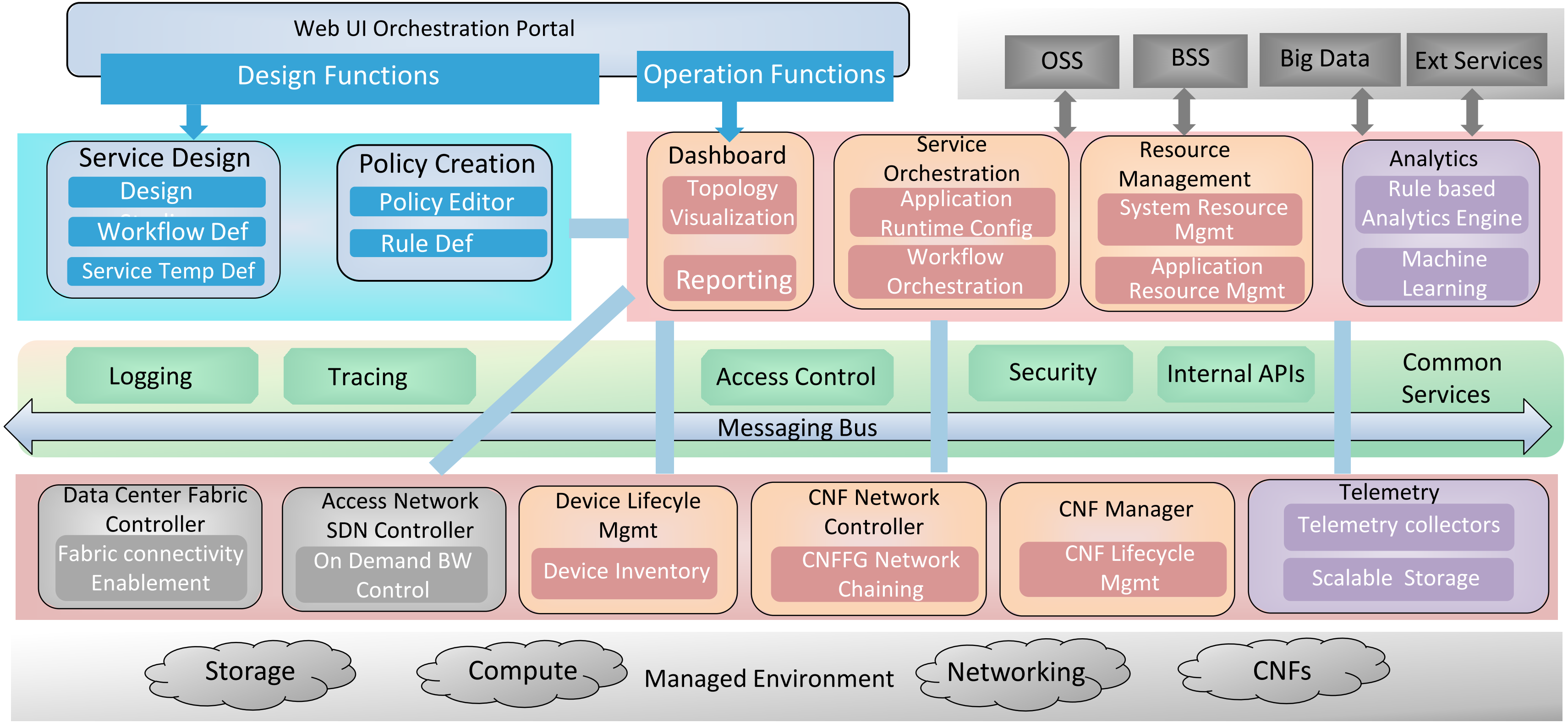
Service
Automation & Orchestration

Service Lifecycle Management & Configuration

Service
Monitoring & Analytics

Telemetry and Analytics Engine

A pragmatic NFV Software Architecture in the Cloud Native Environment



Web UI Orchestration Portal

Design Functions

Operation Functions

OSS

BSS

Big Data

Ext Services

Service Design
- Design
- Workflow Def
- Service Temp Def

Policy Creation
- Policy Editor
- Rule Def

Dashboard
- Topology Visualization
- Reporting

Service Orchestration
- Application Runtime Config
- Workflow Orchestration

Resource Management
- System Resource Mgmt
- Application Resource Mgmt

Analytics
- Rule based Analytics Engine
- Machine Learning

Logging

Tracing

Access Control

Security

Internal APIs

Common Services

Messaging Bus

Data Center Fabric Controller
- Fabric connectivity Enablement

Access Network SDN Controller
- On Demand BW Control

Device Lifecycle Mgmt
- Device Inventory

CNF Network Controller
- CNFFG Network Chaining

CNF Manager
- CNF Lifecycle Mgmt

Telemetry
- Telemetry collectors
- Scalable Storage

Storage

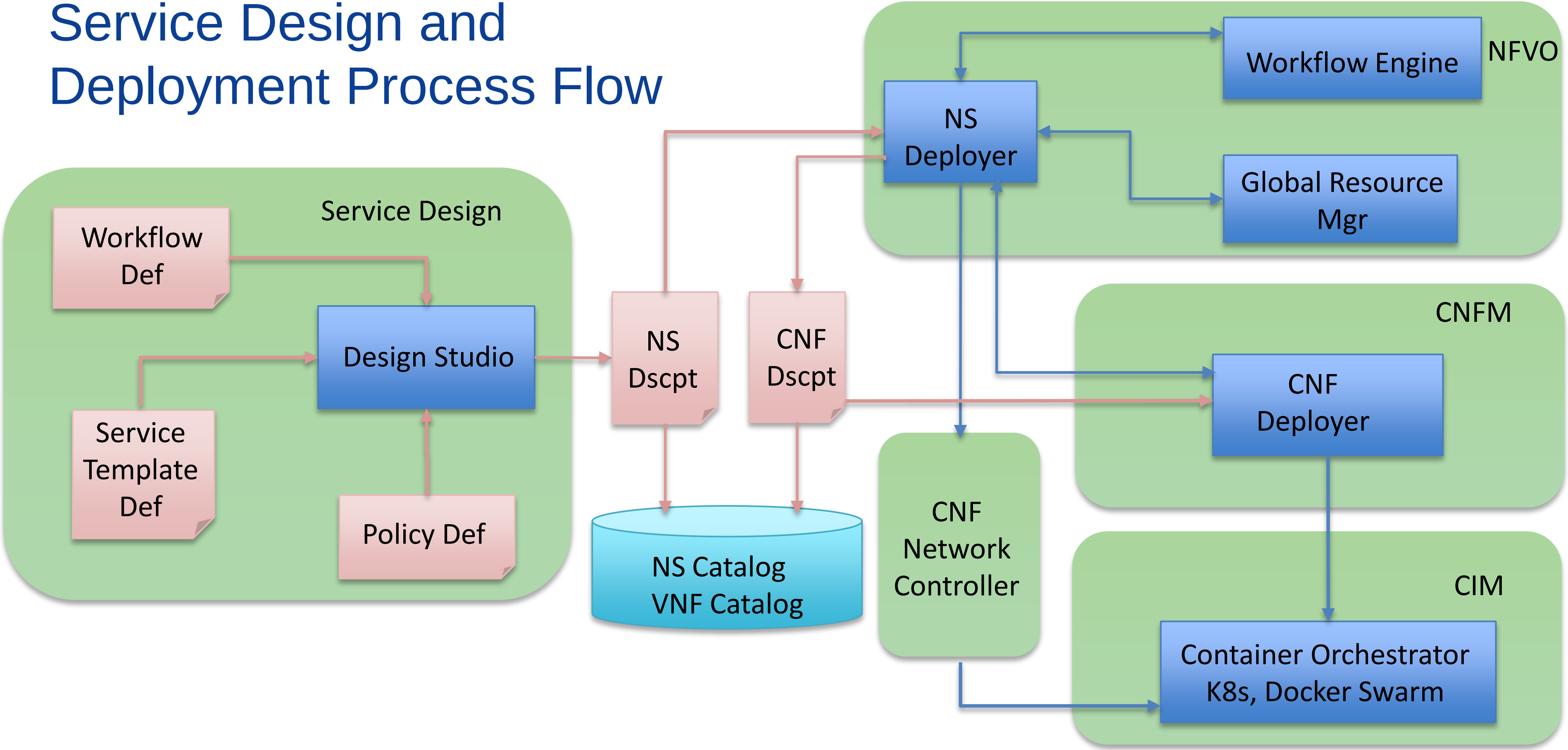
Compute

Managed Environment

Networking

CNFs

Service Design and Deployment Process Flow



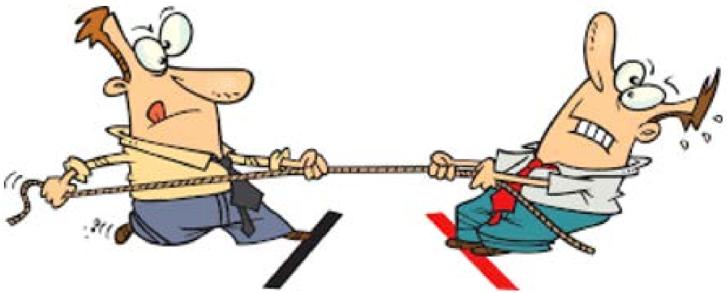
Topology and Orchestration Specification for Cloud Applications (TOSCA)

- Common Data Modeling Language
- Facilitates high levels of service portability
- Managed by industry group OASIS
- At the center of many open NFV orchestration projects
  - Cloudify
  - Tacker
  - Open-O

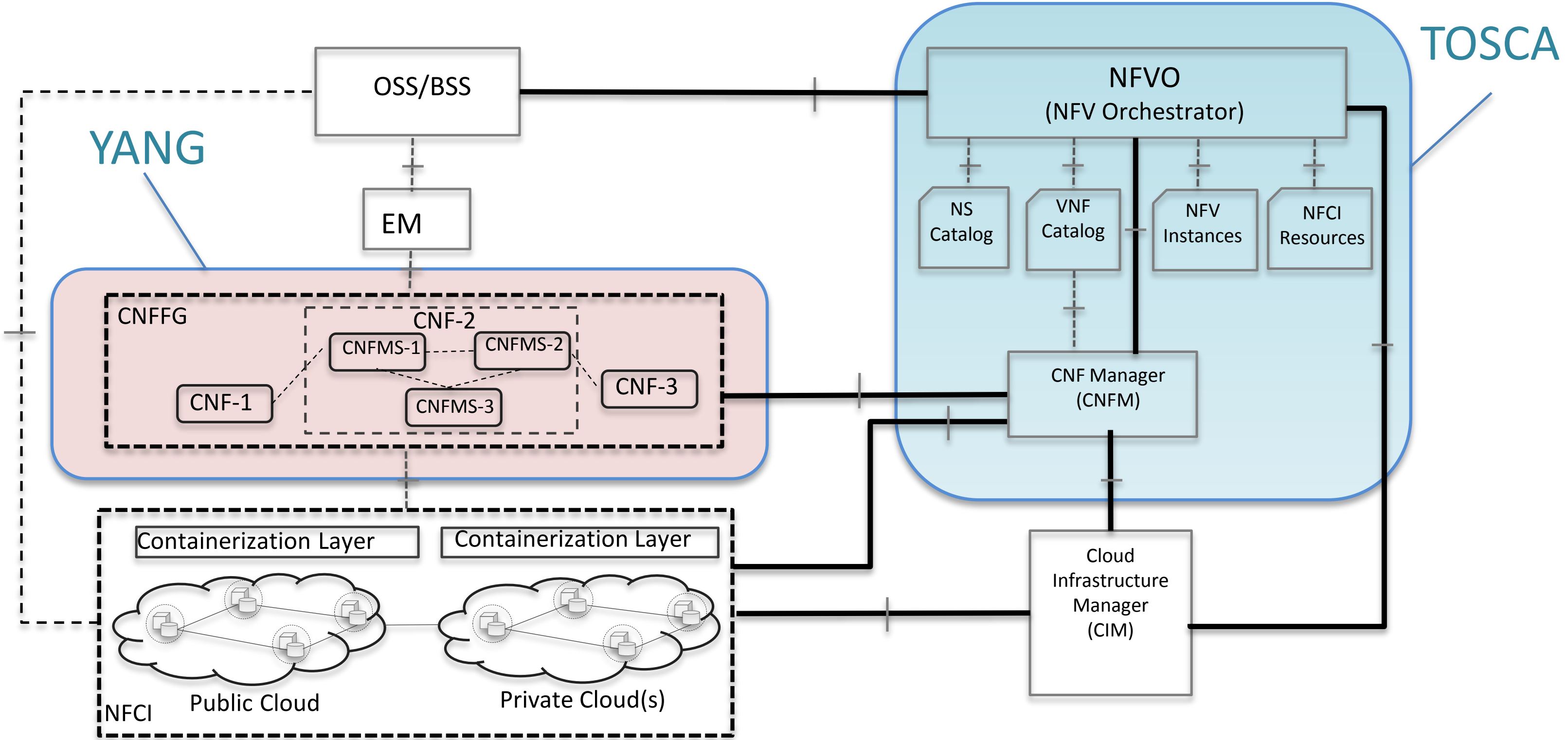
TOSCA or YANG

- TOSCA
  - ➢ Strong in orchestration
  - ➢ The goal is to deploy workload into the cloud
  - ➢ Capable of modeling topology with relationships
- YANG
  - ➢ Strong in device configuration
  - ➢ Capable of modeling attributes of virtual or physical devices

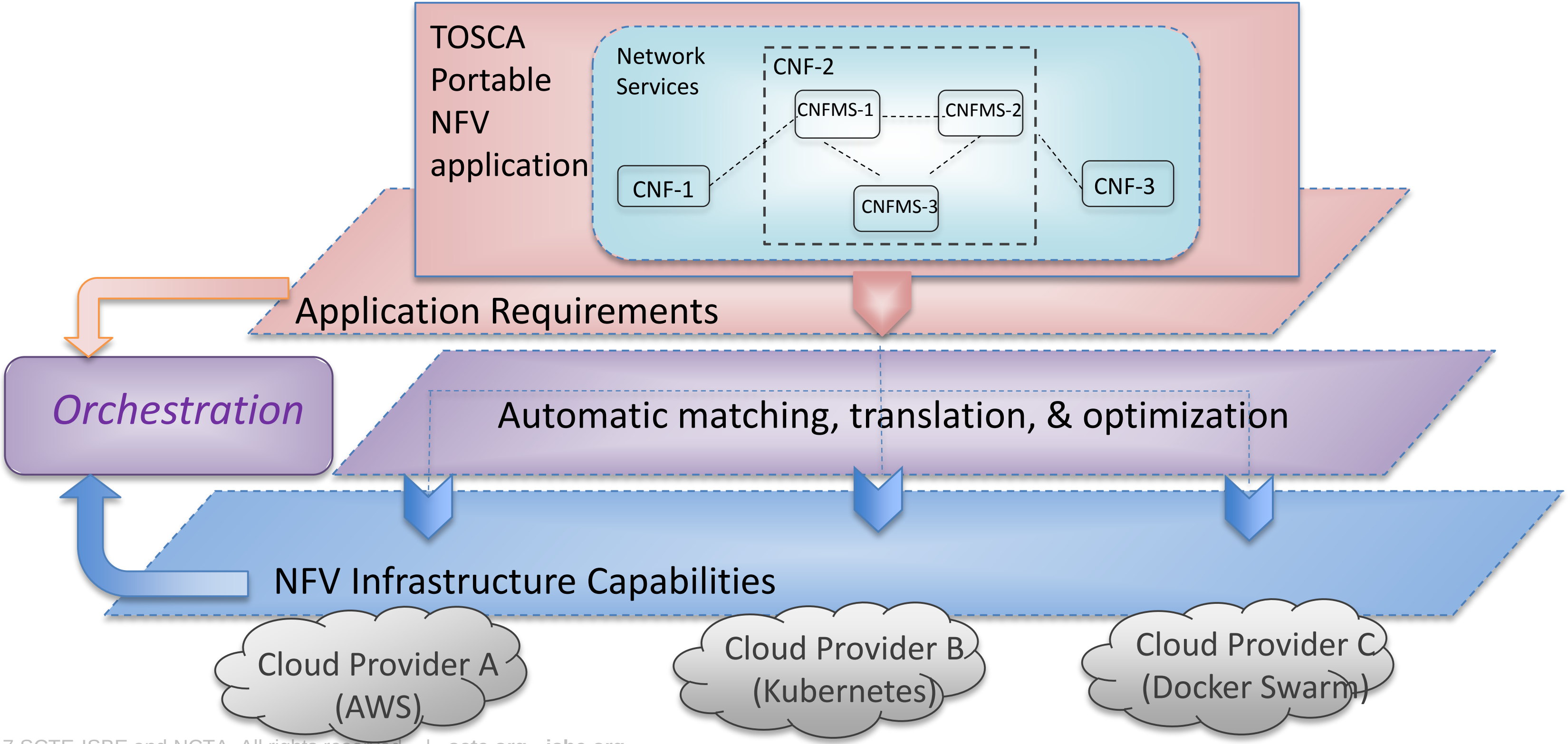
YANG?! TOSCA?!



TOSCA and YANG are complementary in NFV



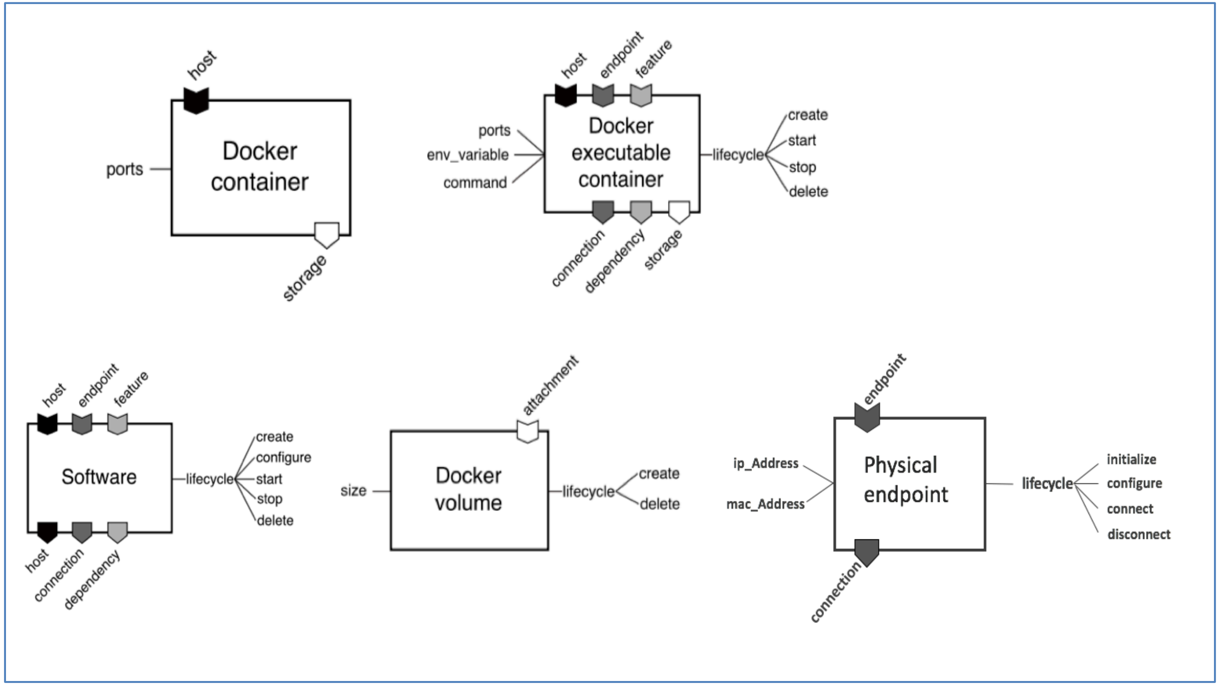
TOSCA to support NFV MANO portability


TOSCA Portable NFV application
Network Services
CNF-2
CNFMS-1
CNFMS-2
CNFMS-3
CNF-1
CNF-3
Application Requirements
Orchestration
Automatic matching, translation, & optimization
NFV Infrastructure Capabilities
Cloud Provider A (AWS)
Cloud Provider B (Kubernetes)
Cloud Provider C (Docker Swarm)

TOSCA Simple Profile in YAML specification

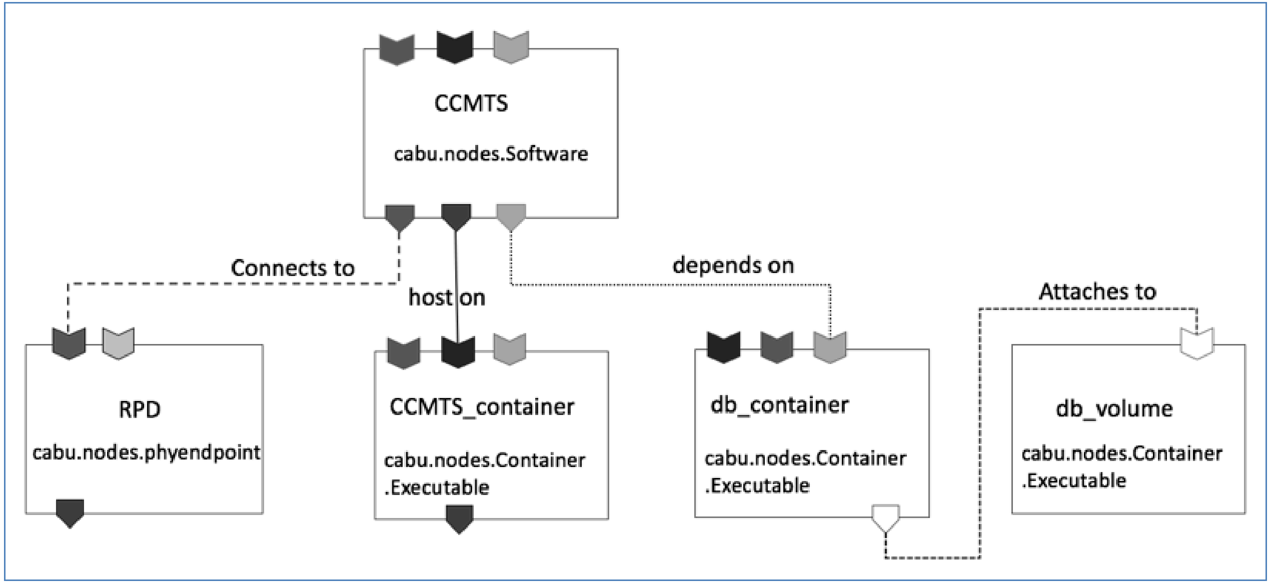
- Declarative meta model to describe cloud workloads
- TOSCA meta model contains
  - A graph of node templates and relationship templates
  - Node types
    - May define lifecycle operations
  - Relationship types
    - May define lifecycle operations
- TOSCA based orchestration engine
  - Uses the lifecycle operations to instantiate single components at runtime
  - Derives the order of the component instantiation using the relationship between components

TOSCA Custom types for an example CMTS Orchestrator



- Add custom node types
  - ➢ Docker container
  - ➢ Docker executable container
  - ➢ Software
  - ➢ Physical endpoint
- Add custom artifact types
  - ➢ Volume
  - ➢ Dockerfile

TOSCA model of an example CMTS Orchestrator



- Model the container deployment artifact
  - ➢ With custom node types
  - ➢ And custom artifact types
- Model the network topology
  - ➢ With standard TOSCA relationship
- Model lifecycle operations and workflow
  - ➢ With lifecycle interfaces in the node type

TOSCA Snippet for a CMTS Orchestrator in Cloud Native environment

```
node_templates:
ccmts:
  type: cabu.nodes.software
  requirements:
    - host: ccmts_container
    - connection: RPD1
  interfaces:
    standard:
      create:
        implementation: scripts/api/install.sh
        inputs:
          repo: <git_repo_where_the_scripts_are_installed>
          branch:  {get_input: api_branch}
       configure:
        implementation: scripts/api/configure.sh
       start:
        implementation: scripts/api/start.sh
       delete:
        implementation: scripts/api/uninstall.sh
```

```
ccmts_container:
  type: cabu.nodes.Container.Executable
  artifacts:
    ccmts_image:
      file: ccmts:1.0
      type: cabu.artifacts.Image
      repository: <cabu_docker_hub_url>
    requirements:
      - storage:
          node: ccmts_volume
          relationship:
            type: tosca.relationships.AttachesTo
            properties:
               location: /data/ccmts/db
ccmts_volume:
  type: cabu.nodes.Volume
```

SCTE · ISBE

THANK YOU!

YuLing Chen
yulingch@cisco.com
408-393-5606

Alon Bernstein
alonb@cisco.com