OOM Casablanca Goals

Roger Maitland & James MacNider - Amdocs

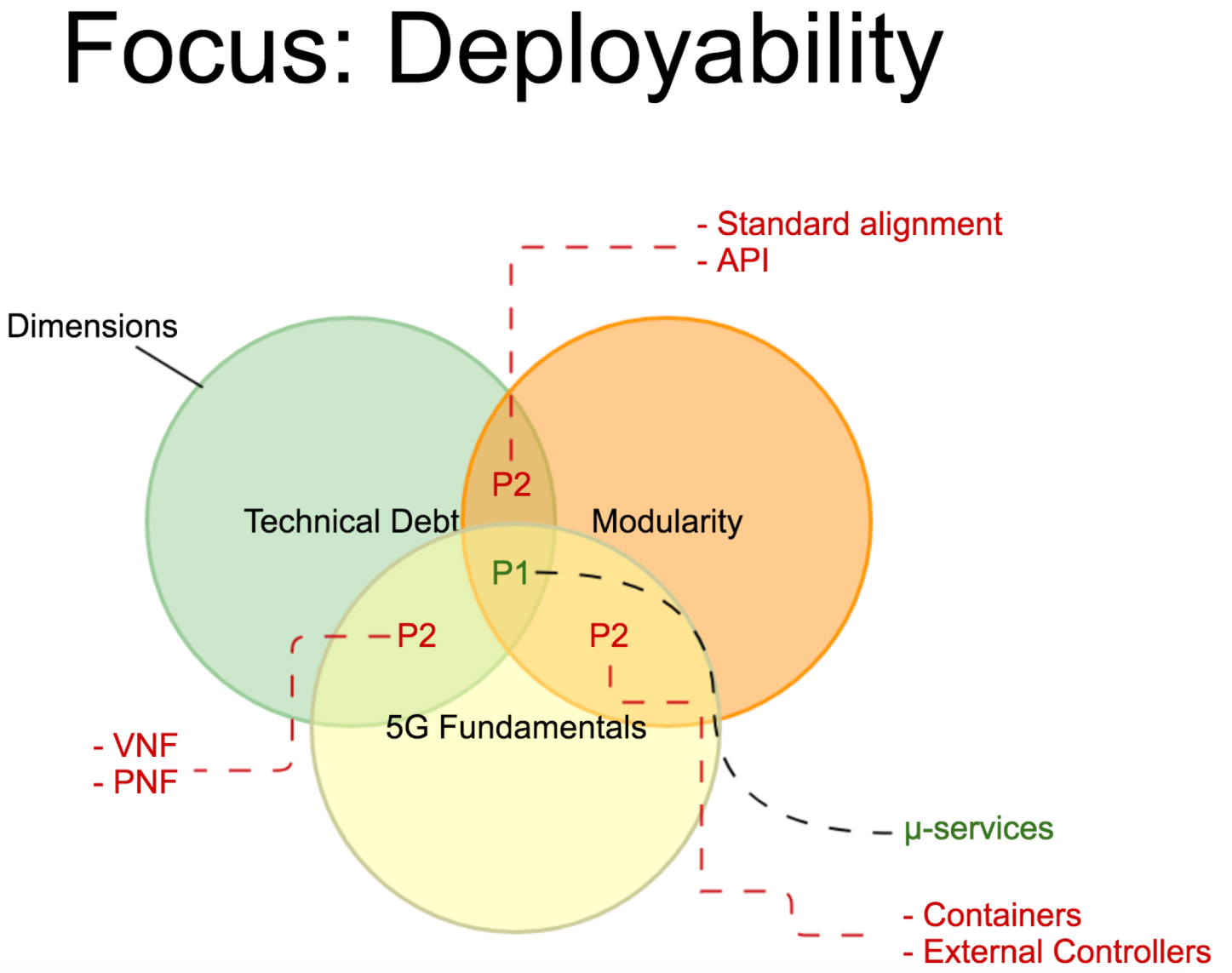
May 23, 2018

Addressing ONAP Platform “Deployability” with OOM

- Highest priority for Operators is “Deployability”
- Once past day 0 activities, “Deployability” becomes life-cycle management (LCM)
- OOM Casablanca will extend the Beijing functionality as described in this presentation

Note

Validation of Beijing release incomplete – rework in Casablanca likely



OOM Casablanca Highlights

New or Enhanced Functionality

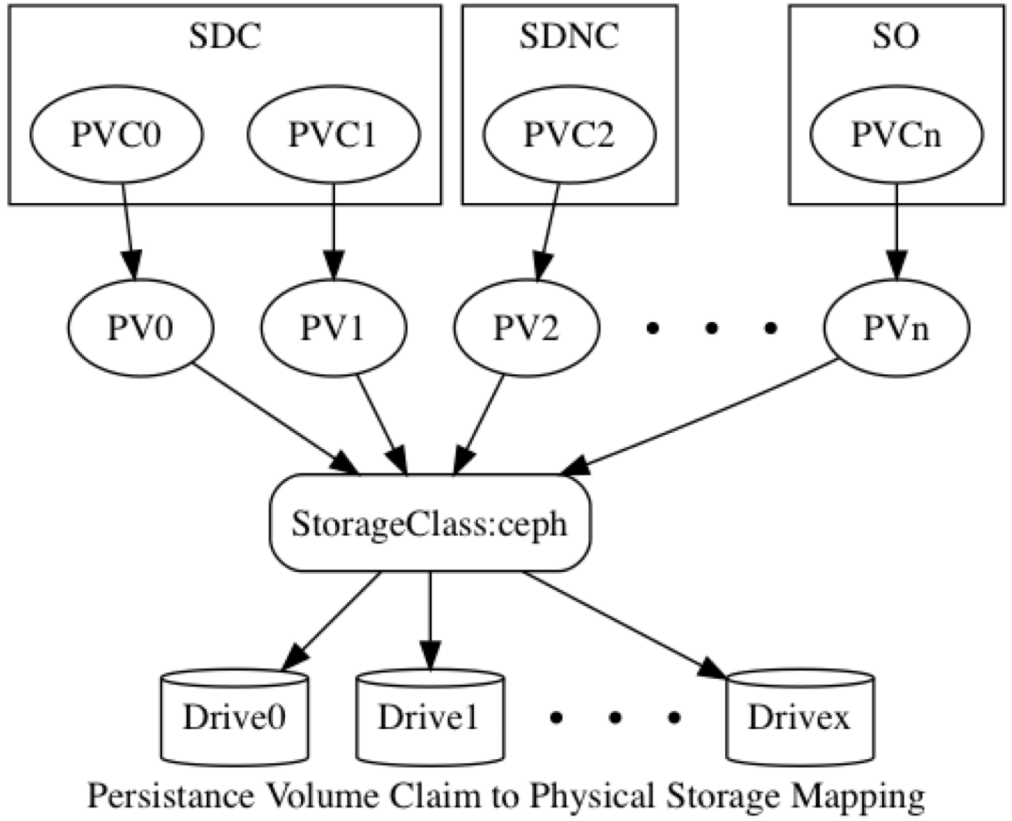
- Storage Architecture
- DBaaS
- Isolated Deployments / Static Images
- Pod Placement Rules
- Geo-redundancy

OOM Related Activities

- Chart Ownership
- CICD

Storage Architecture

- ONAP deployments will be diverse
- Likely storage technologies:
  - *Private*: Fibre Channel/iSCSI SAN/NAS, Ceph/Gluster software cluster, etc
  - *Public*: GCEPersistentDisk, AWSElasticBlockStore, AzureFile, AzureDisk, etc.
- Paradigm: Operators select **Storage Class**, ONAP components consume **Persistent Volumes** via **Persistent Volume Claims**
- Enables: Restartability & Backup and Restore



Persistance Volume Claim to Physical Storage Mapping

Data outside a Persistent Volume is Ephemeral

Database as a Service

Components that use the same database technology, could share a single cluster with separate schemas and credentials

Benefits:

- Reduces the ONAP platform footprint
- Common helm charts limit effort required by individual projects
- Project teams share a common redundancy strategy
- Simplifies cluster storage and management across the deployment

Three Steps:

1. Common DB Charts:
   - `kubernetes/common/postgres`
   - `kubernetes/common/mysql`
2. Clustered DBs:
   - `kubernetes/common/mariadb-galera`
3. Shared DBs:
   - Common DB instance, separate tables

Isolated Deployments / Static Container Images

Application container images should be ‘pre-baked’

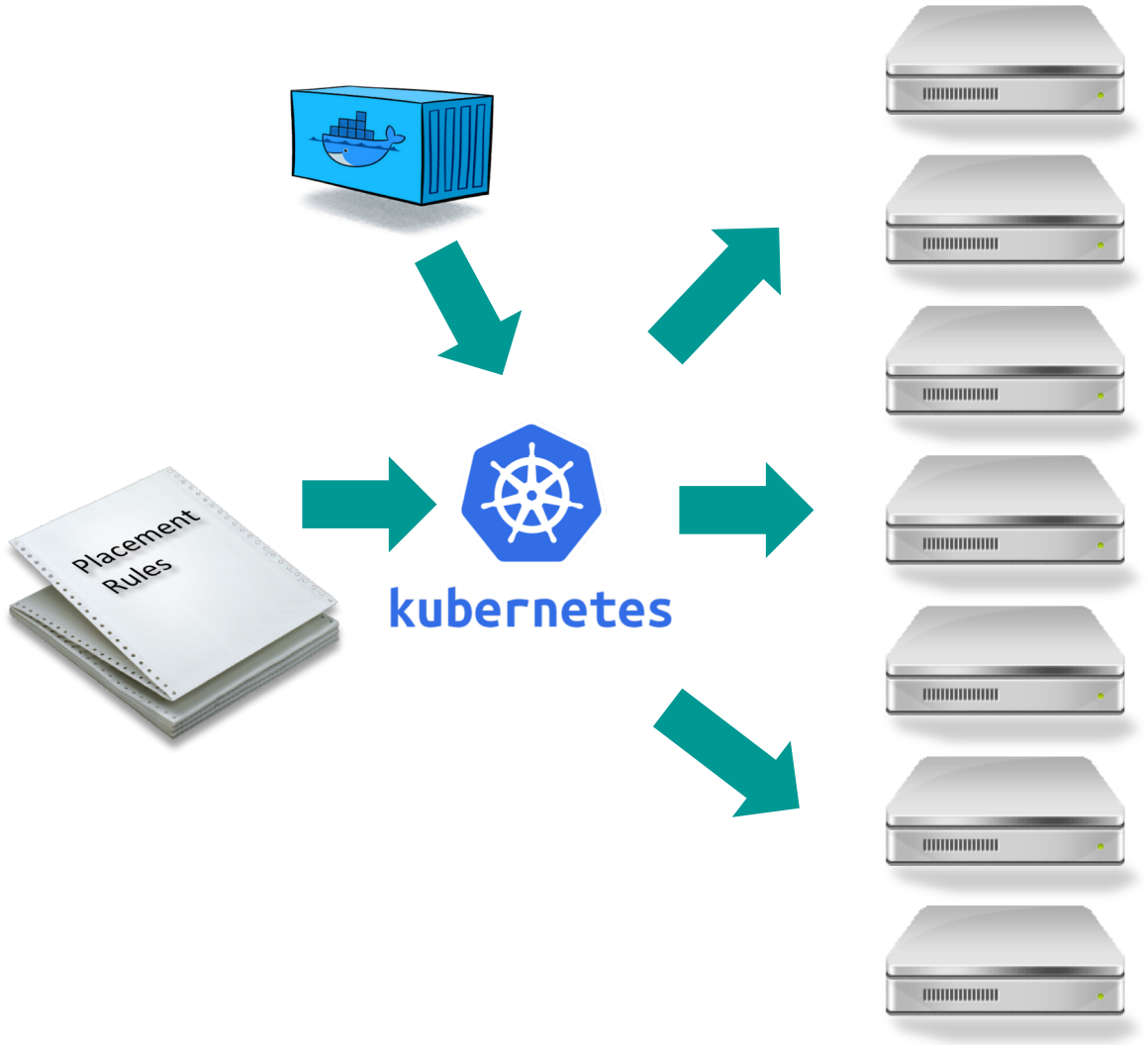
- ONAP deployment must be possible without access to the open internet
  - OOM enables redirection of the standard ONAP nexus with a 1-line change in OOM helm charts to get to docker images hosted on an internal nexus repository
  - `onap/values.yaml: repository: nexus3.onap.org:10001`
- No operations to install required packages during instantiation permitted
- All containers should be based upon on a small set of hardened and certified OS images
- Limit application-specific initialization where possible to reduce start times
  - DB schema creation would be an example exception

Benefits:

- Minimize time to instantiate when healing, scaling or migrating components
- Limit impact of outages and upgrades
- Deployment reproducibility
- Increased platform security

Pod Placement Rules

- Kubernetes distributes Containers to Nodes (physical or virtual)
- Currently largely unconstrained
  - StatefulSet – auto anti-affinity
- Adding:
  - Affinity / Anti-Affinity Rules
  - DeamonSet – one / node
    - Log Shipper / Logstash
    - Consul Agent
  - Resources - Memory / CPU
    - Ensure sufficient resources
  - nodeSelector – add zones
    - Distribute to multiple locations



Geo-Redundancy

Distribution of ONAP Components across a network is required. Two methods:

1. Cluster Federation – a single ONAP deployment is spread across multiple federated K8s clusters
   - Need to solve communication with duplicate components
2. MEF – two or more independent (partial?) instances of ONAP cooperating via MEF I/Fs
   - Existing ONAP functionality

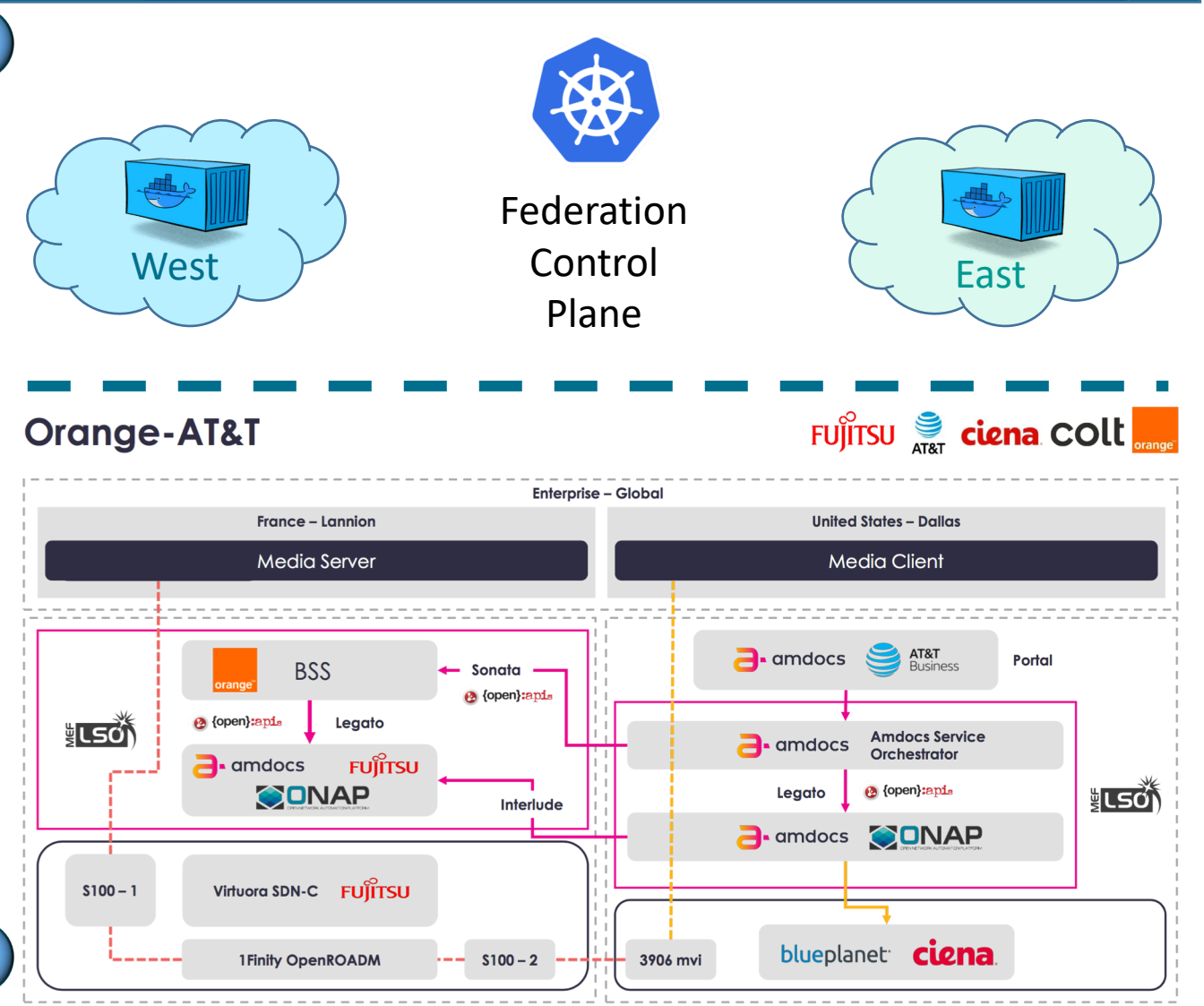


Chart Ownership

The project teams will take formal ownership of the helm charts for their projects in the Casablanca release

- OOM team will be primarily focused on deployment hardening, maintainability, and new features

Benefits:

- Project teams know best how to update the charts to address gaps for:
  - Changes to configuration files
  - Application scaling
  - Managing nodeports
- Project teams will to be fully empowered to implement and test their (expected) Casablanca requirements for:
  - Software Upgrade and Roll-back support
  - DB migration scripts (both forward and backward)
  - Resource requirements for each container
  - (Anti-)affinity rules

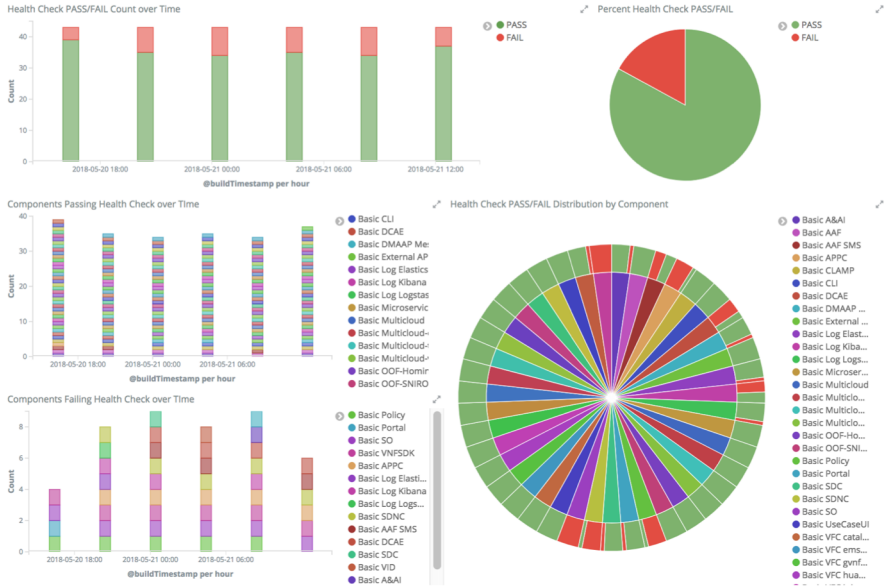
CI/CD

- Improve ONAP quality and predictability with Continuous Deployment

- Enhance the current CD systems
  - Centralized ONAP Helm repo (no need for local ‘make’)
  - Cooperation with OPNFV Clover project
  - Long lived CD system – changes applied as upgrades, failures removed with roll-backs
  - Kubemonkey to test resiliency
  - End-to-end test suites continuously doing integration testing

- How about two simple rules?
  1. All gerrit submissions must pass an CI/CD end-to-end test suite.
  2. The maximum size of a gerrit submission be constrained to some reasonable number (~5000 SLOC?).

  Simple but profound:
  - Essentially forces all projects to work “upstream first”
  - Dumping large submissions once a release would not be supported



Dashboard from one of the CD systems

More Information…

- OOM Wiki: OOM for Production-Grade Deployments
- OOM Documentation on ReadtheDocs