Clover Overview: Gambia release

April 16, 2018

Motivation

1. Future Telecom Services, e.g. 5G's top use cases
   - 50 billion IoT devices by 2020
   - Exceptional user experience AR/VR
   - Ultra low latency services (extending cloud to the edge), autonomous vehicles
2. Zero Touch
   - ONAP, ETSI ZSM
   - "DT: Brutal Automation is Only Way to Succeed" (Lightreading)
3. Application Innovation
   - Monetization with close user engagements
   - Data analytics driven automation and optimization

=> Call for Cloud Native, Web Scale, and Deployable VNFs

(More info see ONS 2018 LA Presentation)

What is Clover?

- A framework stack to support VNFs as cloud native micro-service applications and are deployed continuously At Scale
- VNF: any network software, including management, control or data
- Cloud native: as defined by CNCF
  - Containerized (Prerequisite)
  - Dynamically managed (Kubernetes, prerequisite)
  - Micro-service oriented (Clover's focus)
- Micro-service support
  - Service mesh (Istio)
  - Visibility, Traffic control policy, Security
  - A set of common micro-service utilities/components
- Continuously deployment
  - Modern CI/CD and policy pipelines for continuous Canary, A/B, Green/Blue... deployments

Problems to solve (1)

- Growth of micro-services often leads to “Container Sprawl”.
- This problem can be esp. prominent in SP and NFV (“At Scale”).
- **Solution**: A dedicated service layer: Service Mesh.

Container Sprawl examples

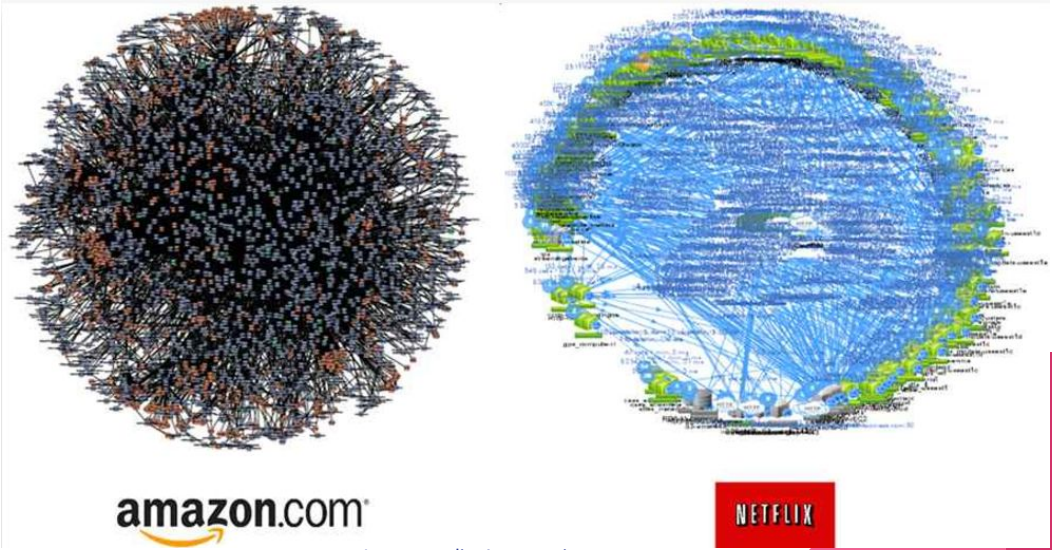
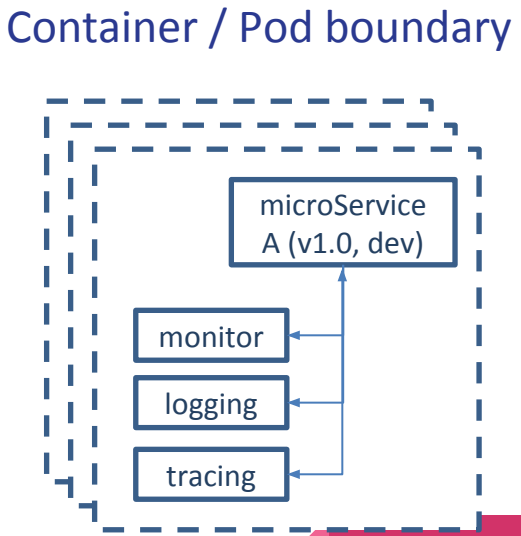


Image credit: Appcentrica

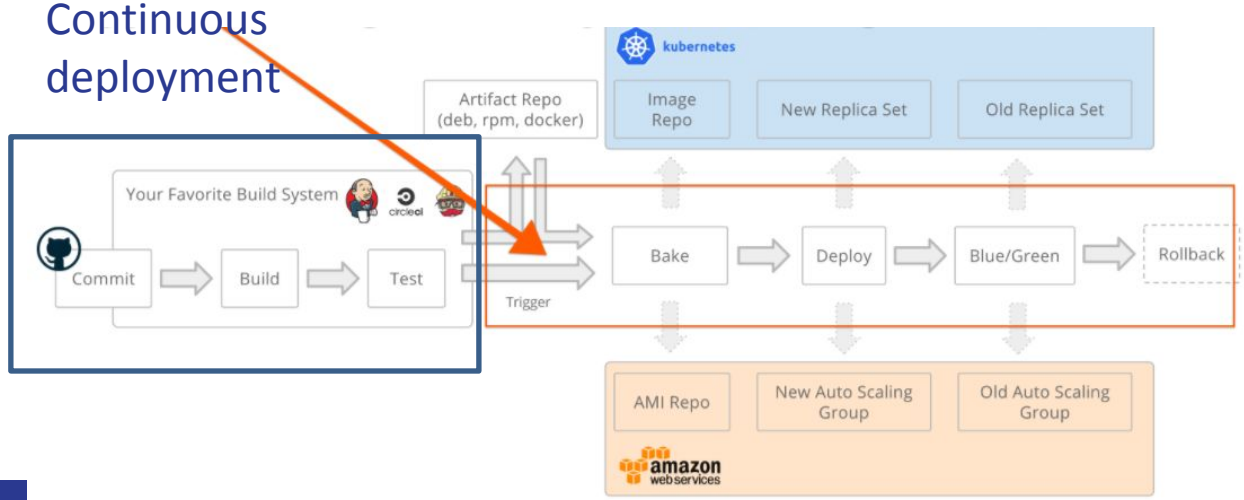
Problems to solve (2)

- Lack of a common set of tools for visibility and traceability: Monitoring, Logging, Tracing…
- Micro-services developed independently lack compatibility.
- Weak support for legacy software components.
- Lack of agnosticity to development and deployment environments.
- NFV use cases demand additional common utility services
  - Network tracing, nat, dpi, etc.
  - Storage for cloud native



Problems to solve (3)

- Continuous Delivery – a core benefit of cloud native applications is that its components (micro-services) can be updated live independently. Currently CI/CD pipeline in OPNFV **and other LFN** projects largely stops at CI level.
- Building a **reliable**, automated (zero-touch), platform independent and at-scale continuous deployment system is an important challenge facing NFV use cases. **We will address it by extending XCI to true CD (see** ONS Presentation**)**



Problems to solve (4)

- NFV specific use cases introduce new challenges to solve that are different from the general cloud native cases.
- **Networking:** network needs for gateway (intermediate systems) common in NFV use cases. Potential network performance issues, security issues. This is distinct from the typical container networking design options. Both need to be addressed.
- **Cloud native storage needs, and stateful VNF support.**
- Edge related challenges, to be solved in collaboration with edge related projects.
- **Required e**nhancements **and** additional common services for NFV
  - Interfacing with other OPNFV or LFN projects (e.g. ONAP)
- Best practice guides of developing cloud native VNFs
- **Influence/**contribute upstream for the identified NFV specific features.

Clover Framework

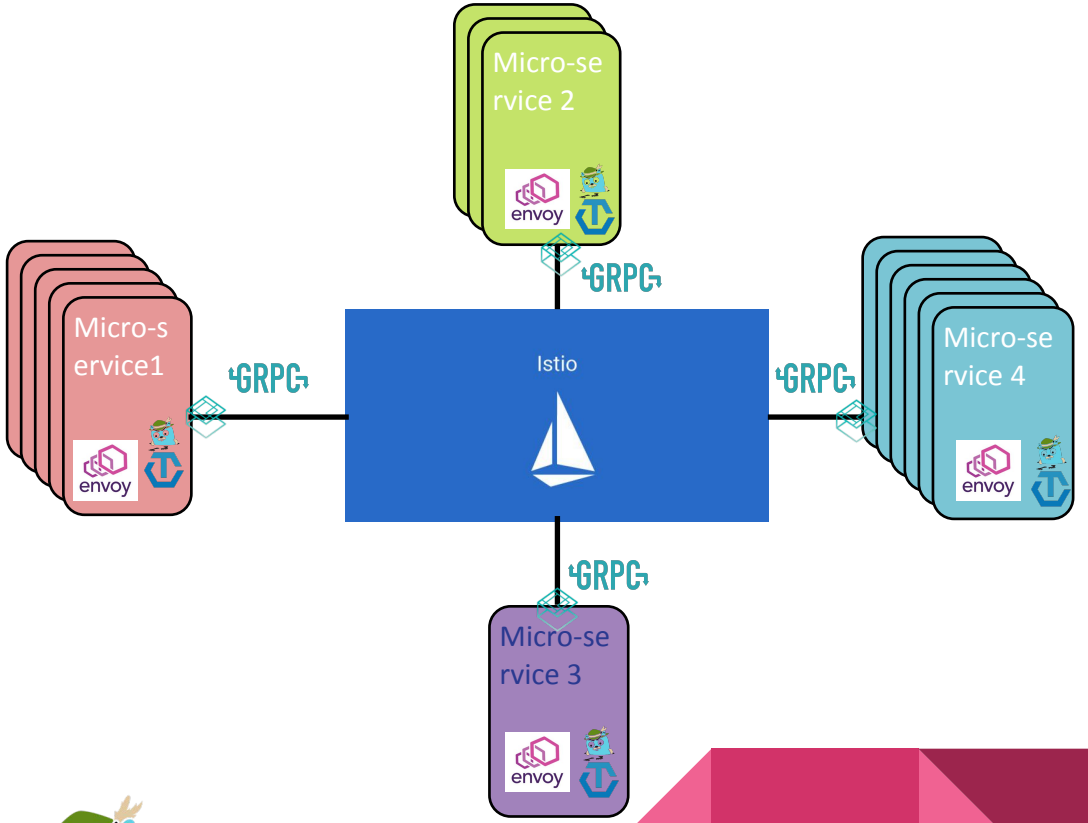
Out of scope; existing projects

Container

Orchestration

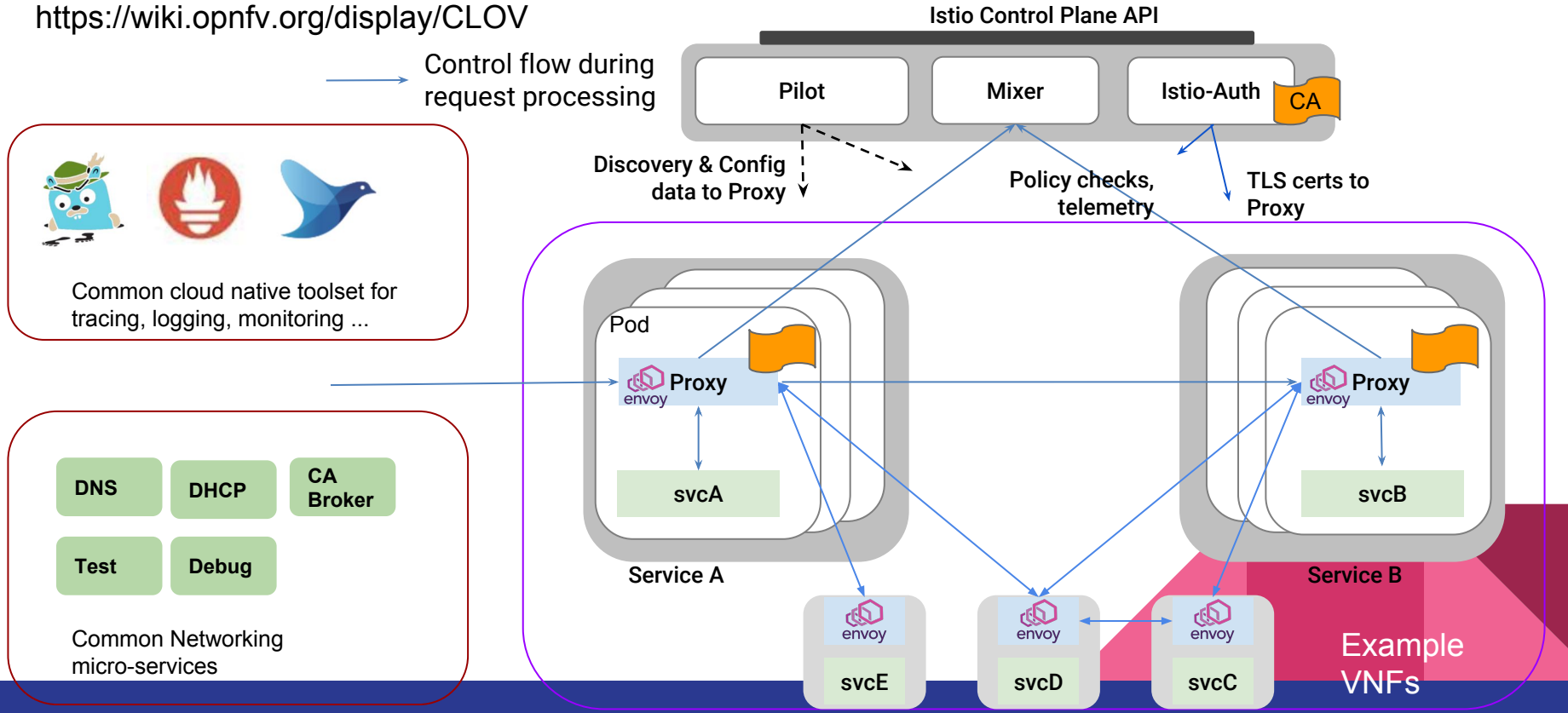
Continuous delivery

Common Services



Clover Framework

https://wiki.opnfv.org/display/CLOV

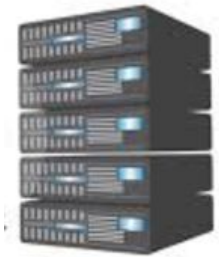


Target computing environments

Developer

Tiny

OPNFV



Small
(Edge
/Access)

OPNFV LaaS

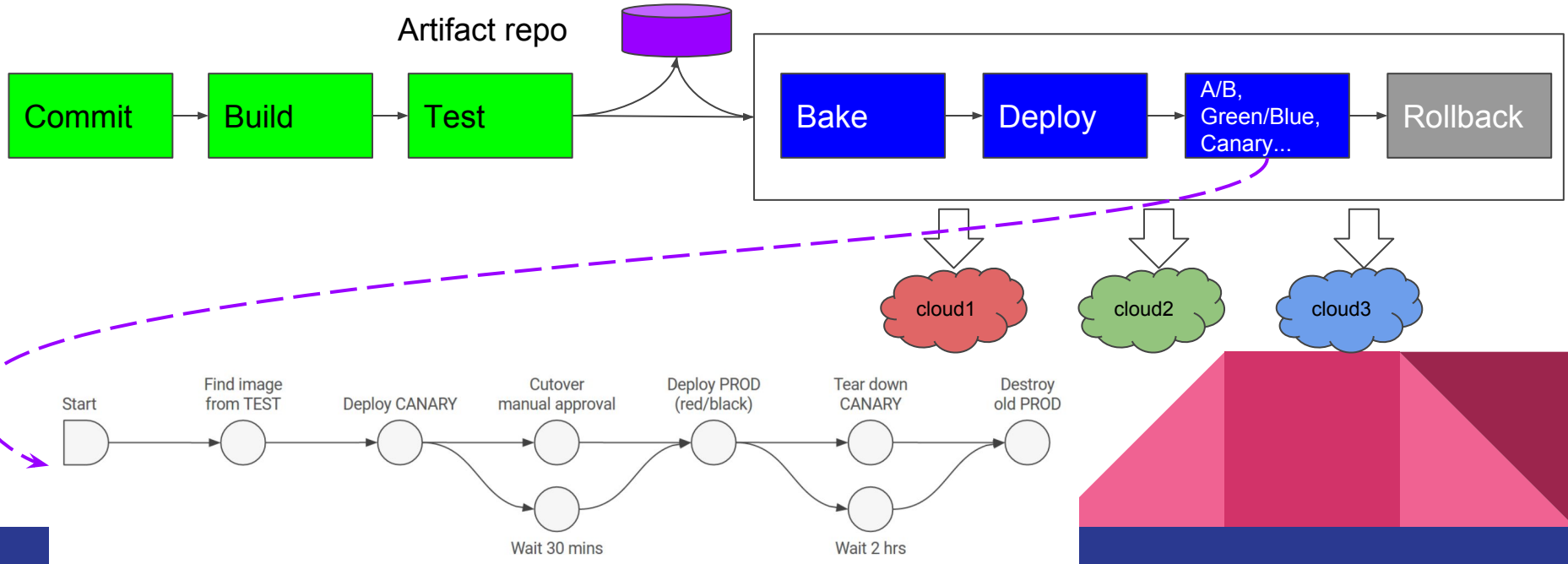
Medium
(Aggregation
/Regional)

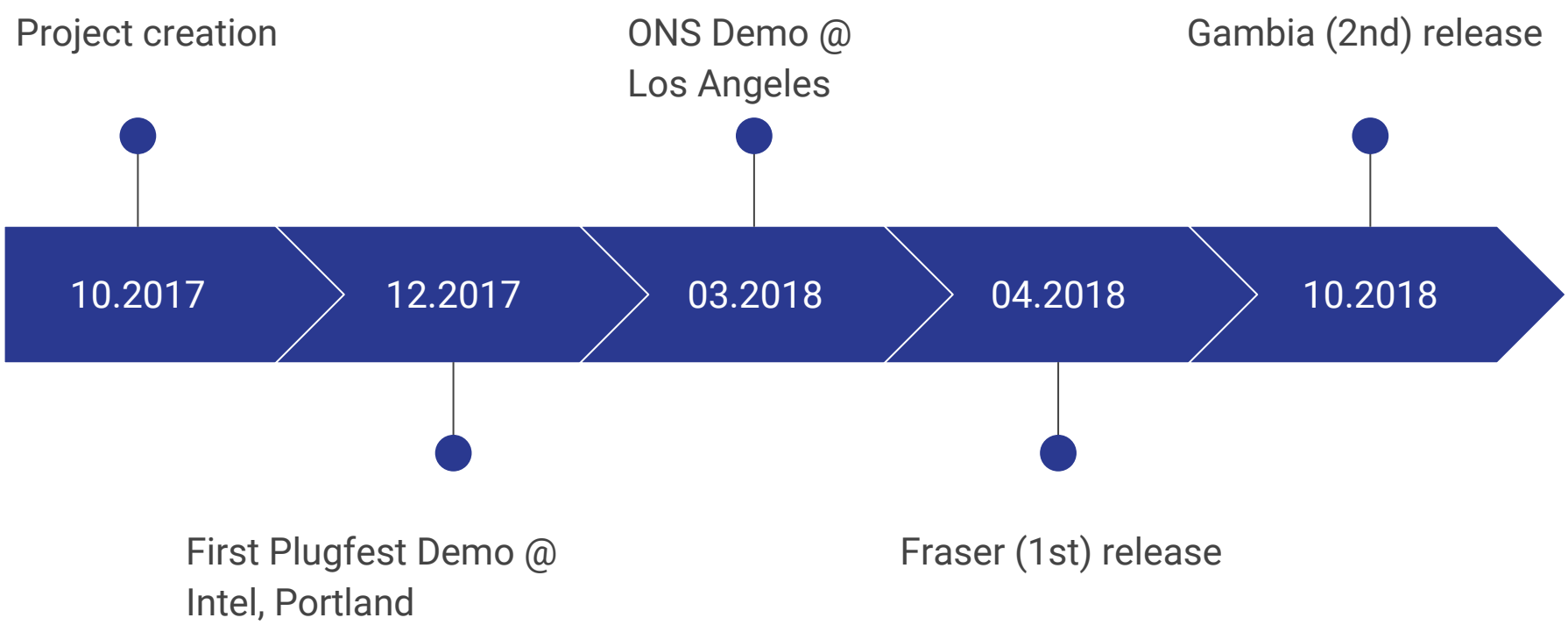
GKE, AWS, …

Huge
(Global data center)

Cloud native CI/CD

- CD: Continuous delivery (and deployment) for applications
  - Fast-paced user driven/data driven CI/CD is a core competency for Cloud Native DevOps.

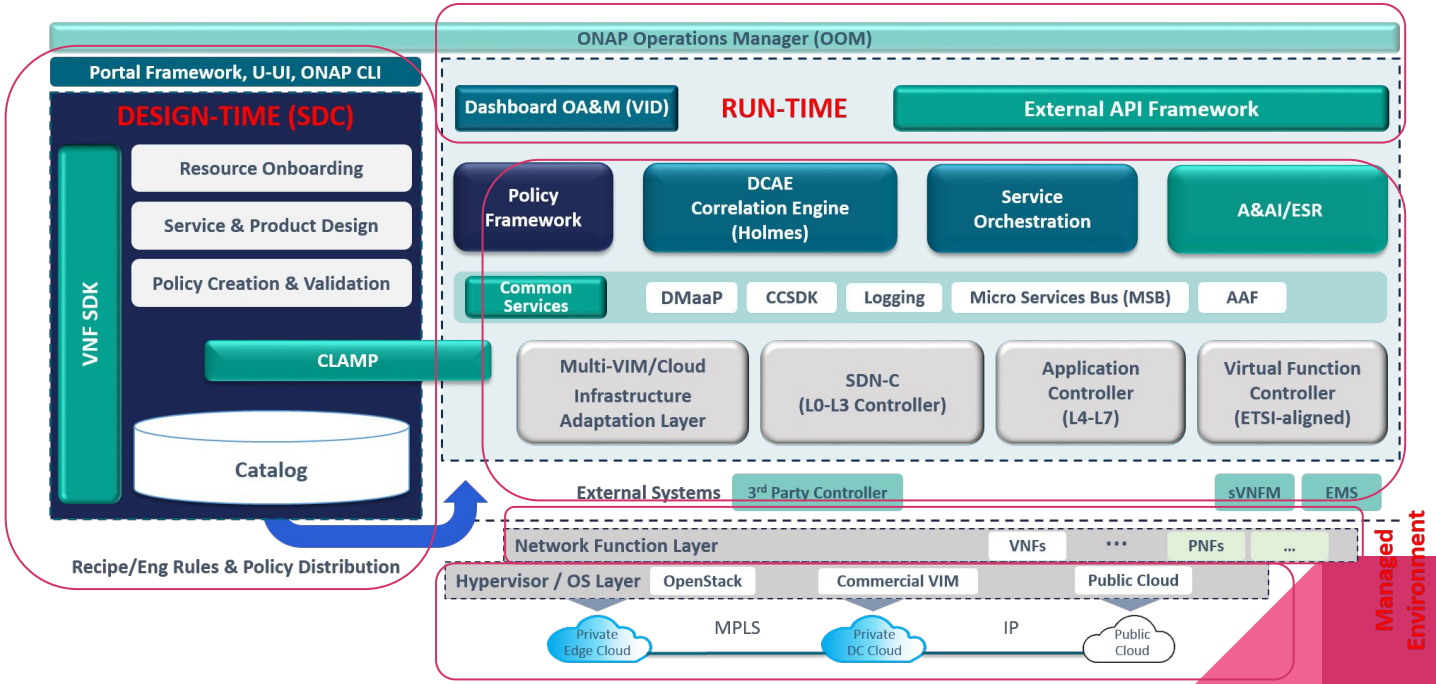



Project creation
ONS Demo @
Los Angeles
Gambia (2nd) release
10.2017
12.2017
03.2018
04.2018
10.2018
First Plugfest Demo @
Intel, Portland
Fraser (1st) release

Cloud Native for ONAP

For discussion

ONAP Architecture



Potential next steps

Other ideas?

- May explore common use cases
- ONAP may select 1 or 2 components to experiment
- The selected use cases or components may adopt CD on a shared long running testbed
- Joint release planning on any dependencies
- Continued discussions