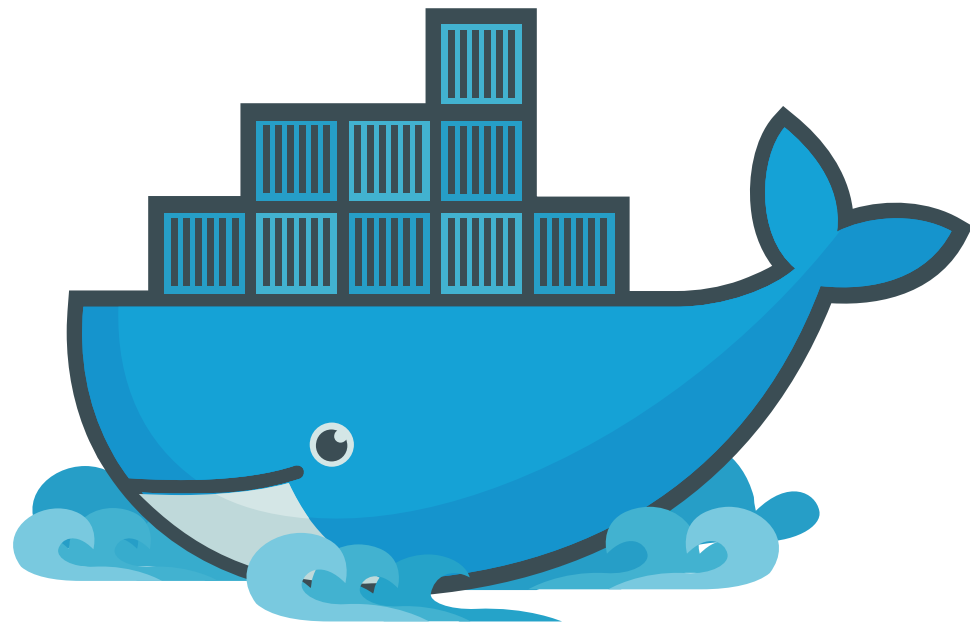


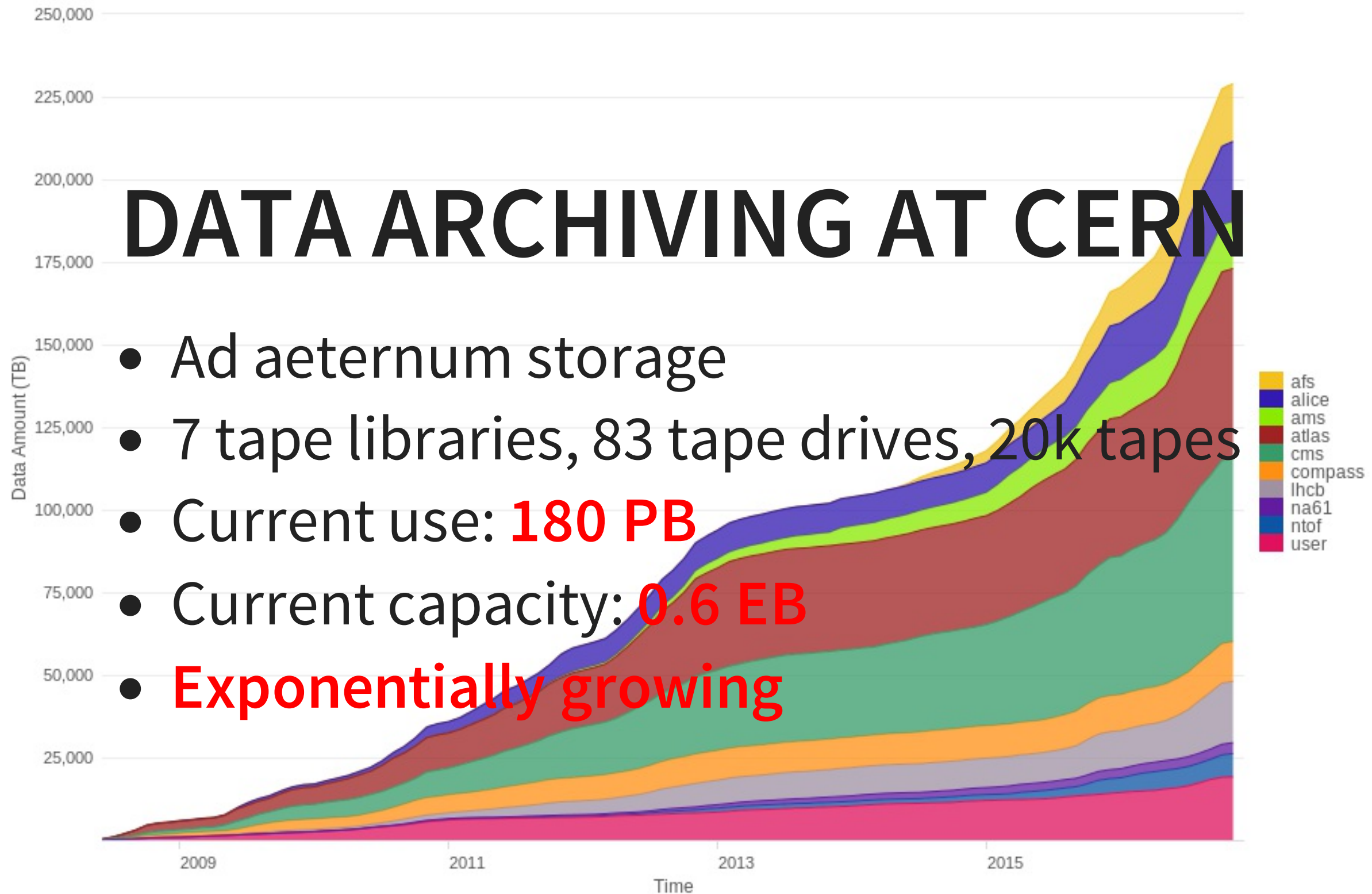
CI AND SYSTEM TESTS WITH



EOS + CTA DEVELOPMENT USE-CASE

Julien Leduc from IT Storage group [CERN](#)

DATA ARCHIVING AT CERN



- Ad aeternum storage
- 7 tape libraries, 83 tape drives, 20k tapes
- Current use: **180 PB**
- Current capacity: **0.6 EB**
- **Exponentially growing**

DATA ARCHIVING AT CERN

EVOLUTION

- EOS + tapes...
 - EOS is CERN strategic storage platform
 - tape is the strategic long term archive medium
- EOS + tapes = ❤️
 - You just met CTA: CERN Tape Archive

CTA + EOS DEVELOPMENTS

Tightly coupled software ⇒ **tightly coupled
developments**

Extensive and systematic testing is paramount to limit
regressions

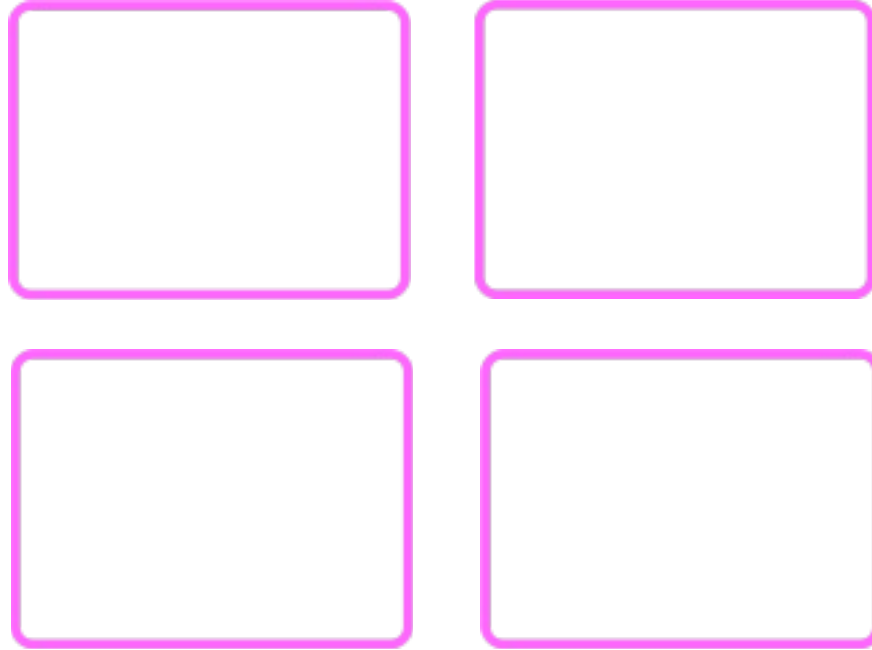
CASTOR INTEGRATION TEST

- Easy situation:
 - all components are within **one git repository**
 - Puppet deploys development instances on VMs
 - **Limited external dependencies** per instance: 1 database, 1 virtual tape library

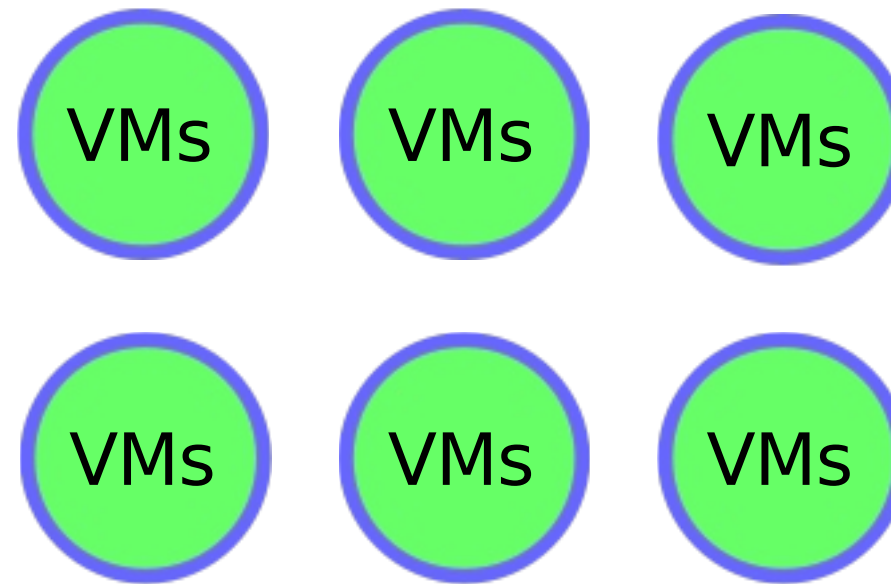
CASTOR INTEGRATION TESTS

- But several issues with VM/Puppet approach:
 - deploying a developer instance from scratch takes **looonnng time...**
 - code changes in CASTOR often require Puppet **manifest change**
 - **real tape hardware** tests are way further down the road in separate hostgroups, environments...
 - which implies **ad hoc** developer tests...

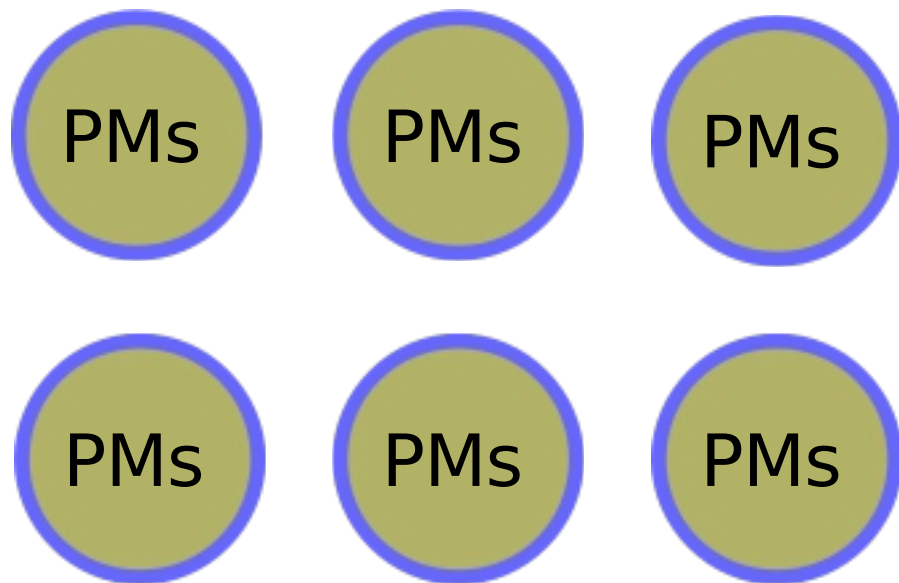
Dev environment



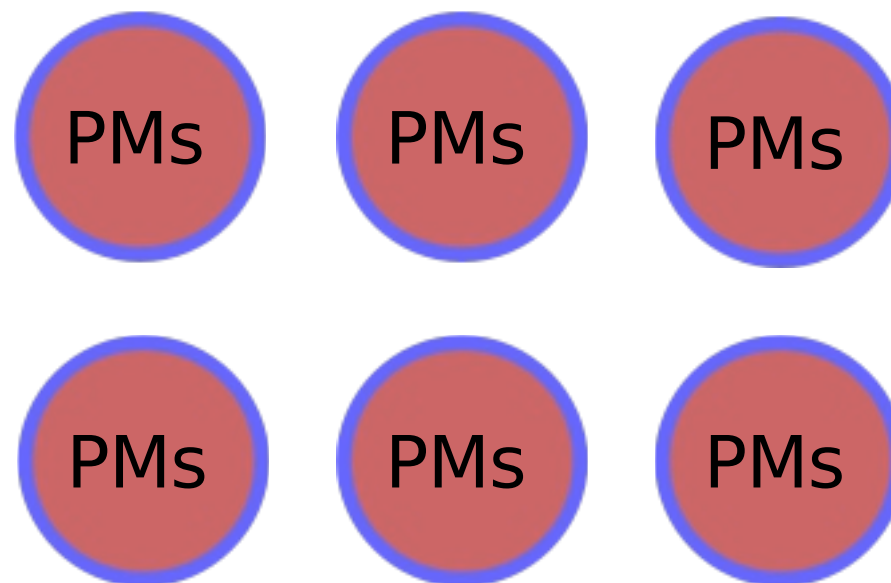
Dev tests



Prod QA



Prod master



CTA+EOS INTEGRATION TESTS

- Complex situation:
 - **2 distinct software projects**
 - **More external dependencies** per instance: 1 database, 1 virtual tape library, 1 objectstore

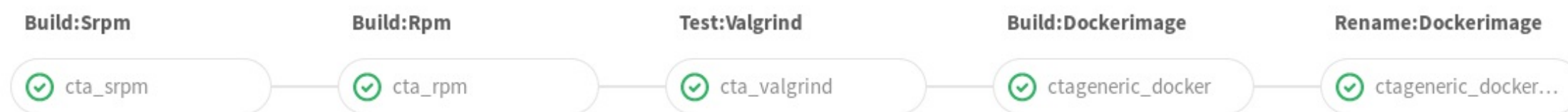
CTA+EOS INTEGRATION TESTS

- How to fix everything?
 - I am **lazy** and **impatient**
 - no manual operation → **CI**
 - make it **fast**
 - Must allow similarly **easy beta testing deployments** for administrators/users (simple and bulletproof)
 - In a second time: how to test **real tape hardware?**

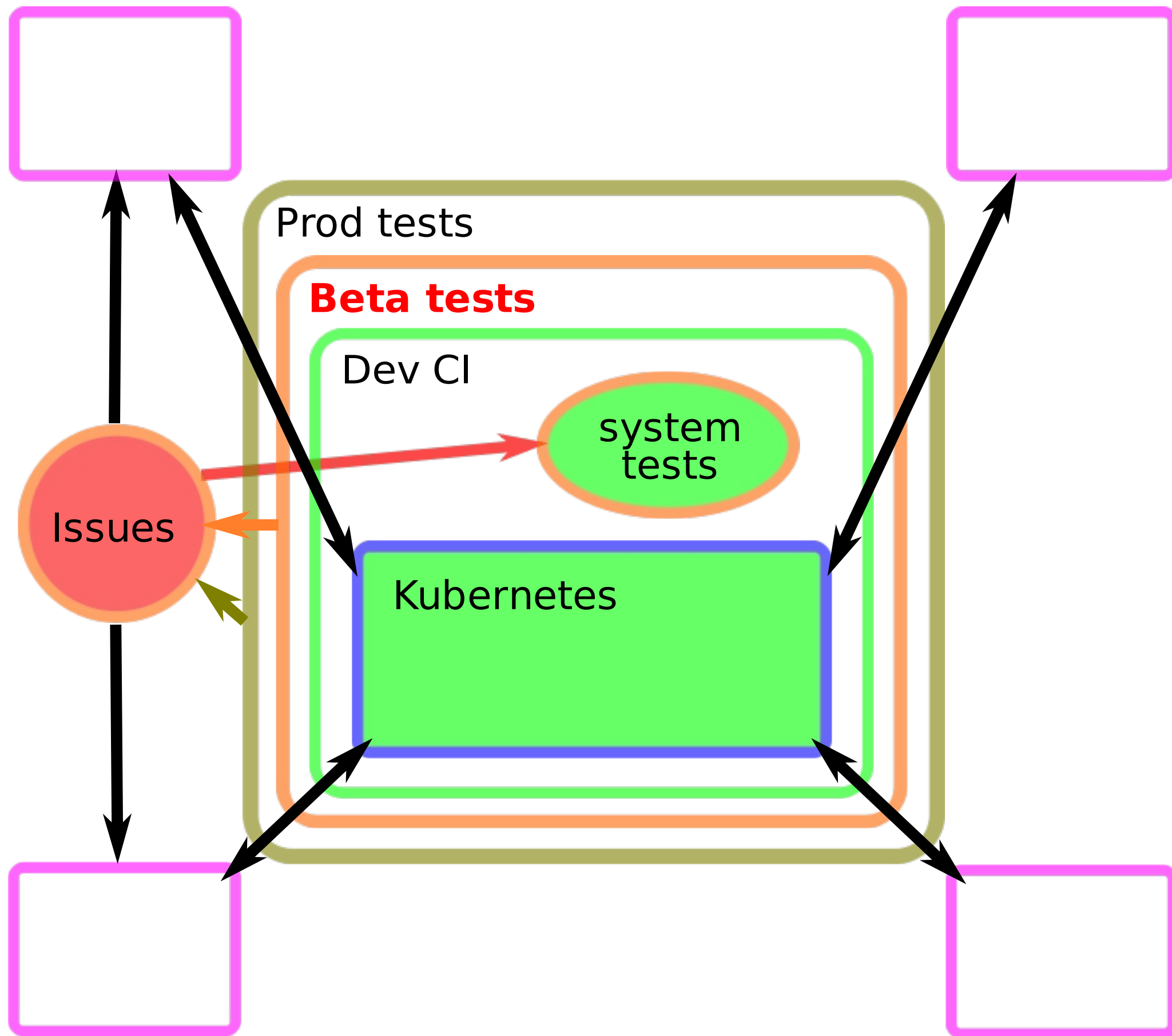
CTA CI



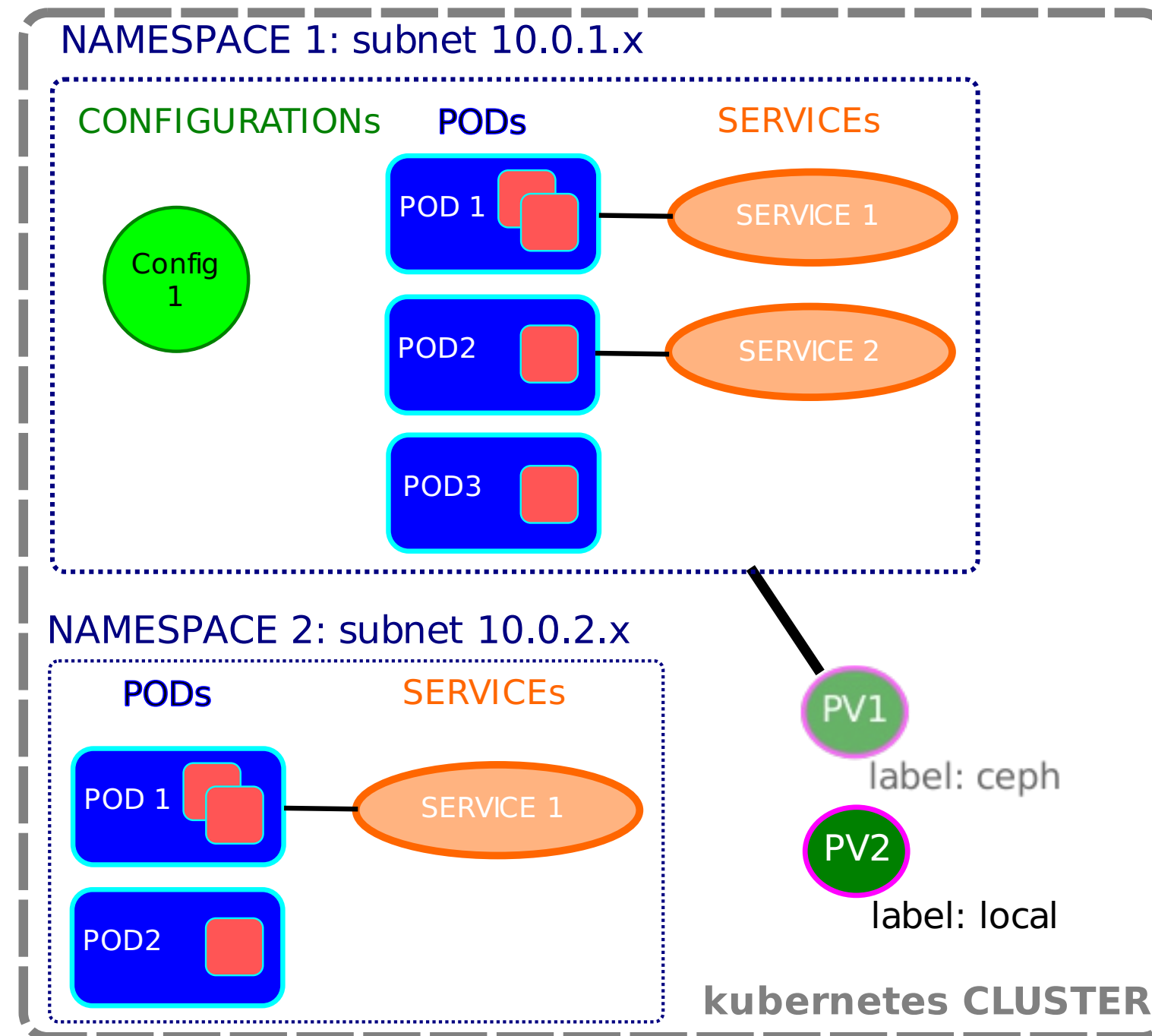
Implemented in CERN Gitlab instance



- Build software: CTA RPMs available as **artifacts**
- Build and publish a **generic Docker image** in gitlab registry
 - Contains **all required RPMs for instantiation** (CTA artifacts, specific EOS version, specific XROOTD version)
- Run **system tests** in custom kubernetes cluster



BASIC KUBERNETES CONCEPTS



KUBERNETES RESOURCES

System tests on dedicated kubernetes clusters

- One **Puppet deployed** kubernetes cluster per developer on one VM
- Kubernetes resources per cluster:
 - 1 **Oracle database** (+ unlimited sqlite accounts)
 - 1 **Ceph objectstore** (+ unlimited local objectstores)
 - 10 **Virtual tape libraries**: 2 tape drives, 10 tapes

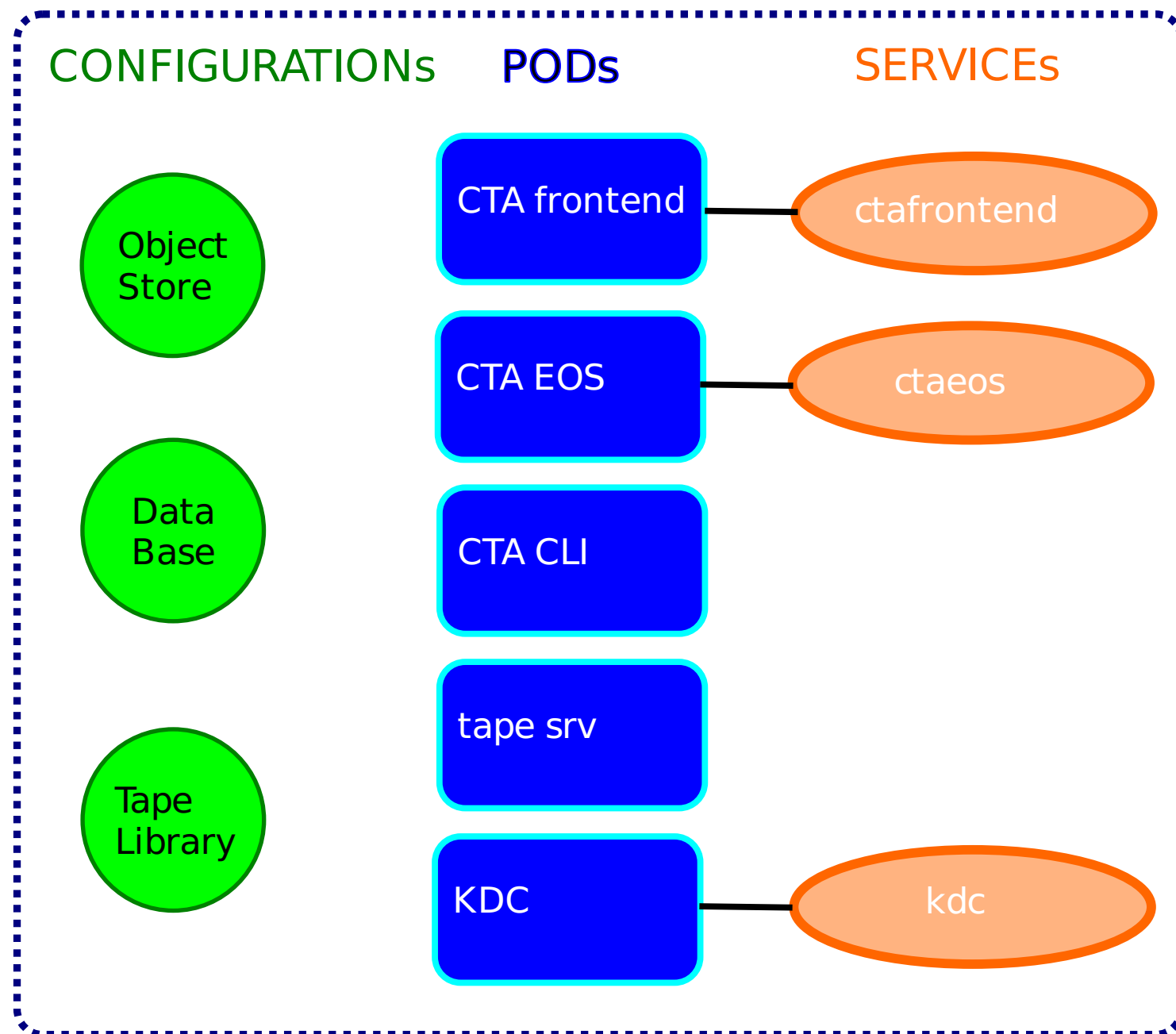
INSTANTIATING A TEST



- Create k8 **Namespace**
- Instantiate all **Services** in the namespace
- Consumable resources are implemented as **Persistent Volumes**
 - Issue a **Persistent Volume Claim** with selector
 - Instantiate associated **Configuration** in the Namespace
- Instantiate all the **Pods** with their associated containers to implement all the services
- Wait for all the pods to be ready

INSTANTIATING A TEST

NAMESPACE

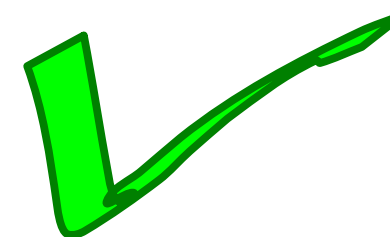


SYSTEM TEST

```
setup EOS WFE  
xrdcp file -> ctaeos  
- is it on tape?  
remove EOS disk copy  
retrieve from CTA  
- is is back in EOS?
```



GITLAB



REAL TAPE DRIVE TESTS

- Deploy Puppet manifest on **real hardware**
- Add **physical tape library resources** in hiera
- **Increase timeouts** for system tests



We can deploy the same kubernetes instance on real tape hardware and run exactly the same system tests.

THE END

- Very powerful approach **addresses and federates all our use cases**
- Fast, flexible, isolated and self contained in software repository

TO DO

- **Evangelise**
- Write and structure more system tests
- Bulletproofing **reproducibility** for regression tests
- Evaluate possible production use 😊

Activities Firefox

root@ctadevjulien:~/cta-orchestration/CTA

```
[root@ctadevjulien CTA]# export NAMESPACE=ctatest
[root@ctadevjulien CTA]# ./create_instance.sh -n $NAMESPACE
Creating ctatest instance
namespace "ctatest" created
creating configmaps in instance
configmap "objectstore-config" created
configmap "database-config" created
Requesting an unused MHVTL librarypersistentvolumeclaim "claimlibrary" created
.configmap "library-config" created
got library: sg32
creating services in instance
service "ctacli" created
service "ctaeos" created
service "ctaf frontend" created
service "kdc" created
creating pods in instance
pod "init" created
Waiting for init.....OK
Launching pods
pod "ctacli" created
pod "tpsrv" created
pod "ctaeos" created
pod "ctaf frontend" created
pod "kdc" created
Waiting for other pods.....OK
Waiting for KDC to be configured...[]
```

Tue 09:42

Cluster - ctadevjulien.cern.ch - Mozilla Firefox

Cluster - ctadevjulien... x Kubernetes Cluster x +

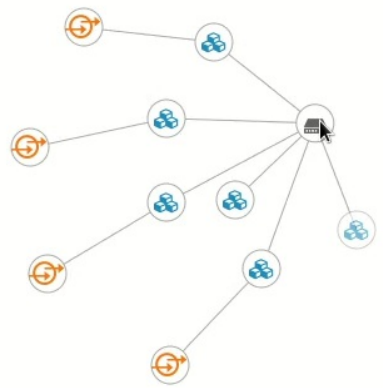
https://ctadevjulien.cern.ch:9090/kubernetes#/topology/ctatest

CENTOS LINUX

ctadevjulien.ce... Dashboard Cluster

Project: ctatest

Overview Nodes Containers Topology Details Volumes



Select an object to see more details.

- Pod**
Pods contain one or more containers that run together on a node, containing your application code.
- Replication Controller**
Replication controllers dynamically create instances of pods from templates, and remove pods when necessary.
- Service**
Services group pods and provide a common DNS name and an optional, load-balanced IP address to access them.
- Node**
Nodes are the machines that run your containers.