EPIK8S

Epics on K8S

Why LNF transitioning to Epics

- ❑ No more resources to continue developing custom CS
- ❑ Many new projects incoming: the most important EUpraxia
- ❑ Epics well integrates into our pre-existing CS this why we chosen to go with Epics instead of Tango

But:

- ❑ No profound knowledge
- ❑ Great change dealing with (in principle tons of) variables instead of objects and functions
- ❑ Many distributed DBs on files can be a problem not a clear way to keep track of ioc/variables/initializations/devices
- ❑ Many different GUI: memdm, edm,caqtm, pydm, css, phoebus and many different implementation for the control of the same class of devices
- ❑ New promising epics services in the ecosystem of phoebus needs an IT effort to be implemented (mongo, elasticsearch, kafka)

Epics on K8S: Motivations

- Limited manpower requires:
  - IT resources focus on IT infrastructure not on Services required by Control System
  - Control System resources focus on application and control services not on IT infrastructure!

- Applying modern industry standards learnt with !CHAOS to manage client/server EPICS services
  - Containers
    - Package IOC software and execute it in a lightweight virtual environment
  - Kubernetes
    - Centrally orchestrate all IOCs and services at a facility
  - Helm Charts
    - Deploy IOCs/Services into Kubernetes with version management
  - GIT Repositories
    - Source, container and helm repositories manage all the above assets. No shared filesystems required.
  - Continuous Integration / Delivery
    - Source repositories automatically build assets from source when it is updated

- Could simplify workflows to promote sharing, collaboration (i.e INFN rete epics)
- Could simplify building new infrastructures integrating pieces and not with the ”reinvent the wheel” approach
- Could simplify building control systems for small experiments without having EPICS/IT particular knowledge

EPIK8S

Epics on K8S: Motivations
Ideal timing window!

Opportunity to apply on a real and small facility (ELI now SSRIP)

Opportunity of our epics network that could enrich and contribute to have common workflow, catalogues and tools

Opportunity to work with Diamon pushing on the same direction (https://epics-containers.github.io/main/index.html)

New IT technologies (kafka, elasticsearch…) now used in modern epics ecosystems best fit in cloud aware infrastructures like k8s. Sooner or later epics combined with dockerization and orchestration will be the standard for new facilities.

Desiring to lay the groundwork for the future by drawing from the past?

EPIK8S

EPIK8S keywords & Benefits

- Everything on **GIT**
  - ✓ Traceability
  - ✓ Reproducibility
  - ✓ Continuous Integration
- **ArgoCD**
  - Single Source Of Truth (keep the cluster aligned with GIT)
  - Convenient GUI to manage applications(ioc,services,ui) lifecycle
  - super easy cluster disaster recovery and rollback (create/recreate everything from GIT)

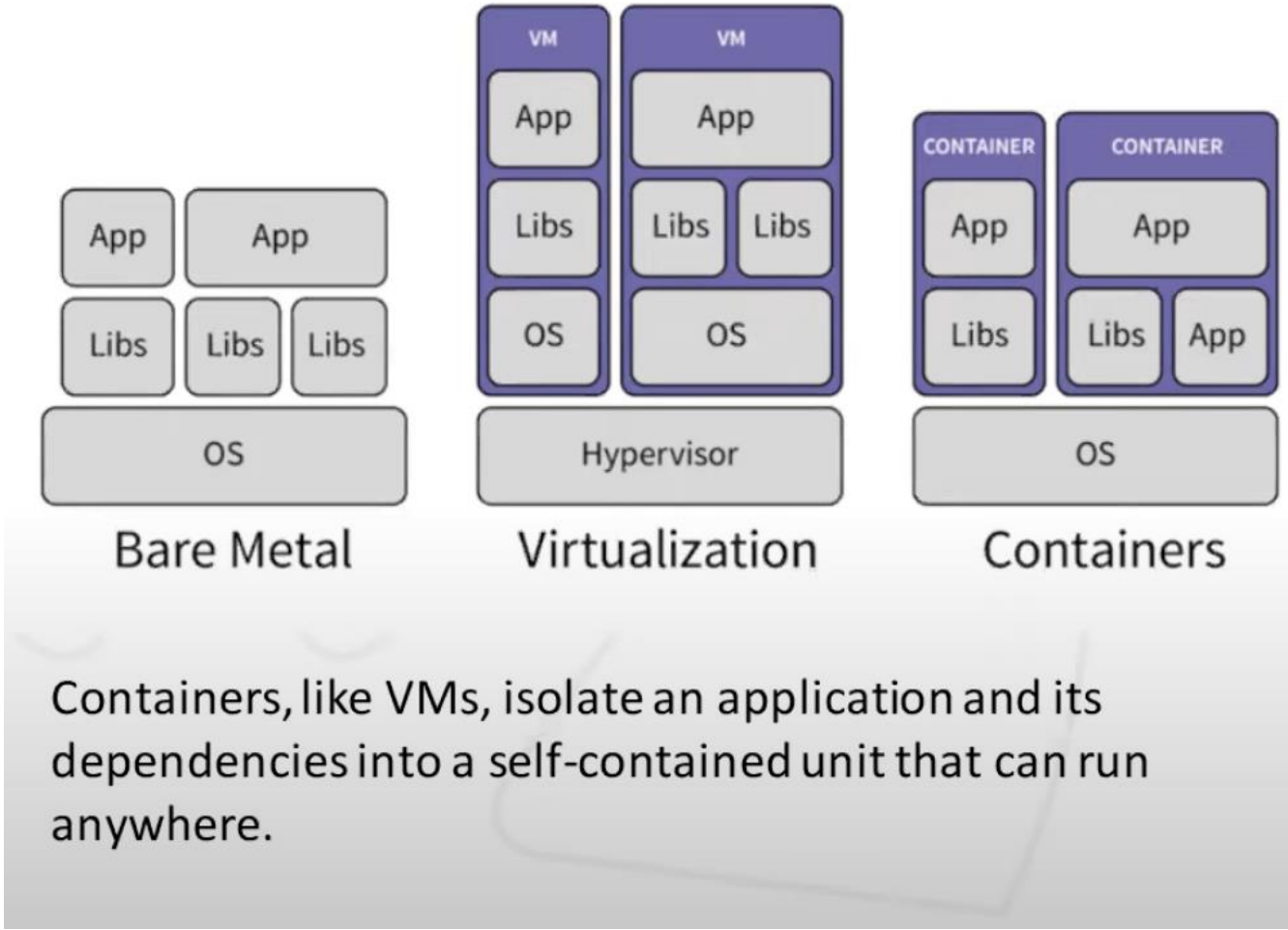
- **K8s** and Dockerized Infrastructure
  - Containers are decoupled from the host OS and each other.
  - Isolation protects against most security vulnerabilities
  - Run anywhere: develop, test, demo on a laptop or home machine
  - Kubernetes provides economy of scale through centralized:
    - Software deployment and management
    - Logging and Monitoring
    - Resource management: Disk, CPU, Memory
    - Remove maintenance of internal management tools
    - Remove need for shared filesystem
    - Remove the need to build a binary for every IOC

EPIK8S

EPIK8S benefits

- ❑ Auto start IOCs when servers come up
- ❑ Restart crashed IOCs
- ❑ Manually Start and Stop IOCs
- ❑ Allocate the server which runs an IOC
- ❑ Move IOCs if a server fails
- ❑ Throttle IOCs that exceed CPU limit
- ❑ Restart IOCs that exceed Memory limit

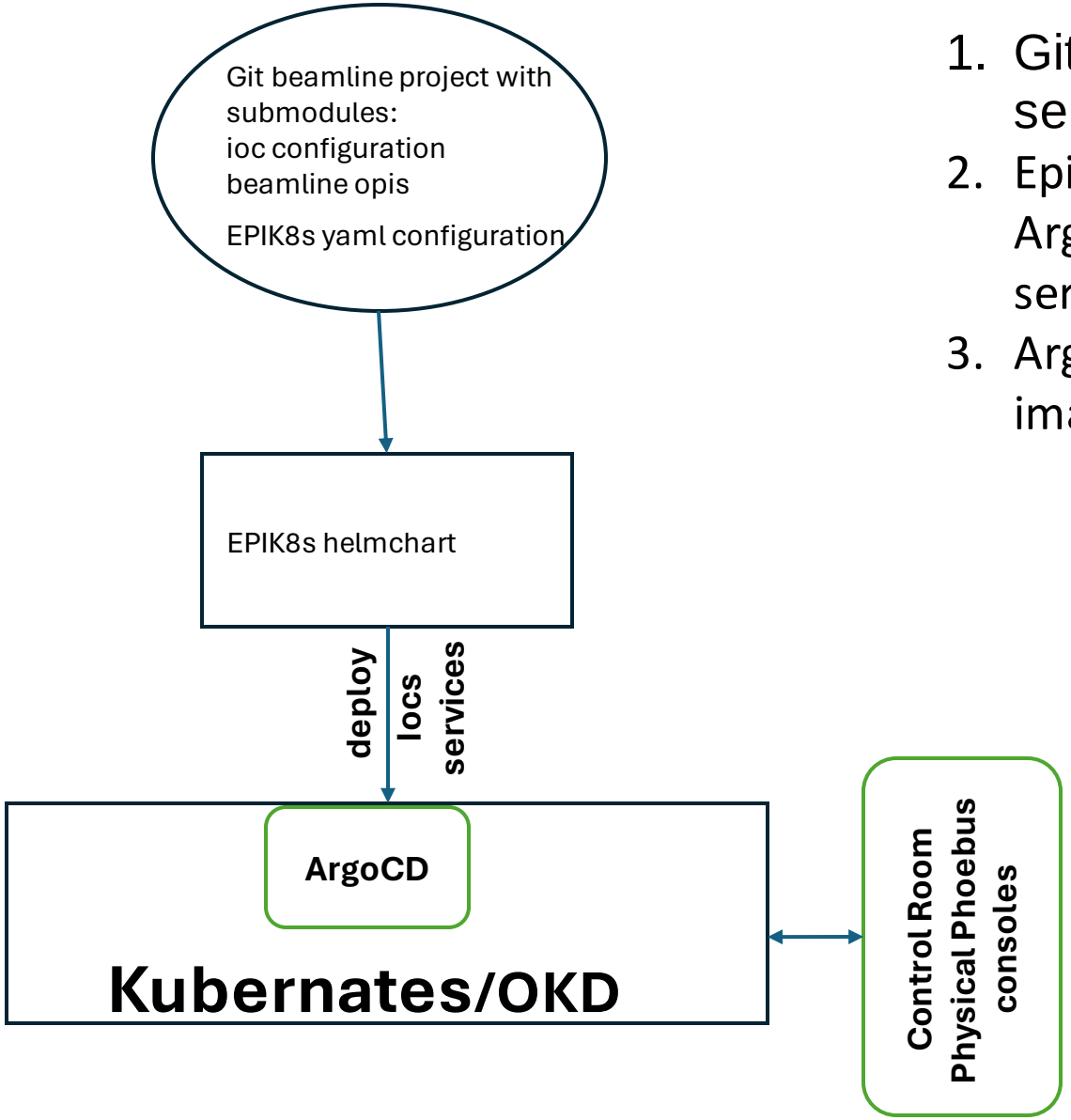
- ❑ Deploy versioned IOCs to the beamline
- ❑ Track historical IOC versions
- ❑ Rollback to a previous IOC version
- ❑ Monitor IOC status and versions
- ❑ View the current log
- ❑ Connect to an IOC and interact with its shell



Courtesy of Giles Knap (Diamond)

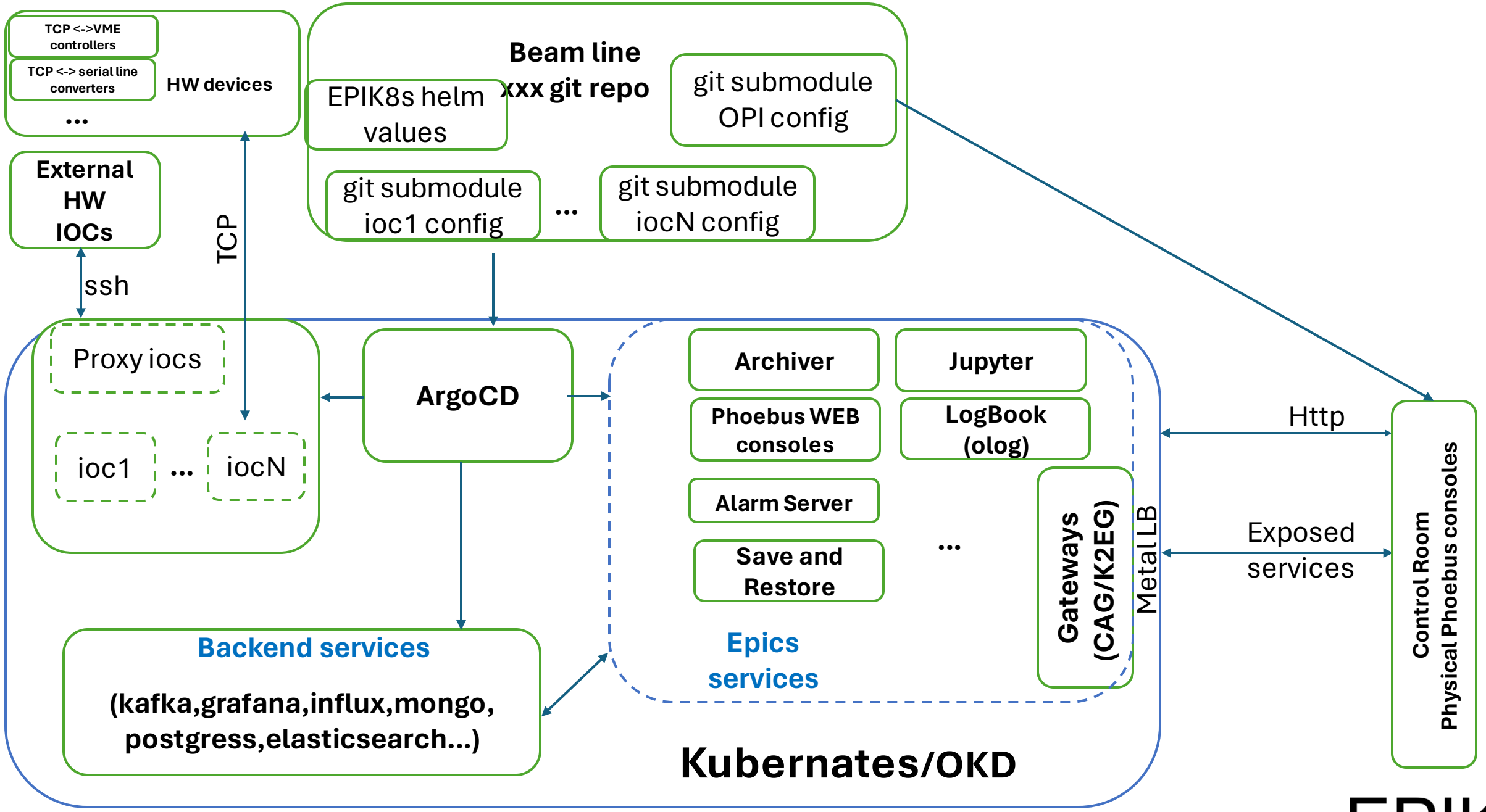
EPIK8S

Workflow in evaluation at LNF for EPICS



1. Git beamline project + settings of epics services
2. Epik helm chart apply settings and creates ArgoCD applications that manage services and services lifecycle
3. ArgoCD manage resources and keep aligned git image with running image

EPIK8S


TCP <->VME controllers
TCP <-> serial line converters
HW devices
...
External HW IOCs
ssh
TCP
Beam line xxx git repo
EPIK8s helm values
git submodule OPI config
git submodule ioc1 config
...
git submodule iocN config
Proxy iocs
ioc1
...
iocN
ArgoCD
Archiver
Jupyter
Phoebus WEB consoles
LogBook (olog)
Alarm Server
Save and Restore
...
Epics services
Gateways (CAG/K2EG)
Metal LB
Backend services
(kafka,grafana,influx,mongo, postgress,elasticsearch...)
Kubernates/OKD
Http
Exposed services
Control Room Physical Phoebus consoles

EPIK8S

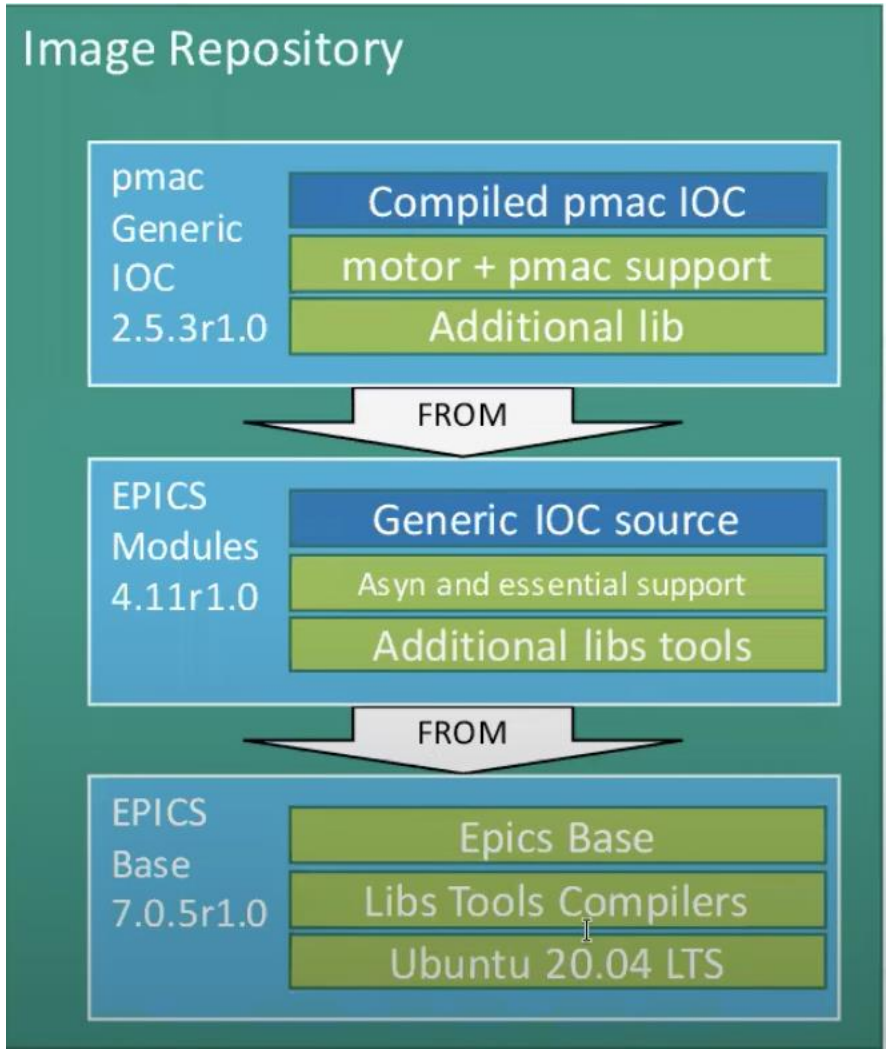
Considering merging Diamond workflow
https://github.com/epics-containers

In common:

have a yaml that generates the file to start an ioc through 'ibek'--> standardization
Containers are layered by specifying the needed support so that you can just get what you need
The yaml files can be collected to build an epics configuration asset of your beamline

Different (at the moment):

In our experimental workflow ArgoCD has a central role and it's used to start all ioc/services. Once is started in principle you could interface just argoCD and no more with K8s.
Diamond has an interesting set of python applications to handle ioc lifecycle.



Courtesy of Giles Knap (Diamond)

EPIKS

Why not describe your beamline in a single place?

1. It would be nice to have a library of device class support and a simple way to instantiate, deploy and configure relative IOCs. Share this library with the community.
2. It would be nice to have a tool to generate and configure OPIs and automatically generate simple operator control interfaces ready to use (with the option to use simulators for early prototyping)
3. It would be nice to have a way to have abstractions of device classes to simplify 2 and allow rapid prototyping of experimental algorithms and procedures

Tool	Library of IOCs	IOC OPIs	Control OPIs
Ibek https://github.com/epics-containers/ibek	Yes	+PVI YES	NO
Star tools https://baltig.infn.it/star-controls/star-controls-hw	Yes	Yes	Yes

In my opinion

Diamond tools are more structured and mature and go in the right direction.
Star (Francesco's tools) are good proof of concept of what would be nice to have.
Both use *yaml(s)* to keep the information.
Work together with Diamond to have a workflow to generate full featured prototypes for new beamlines from one or few yaml files. Yaml files then can feed a DB (and viceversa) for best exploitation.

EPICS