ATLAS image management

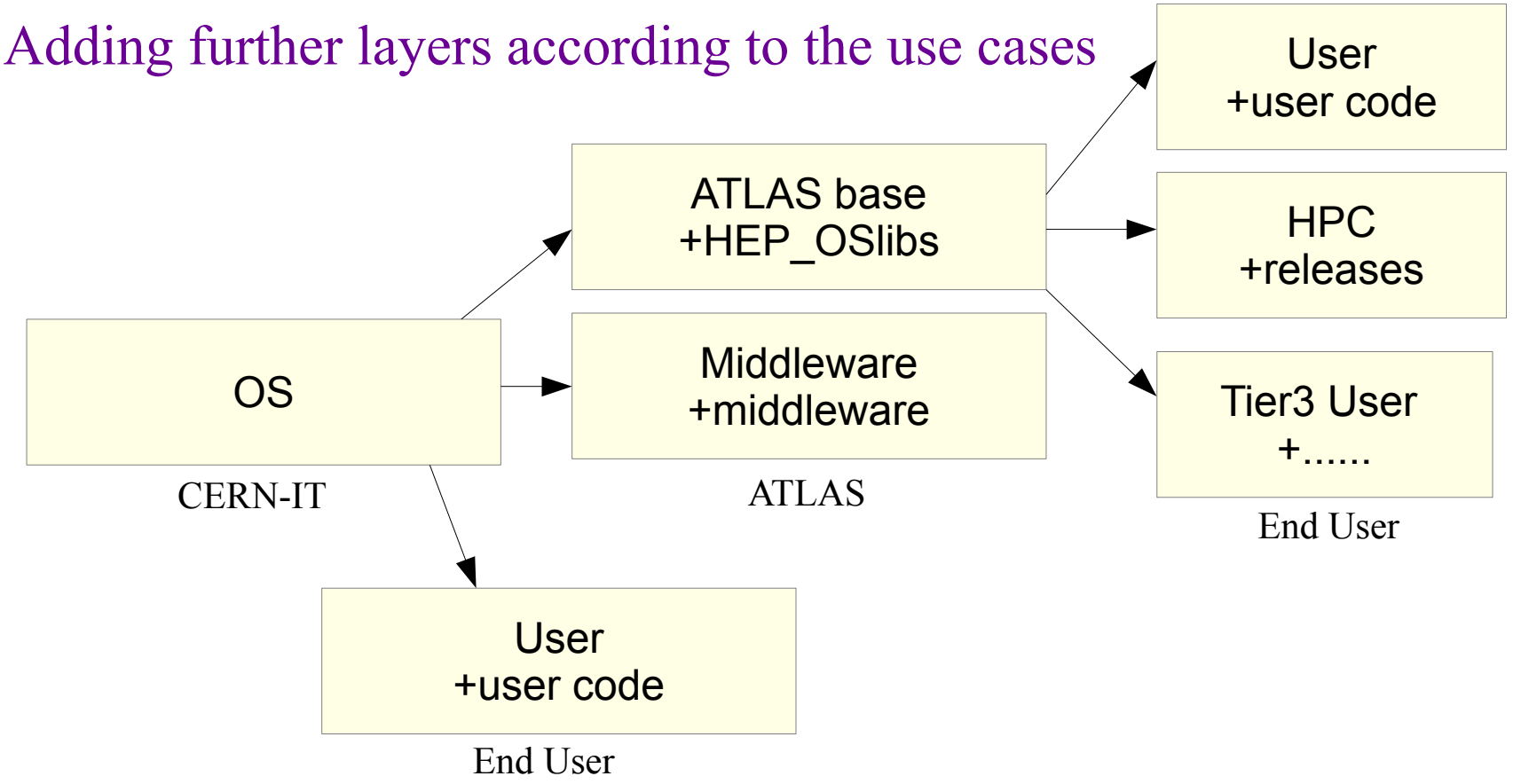
A. Forti

WLCG containers WG

17 October 2018

Images

- All ATLAS images are docker images
- They are built as a hierarchy of docker layers
- CERN-IT OS image is the root layer
  - Adding further layers according to the use cases



Use cases

- Images in CVMFS
  - Base images
  - Middleware images
- User images in docker
  - Analysis software
  - Other software
- HPC fat images
  - Usually built ATLAS base + prod releases
    - Custom + prod releases (Titan)

Images in CVMFS

- Base images OS + HEP_OSlibs
- All the experiment software is from CVMFS
- Imported from docker and unpacked in CVMFS
  - Two stages
    - Could be skipped if CVMFS did it by itself
  - Both img files and fs images
    - Img files to be dropped when infrastructure more developed
- Standard usage from CVMFS
  - Like CMS
- Only images that could be shared with other experiments

User docker images

- Different containers type support: docker, singularity, shifter,
  - OCI compliance will become more important
- Built on top of the base images (or the OS ones)
  - Analysis releases/User analysis
  - Else (ML images don't need ATLAS sw)
- Plan to use them on the grid with singularity
  - Currently reading from github (gitlab?)
    - singularity exec….. docker://userspace/useranalysis
  - Not scalable
    - All image is downloaded
    - Has overheads for each payload
    - singularity caching possible but cumbersome
- gitlab CI infrastructure uses the same registries w docker

HPC fat images

- Not all HPC centres have CVMFS
  - Images can be built from the registries
  - Currently built at BNL
    - Using ATLAS base image and adding manually the production releases from CVMFS to the image
      - They were looking at cvmfs_shrinkwrap as an official solution to do this
    - Then distributed and installed at HPC centres
  - An infrastructure of the type under discussion brings very little at the moment.
    - More useful would be to have the releases in the registry directly

Docker → CVMFS

- A solution that automatizes placing importing the images from docker in CVMFS is needed
  - Would eliminate the manual steps we are already doing which would become unmanageable if we added user images
  - Would make using images more scalable
  - Would make user images more manageable on the WNs
- CVMFS integrated solution preferred
  - Both for singularity file caching and docker graph-driver
  - Interested also in containerd ↔ CVMFS integration development
    - CVMFS as layer cache accessed via standard registries

Images source

- Having the images in CVMFS will move the validation of the images source from the job to the mechanism that pulls the images in CVMFS
  - ATLAS registry spaces the pilot is allowed to download from.
    - docker://atlas/* yes
    - docker://unknownuser/* no
    - gitlab.cern.ch://* yes
    - …….
  - Easy to maintain
  - Users using gitlab CI/CD could also advertise they want their image in CVMFS when they create one.
    - A yaml directive?

Docker → CVMFS

- A solution that automatizes placing importing the images from docker in CVMFS is needed
  - Would eliminate the manual steps we are already doing which would become unmanageable if we added user images
  - Would make using images more scalable
  - Would make user images more manageable on the WNs
- CVMFS integrated solution preferred
  - Both for singularity file caching and docker graph-driver
  - Interested also in containerd ↔ CVMFS integration development
    - CVMFS as layer cache accessed via standard registries

Central repository

- Highly desirable if it helps reducing the amount of data distributed and reduces strain on stratum0 and stratum1 and possibly sites squids
- OS images can be shared with other experiments
- Major sharing advantages within the experiment

Conclusions

- ATLAS container usage will start more heavily with production using images already in CVMFS at the end of the year
- User containers will follow more slowly
- In favour of
  - CVMFS integrated common solution
  - Common repository if this reduces the load on the infrastructure and is more naturally supported by the new features.
- Flexible on the timeline for the next year