NAF @ DESY

National **Analysis Facility** at DESY

Christoph Beyer, Thomas Hartmann, Yves Kemp, Christian Voß

https://naf.desy.de

National Analysis Facility at DESY

Result of Continued Deployment and Development

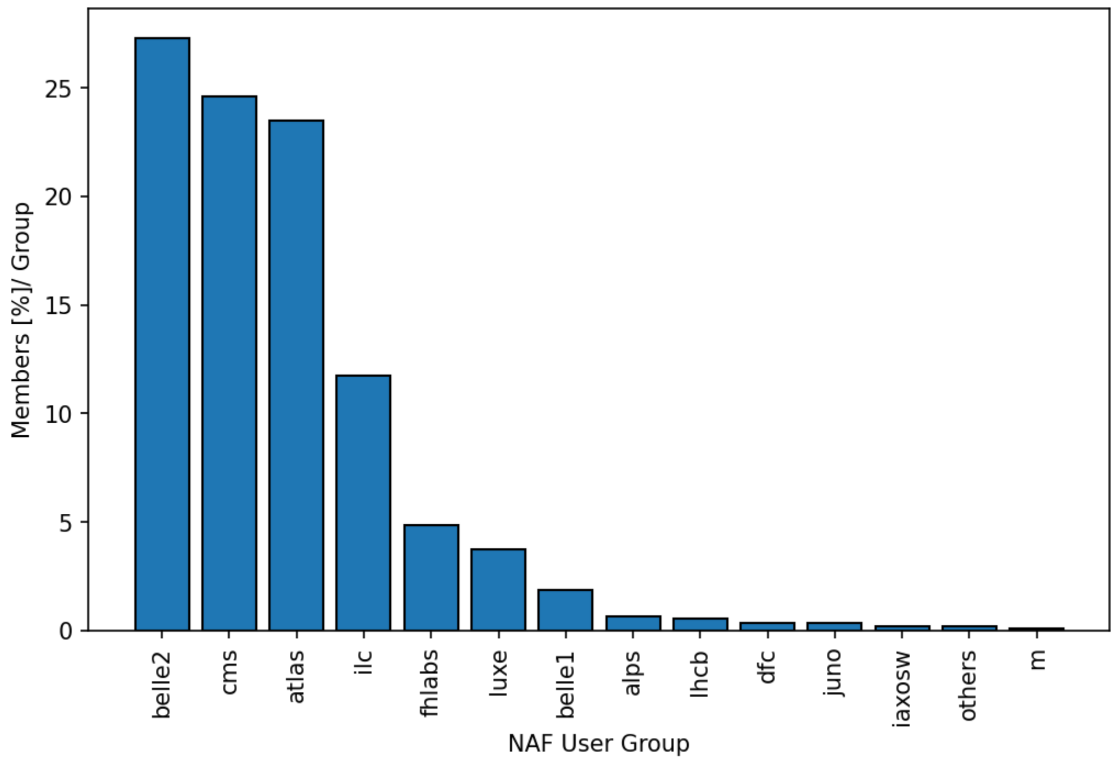
- 15 years user operations
  - NAF v1 started in 2007 in our two lab sites: Hamburg & Zeuthen (Berlin)
  - Reimplemented in 2013 as NAF 2.0 in Hamburg because
    - NAF v1 not integrated into central DESY IT
    - Data analysis over WAN was problematic
- Available to a broad user scope
  - Initial idea: for individual HEP users complementary to Grid workflows
    - faster job turn-around vs. high-latency grid jobs
  - Nowadays
    - Grid *primarily* production
    - NAF resources becoming even more relevant to individual users
    - Users beyond classic HEP experiments → Detector R&D, small scale Intensity Frontier Experiments

User Communities

Who is Working on the NAF

- DESY in general is a major site for the large HEP experiments
  - ATLAS, CMS, LHCb as Tier 2
  - Belle II (Raw Data Centre)
- Host large local groups for ATLAS, CMS, Belle 2
- Host ILC as Tier 0

- DESY hosts many smaller (on-site) communities
  - Detector/Accelerator R&D
  - On-site experiments: ALPS II, IAXO, MADMAX, LUXE ...
  - Theory groups, e.g. ILDG for Lattice QCD community
- DESY is a large Photon Science Laboratory
  - PS relies heavy on POSIX for data access
  - PS also data driven but using different tools
  - Look for synergies between the communities

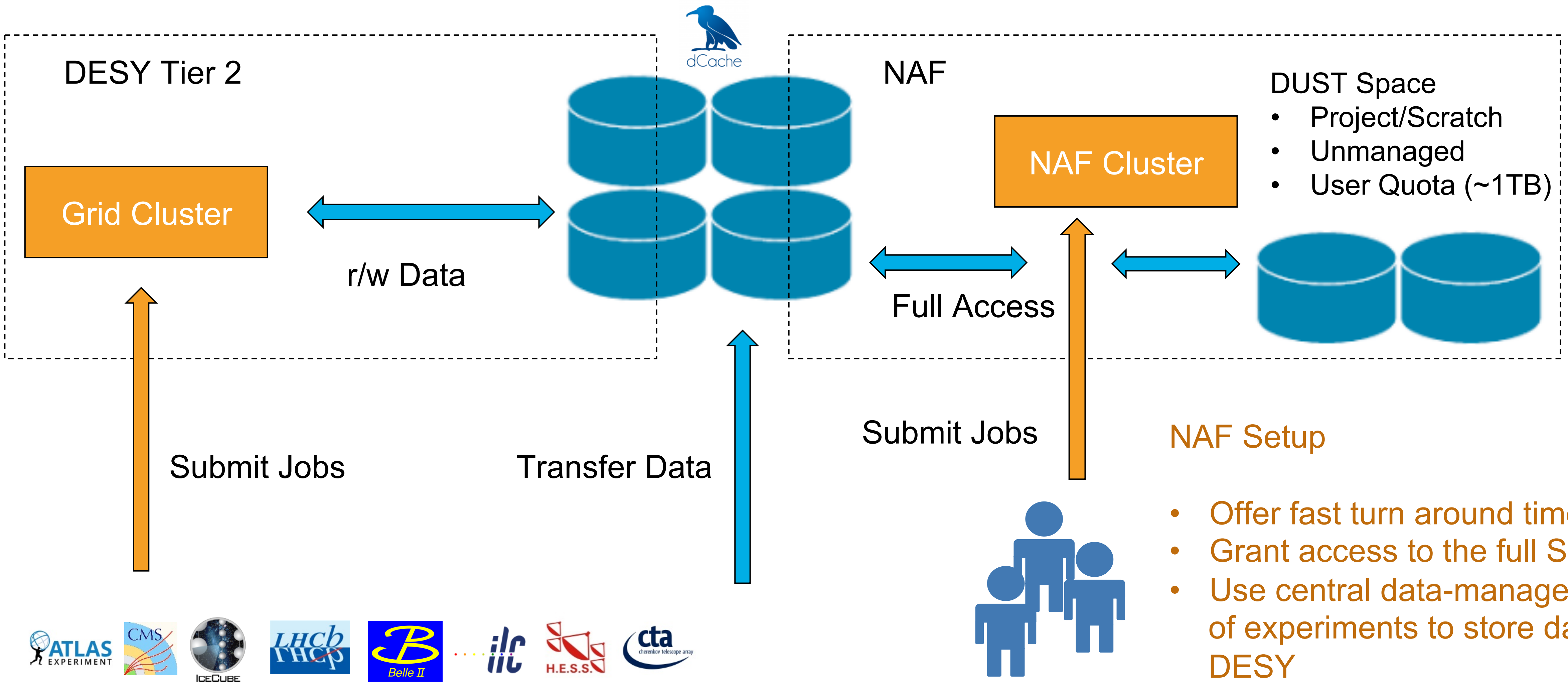


- Only the big four use full Grid workflows
- WLCG/HEP-only solution do not fit us
- E.g. many tools do not support XrootD

Paradigm: HEP Analyses are Data Driven

As Underlying Principle of the NAF

- Almost all HEP data analyses require access to large amounts of data



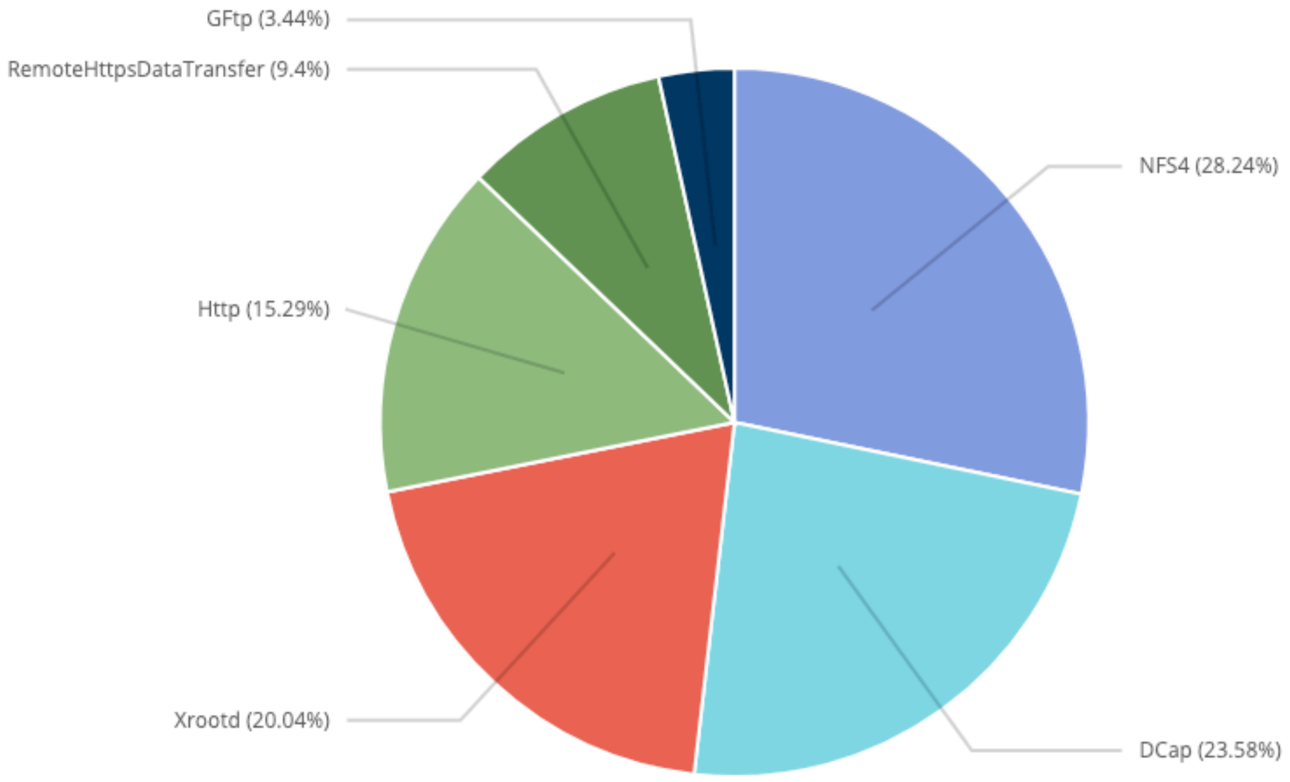
NAF Setup

- Offer fast turn around times
- Grant access to the full SEs
- Use central data-management of experiments to store data at DESY

Data Access

Protocol and Patterns

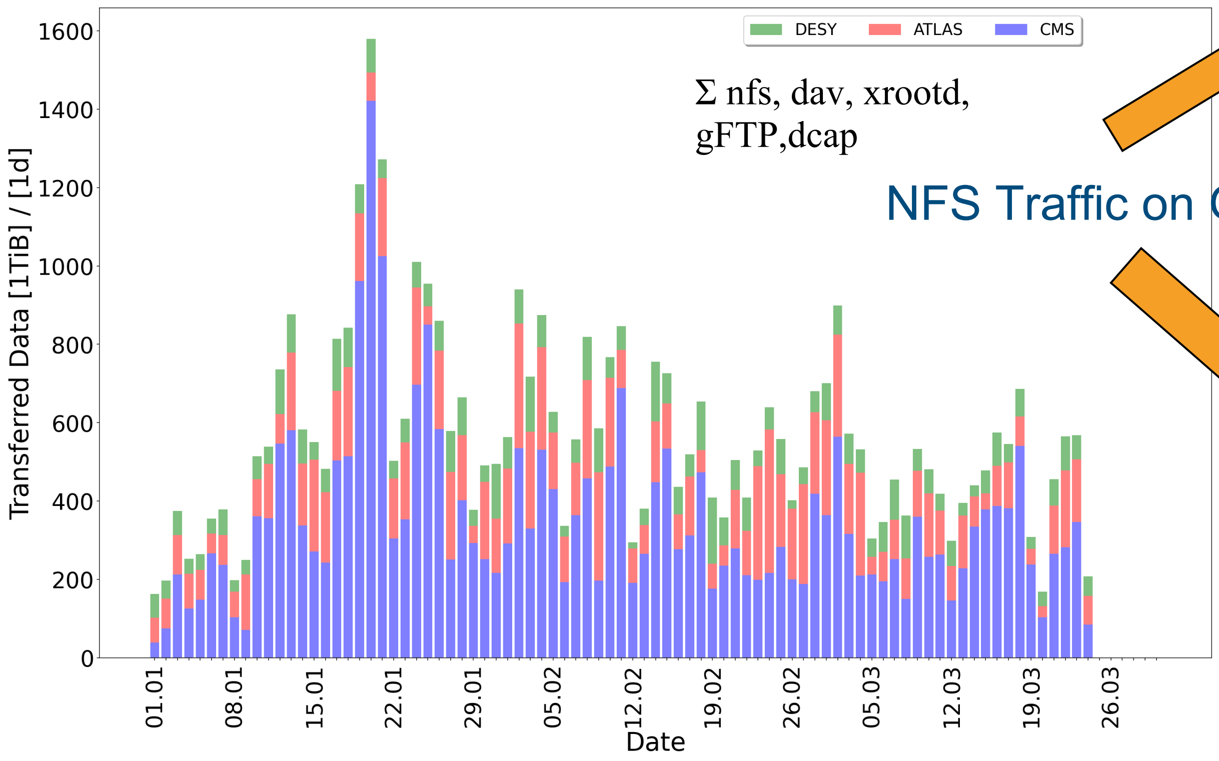
- Support all commonly available protocols for HEP
  - WebDav, XrootD & gsi-FTP, dCap as legacy
- Rely heavily on POSIX-like access
  - Storage infrastructure based on dCache and GPFS
  - Scalable NFS architecture across the NAF
  - Belle II group uses NFS almost to 100%
  - ATLAS users read (Rucio managed) LocalGroupDisk via NFS
  - Smaller groups use tools supporting only local files or industry formats like S3 or Hadoop
- Observed issues with NFS in the past, i.e. hanging clients (esp. SL6) → fixed with help of developers
- NFS makes testing of new tools very easy



Data Usage on NAF

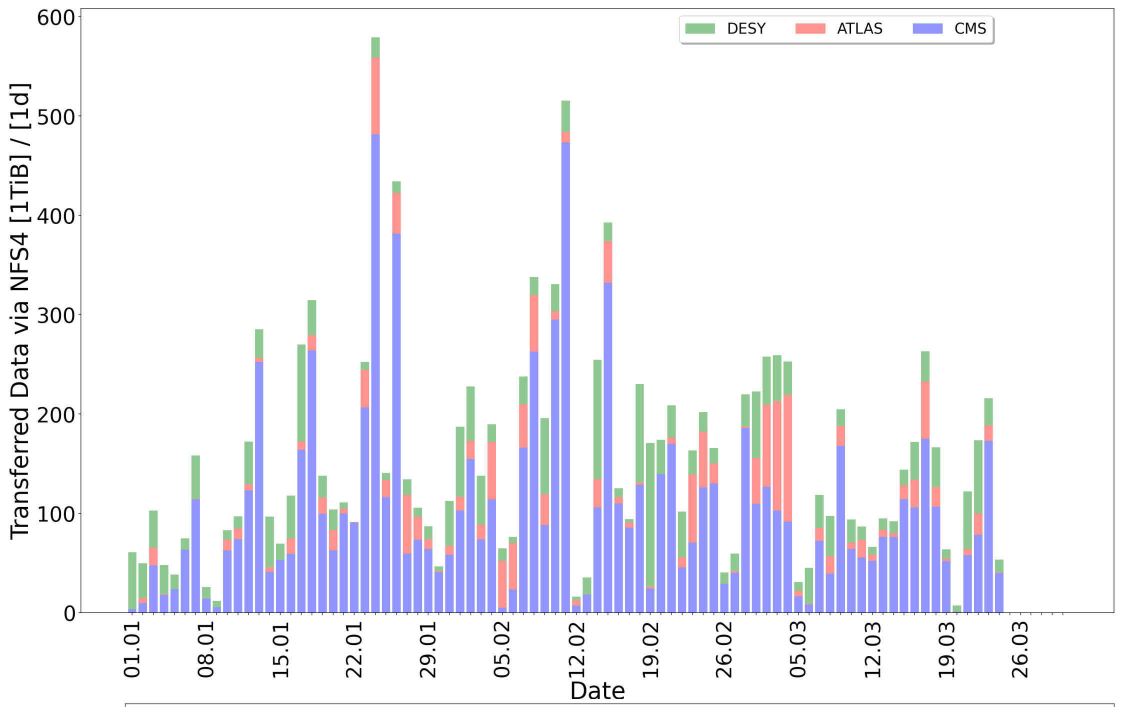
Focus on NFS Access

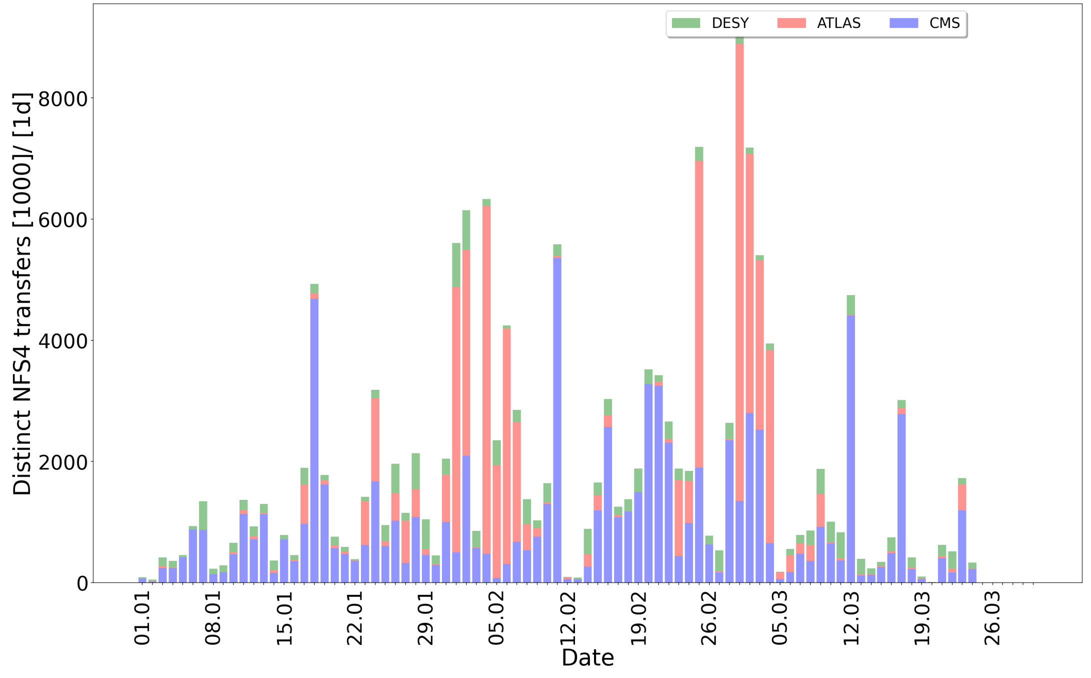
Whole Traffic on Grid SEs



Σ nfs, dav, xrootd, gFTP,dcap

NFS Traffic on Grid SEs



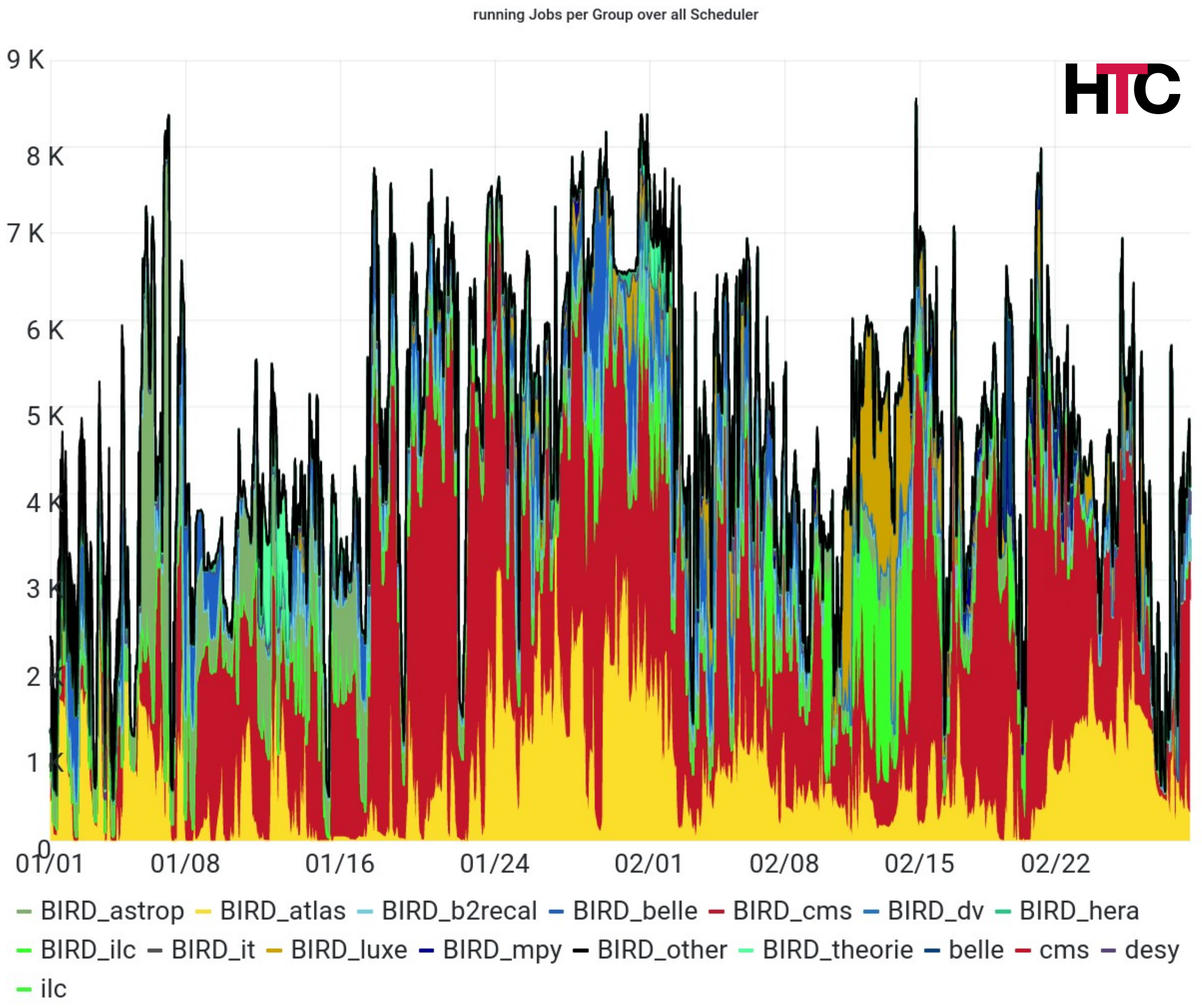


Whole Data transfers for 2022 till yesterday

Computing on NAF

HTCondor Clusters @ DESY Hamburg

- NAF & Grid clusters both run HTCondor
- Idea during migration to merge both clusters but
  - NAF jobs scatter too much w.r.t. Grid jobs
  - Both cluster well utilised limiting synergy
- Paradigm for NAF: complementary to Grid
  - Keep all worker nodes identical (OS, Condor, Access, ...)
  - fast turn arounds compared to Grid submission
  - Jobs have short runtimes compared to Grid
  - Offer more slots of shorter run time
  - Enough resources to keep the cluster below 75% usage
- User groups use NAF quite differently
  - CMS jobs more streamlined closer to Grid
  - ATLAS: individual users with bursts of many short jobs
  - Belle II with focus on working with Jupyter Notebooks

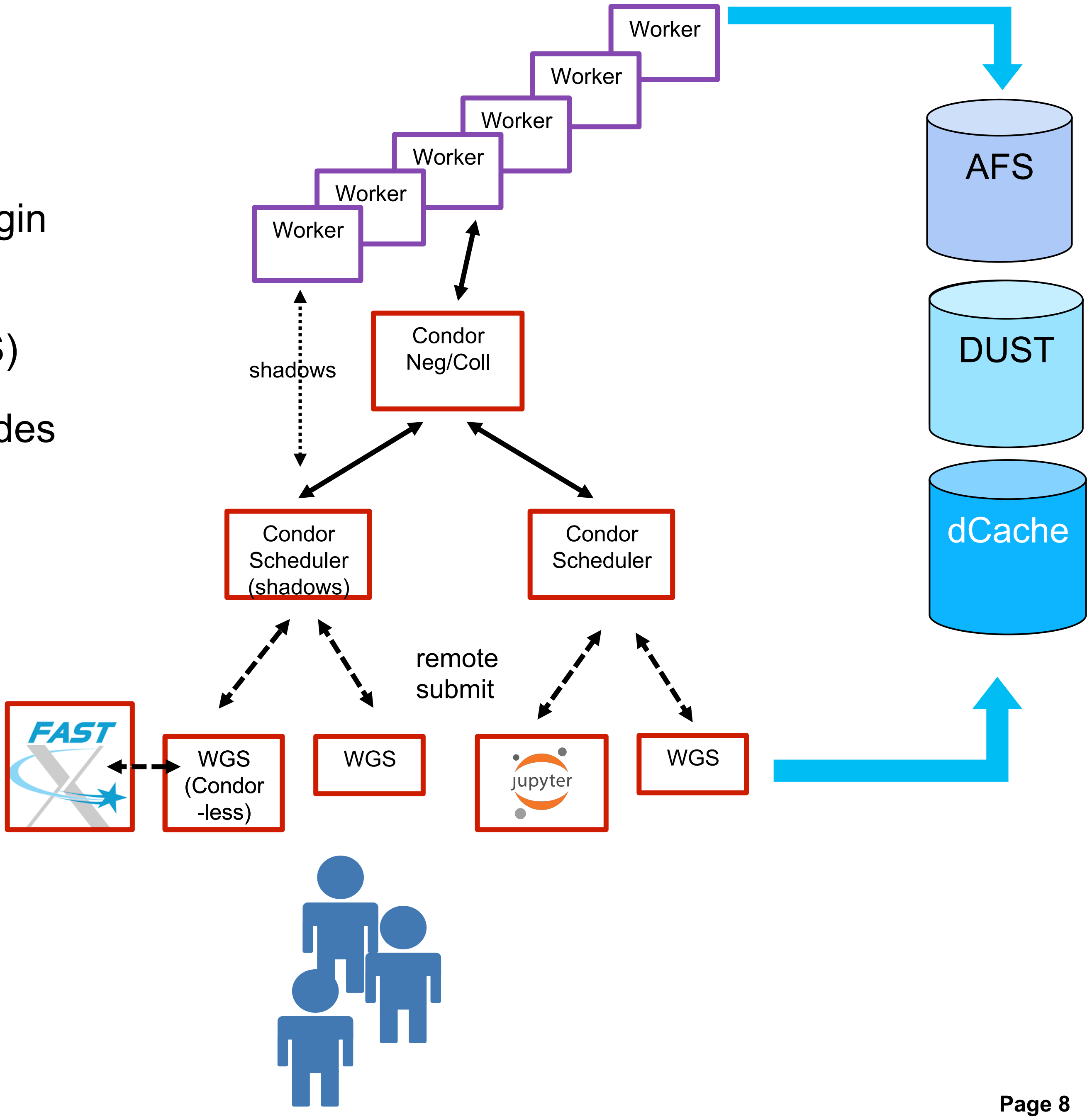


User Access Methods

SSH to Dedicated Login Nodes

- Worldwide access through SSH to dedicated Login Nodes (WGS)
- Per group dedicated Work-Group Servers (WGS)
- Used as interactive work and job submission nodes
- Full access to the storage of the group
- HW for larger groups, VMs for smaller

- Remote submission from WGS
  - Dedicated scheduler nodes
  - Easier draining & rotation

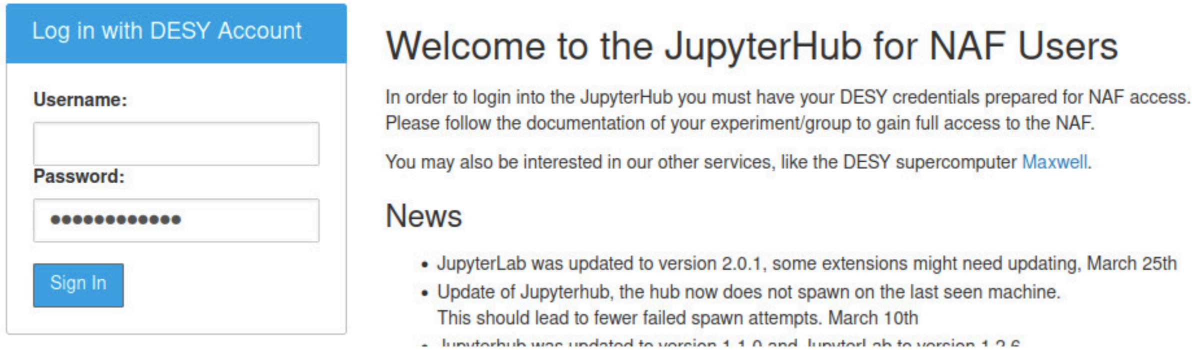


User Access Methods

Jupyter Notebooks through Jupyter-Hub

- Jupyterhub integration in the NAF
  - Spawn notebooks as Condor jobs
    - Limit run time to 24h to avoid abandoned notebooks
    - Standard job limits (1core/2GB)
  - Full storage access
- Upstart responsiveness even more significant
    - Added *fast lane* for Condor notebook jobs
    - Reserve slots exclusively for notebooks (1/node)
- Route notebooks to nodes with smaller loads
- Notebooks as alternative to ssh sessions
  - World-wide access
- Limited adaption so far (more from Photon Science)

Deutsches Elektronen-Synchrotron DESY
A Research Centre of the Helmholtz Association


Log in with DESY Account

Username:

Password:

●●●●●●●●●●●●●

Sign In

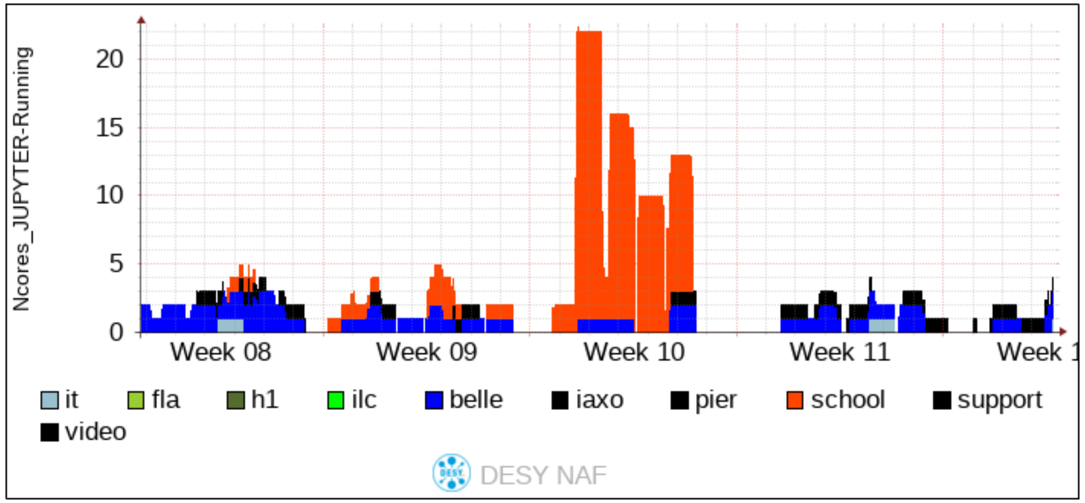
Welcome to the JupyterHub for NAF Users

In order to login into the JupyterHub you must have your DESY credentials prepared for NAF access. Please follow the documentation of your experiment/group to gain full access to the NAF.

You may also be interested in our other services, like the DESY supercomputer Maxwell.

News

- JupyterLab was updated to version 2.0.1, some extensions might need updating, March 25th
- Update of Jupyterhub, the hub now does not spawn on the last seen machine. This should lead to fewer failed spawn attempts. March 10th



Authentication & Authorization

Login and Access to Compute and Storage

- Access to the NAF requires a DESY account
- Use account as login
- Access based on POSIX users:groups
  - Storage authorization via AFS token
  - Classic file ownership
- Jobs running under user IDs
- AFS access requires forwarding of Kerberos tokens
  - Implemented token renewal for long running jobs

- Grid part support classic VOMS access
  - Grid Condor Cluster
  - dCache based Grid SEs
- Watch closely the token development in WLCG and elsewhere
- Prototype infrastructure supporting tokens
- Federated identities requires support as well in the future (opportunity for easier user provisioning)
- *long* running tokenised user jobs realistic?

- See users preferring classic name-space based file access
- Migration to e.g. tokens: probably not easy to implement it due to users’ learning curves

User Support and Governance

Interaction between Admins and Scientists

- NAF is not operated in a vacuum
- NAF Users Committee (NUC) to plan and decide on questions concerning NAF with members from
  - DESY IT
  - All supported experiments
- Discussions on
  - Size of additional procurements
  - Type of procurements
  - Shares between groups
  - Technology evolution

- Infrastructure support
  - Multi-level support within IT
  - Dedicated Queues in our Ticket system
- Experiment support
  - Organized by experiments
  - Mailing lists
- Good communication between experts from IT and experiments needed

Beyond Classic Batch Computing I

Apache Spark and Python Dask

Apache Spark

- Started working with Spark in 2018 for IT data
- Deployment on NAF in 2021 via Singularity
- Deploy master on WGS, worker as batch jobs
- Presented Spark to local groups
- Overall lukewarm reception

Python Dask

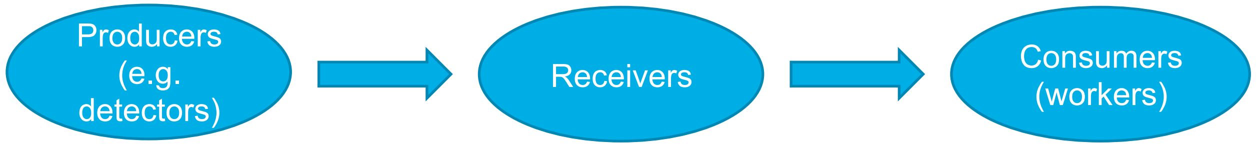
- Popped up from Belle II users when discussing Spark in 2019; Renewed Request in 2021
- Started to work with a pilot user
- Presented workflow to local Belle group
- Lukewarm reception

- Paradigm change: dynamic *memory* scheduling rather than CPU scheduling but with lower I/O load
- Use Apache Spark in production in IT on NAF and dedicated resources
- Unclear how to harmonize throughput vs. “*static”* memory workers
- Hogging Condor resources for sporadic in-memory operations?
- Occasional request from some users → but rarely with some follow up requests

Beyond Classic Batch Computing II

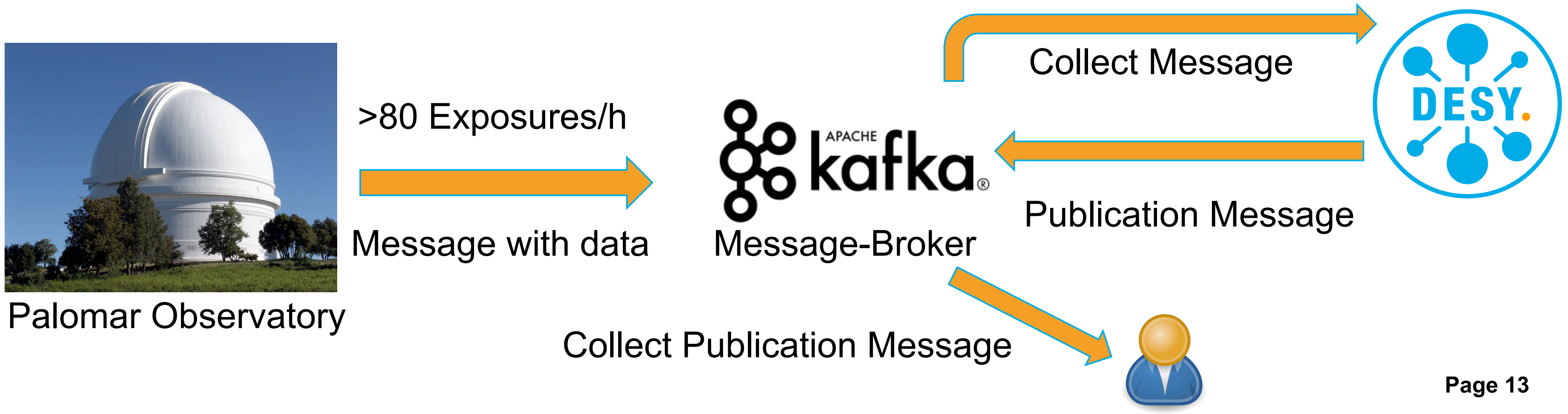
Event Based Workflows Using Function as a Service (FaaS)

- **Message Producer – Broker – Consumer Model**



- Most HEP workflows are user triggered/managed
  - Request data → analyse → plot → fit
  - Usually everything handled/checked manually
- Use events to trigger each step automatically
  - Files written trigger storage events → Storage Event trigger jobs → Job Events trigger fits → Fits notify user
- Main showstopper: Authentication/Authorisation → hope for improvement with emergence of tokens

- Example AMPEL at DESY Zeuthen



Beyond Classic Batch Computing III

GitLab Pipelines for Containers and Jobs & flat bucket event addressing

- GitLab scale out
  - Runners as Condor Jobs
  - Used to compile code/build containers
- GitLab as another interface to Condor submission
  - User payloads as pipeline into jobs
  - Job description in GitLab → build and test
  - Deploy jobs via runners as Condor jobs
- User inertia vs. GitLab workflows...?

- Grid: moving event addressing workflows to AFs?
- Event addressing ~ flat
  - Columnar analyses
  - Higher object store like interfaces rather than POSIX
  - How to harmonize with POSIX-like storage beneath? I/O patterns?
  - Authorization-only future with tokens?
    - At least on the upper levels
  - Broad user adaption?

Summary

Status of the NAF@DESY in 2022

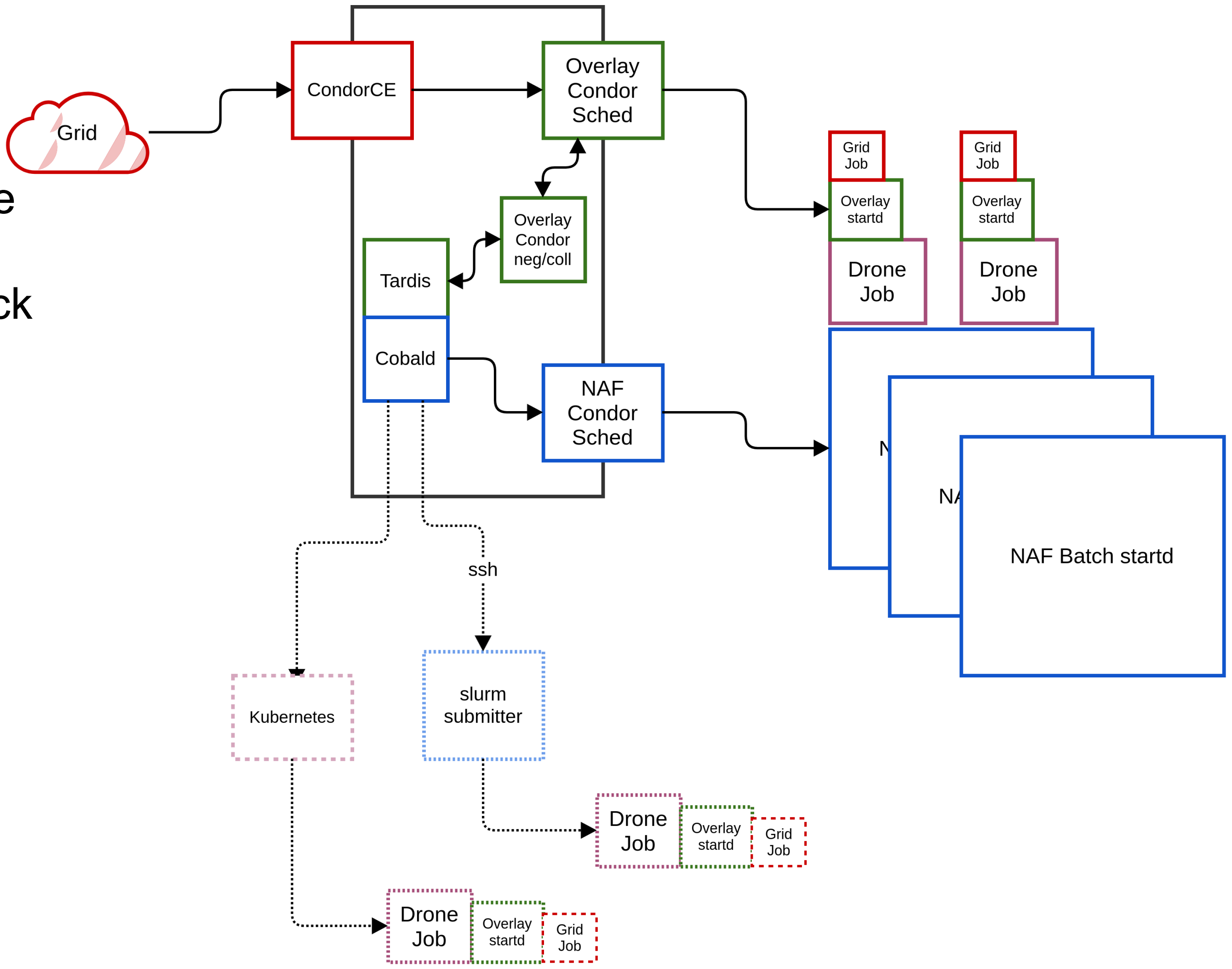
- NAF has been evolving over 15 years
- User centric serving a wide range of experiences
- Storage/data handling developing questions
  - Ownership & auth*
  - Effects of high level addressing on storage infrastructure beneath
- Trading latency for utilization?
- User adaption?
  - fat tail of users with standard tools
  - specialized users/analyses depending on specialized tools?

Appendix

Backfilling?

Opportunistically utilizing compute resource

- back filled NAF & Maxwell each with a dedicated ARC CE
- overlay LRMS over all potential resources would be nice
- aggregation NAF, Maxwell/HPC (Slurm), OpenStack + K8s in a single view
- would require *true* pre-emtable job
    - on-demand freeing Maxwell, NAF,... resources in < O(m)



Grid Backfilling

Opportunistic Grid jobs on the NAF + x?

- backfilled NAF GPU nodes with COVID jobs
  - ad hoc dedicated ARC CE
    - deprecated ARC CE while migrating Grid cluster to CondorCE
  - (similar for Maxwell HPC backfilling)

- consolidated backfill/overflow?
  - investigating KIT’s Cobald/Tardis
  - how to harmonise ID namespaces?
    - e.g., historically grown pool accounts in the Grid vs. local users