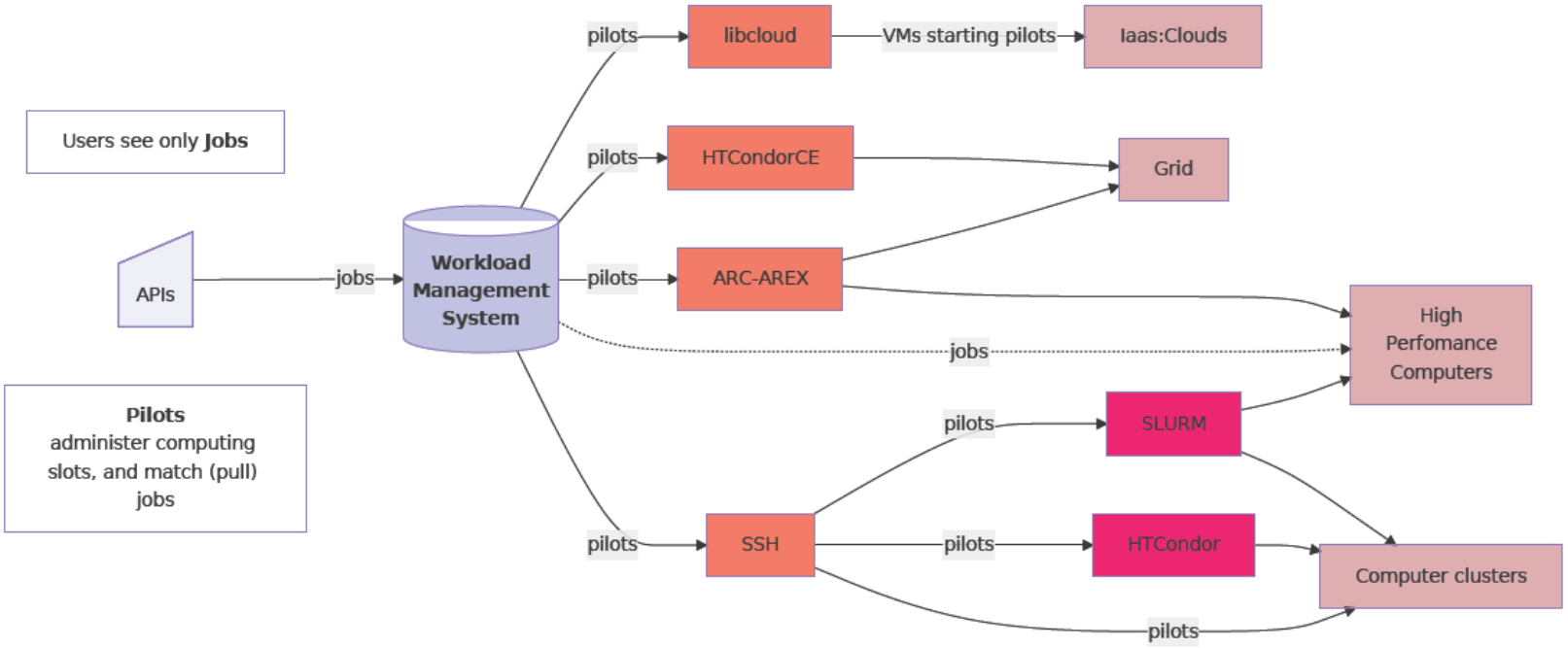
WMS supporting HPC

A.Tsaregorodtsev,
CPPM-IN2P3-CNRS, Marseille,
TGCC LCG-FR NUMPEX discussion
8 July 2026

In a nutshell

- DIRAC is a complete grid solution for one or multiple user communities that need to exploit distributed heterogeneous resources
- Both computing and storage resources can be handled within the same framework with support for large-scale operations

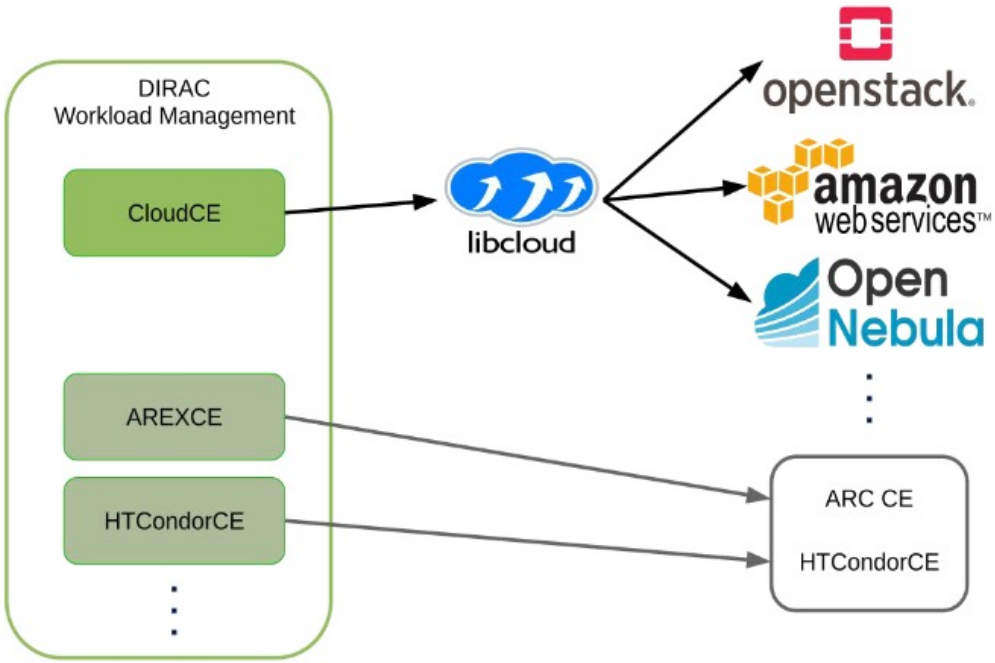
Managing workloads



- « Pull » model with pilot jobs (HTC, Cloud reources)
- « Push » solution for HPC centers that do not allow pilots due to the internet access limitations
- Integrating CWL (Common Workflow Language) for job descriptions in DiracX

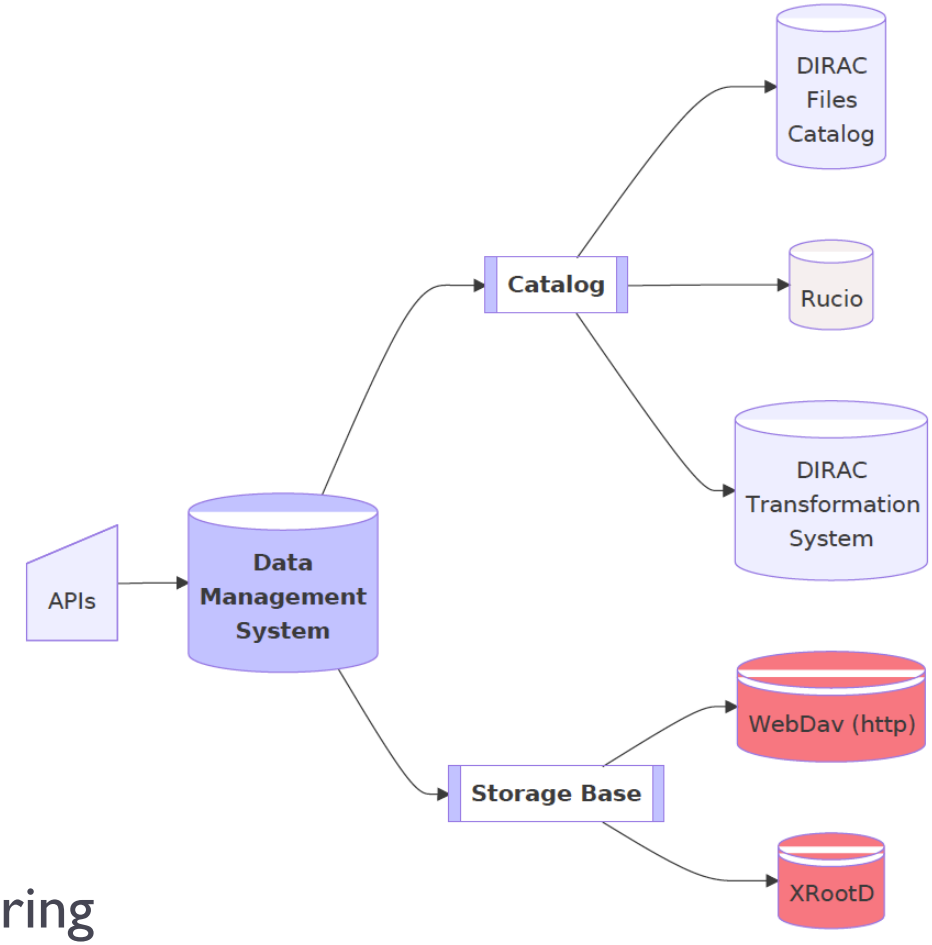
Computing Elements

- ComputingElement interface to abstract access to computing resources
- Grid CEs:
  - AREXCE, HTCondorCE
- CloudCE
  - The pilot payload script and data are added as instance metadata in **cloud-init** format
  - Any VM image containing **cloud-init** to decode and start the DIRAC pilot bootstrap scripts.



Managing data

- Managing files
  - Logical file names - LFNs
  - File Catalogs
    - physical replica locations
    - file metadata
    - DIRAC File Catalog
    - Rucio
  - Storage systems (SEs) with various access protocols
- Support for massive data operations
  - FTS, failure recovery, monitoring

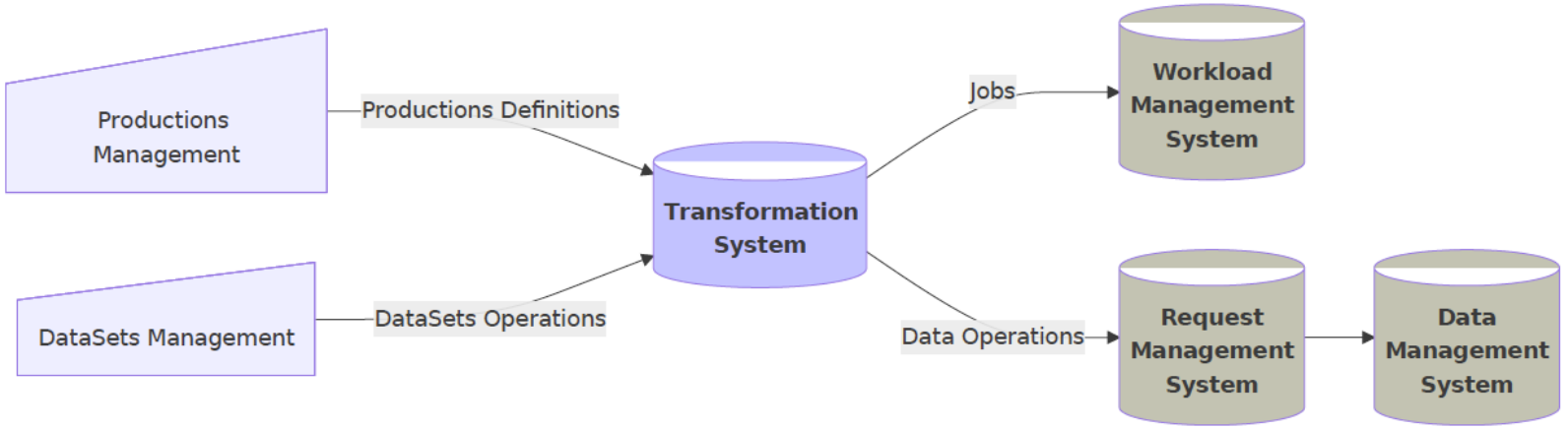


all in 1 solution

- DIRAC is a complete solution including both **workload** and **data** management subsystems
  - Allows to stay in the same software framework for all the tasks of a large experiment
  - Requires a single team of developers and administrators of a distributed computing project
    - Sharing professional experience for different tasks
    - Optimal for managing manpower of the project
- Possibility to use Rucio DMS along with the DIRAC WMS
  - RucioFileCatalog developed by the Belle II experiment
  - Popular solution explored by several communities, e.g. CTAO

- Easily customizable to specific needs of particular communities with the Extensions mechanism

Transformations

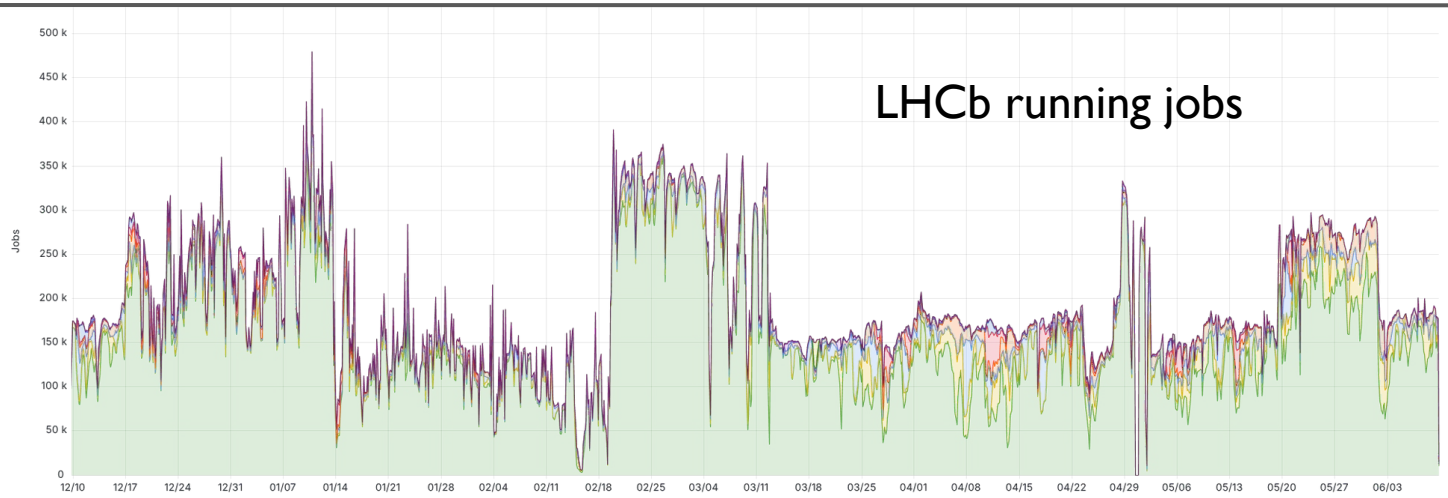


- Transformations System is an engine for managing complex **data-driven** workflows
  - Automation of common tasks
    - Creation, resubmission, deletion of jobs
    - Replication, registration, removal of data files
  - Handling millions of jobs and files
    - Massive operations validation, failure recovery

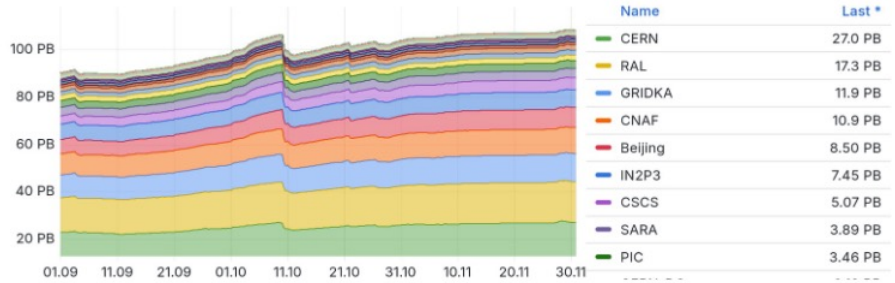
Open source project

- Driven by the needs of the user communities
  - With multiple contributions from community developers
  - Example: DIRAC File Catalog (DFC) was developed initially for ILC and BES III experiments, it is used now by several experiments including LHCb.
- Public code repository in Github
  - GPL v3 license
  - Automated testing, CI with Github actions
- DIRAC Consortium created in 2017
  - Developing and promoting the DIRAC software
  - Holder of the DIRAC software copyright
  - Current members: CNRS, CERN, IHEP, KEK, Imperial College

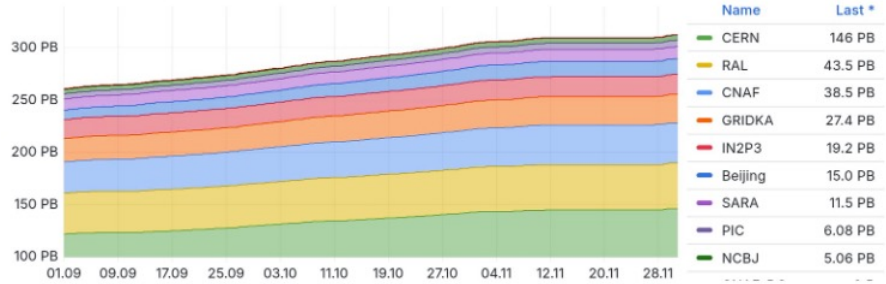
Users: LHCb



LHCb Data



Disk



Tape

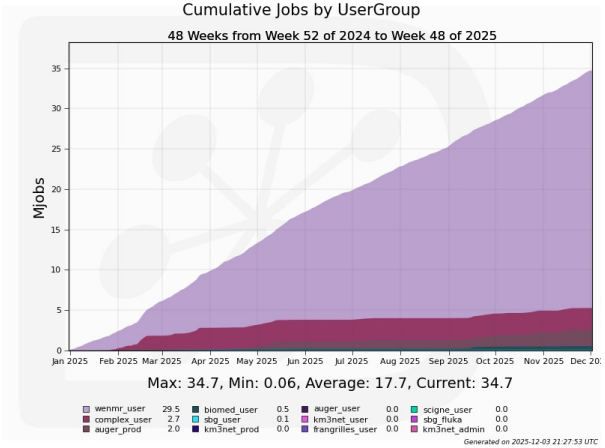
- Running over 500K concurrent jobs on more than 100 sites
- Resources: HTC, HPC, clusters, HLT farm

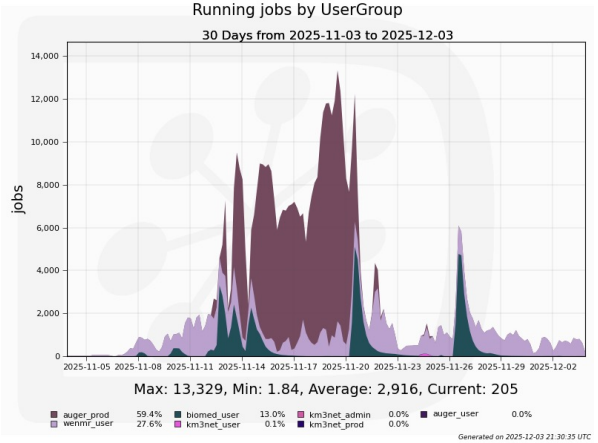
Users

- **Belle II, KEK**
  - Using DIRAC + Rucio combined service for all the production tasks. Developed RucioFileCatalog
- **CTAO** (Cherenkov Telescope Array Observatory)
  - MC production. Developed DIRAC Production System
- **ILC/CLIC, CERN**
  - Future accelerator detector modelling, WMS+DMS Developed service supervision tools
- **EGI Workload Manager**
  - Multiple VOs: WeNMR, biomed, Pierre Auger, KM3NeT,…
- **GridPP DIRAC service**
  - Multiple VOs: T2K, NA62, Euclid, …
  - Developed CloudComputingElement, Multi-VO Catalog

EGI Workload Manager

- Hosted at CC/IN2P3 in Lyon
  - Maintained by a joint team from several CNRS laboratories
  - Cluster of 10+3 medium sized virtual servers + MySQL/ElasticSearch host service
- ~25 VOs
- > 200 registered users
  - Some of them represent large communities
- Running smoothly
  - ~35M jobs in 2025
    - Up to 6K concurrent jobs
    - Up to 200K jobs per day in spikes
- Other multi-community services
  - IN2P3, GridPP, IHEP, etc





Ongoing developments in full swing:

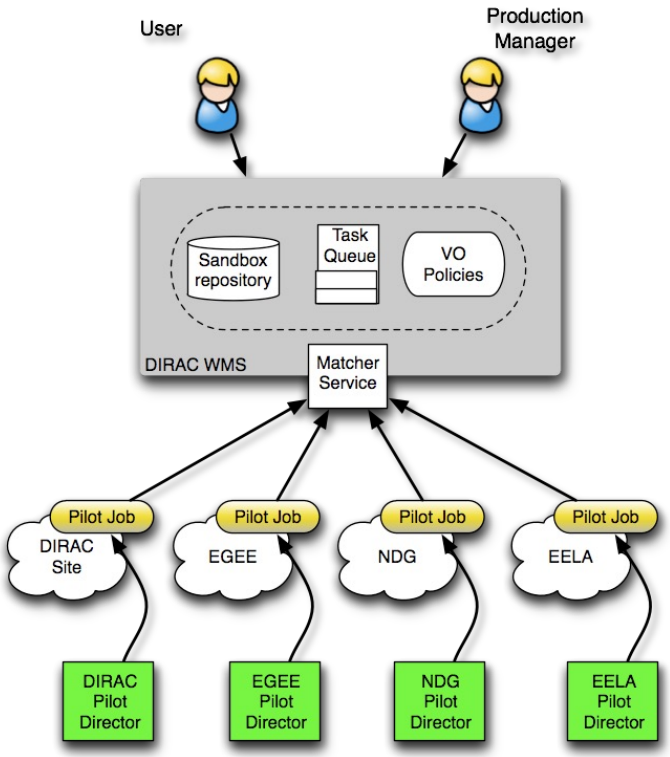
- **DIRAC** software technologies becoming outdated
  - Custom client/service protocol, RPC style, custom client APIs
  - Complex deployment, no tokens,
  - Difficult to onboard new developers
- **DiracX** - complete rewrite of the **DIRAC** software stack while keeping the base architecture and functionalities
- Based in well-established technology standards
  - AAI: OIDC/OAuth2 native
  - REST interfaces :
    - FastAPI, Swagger docs, OpenAPI compliant
  - DiracX web :
    - NextJS, Material UI, Typescript
  - Deployment
    - Kubernetes, Helm charts
    - Also used for continuous integration test
- The WMS DIRAC subsystem is based on the CWL framework to manage complex workflows

DIRAC
THE INTERWARE

More on WMS and HPC

WMS Architecture

- DIRAC WMS architecture is based on the ***PULL*** paradigm and the concept of pilot jobs
- User jobs are submitted to the central Task queue
  - Global view of user tasks allows to apply efficiently scheduling policies
- Pilots are submitted to various computing resources :
  - With specific submission mechanism in each case
- Pilots match (pull) the user payloads and steer their execution
  - Late binding of user payloads to the computing resources
  - Ensure precision of policies application
  - Reduce payload failure rates



Job submission

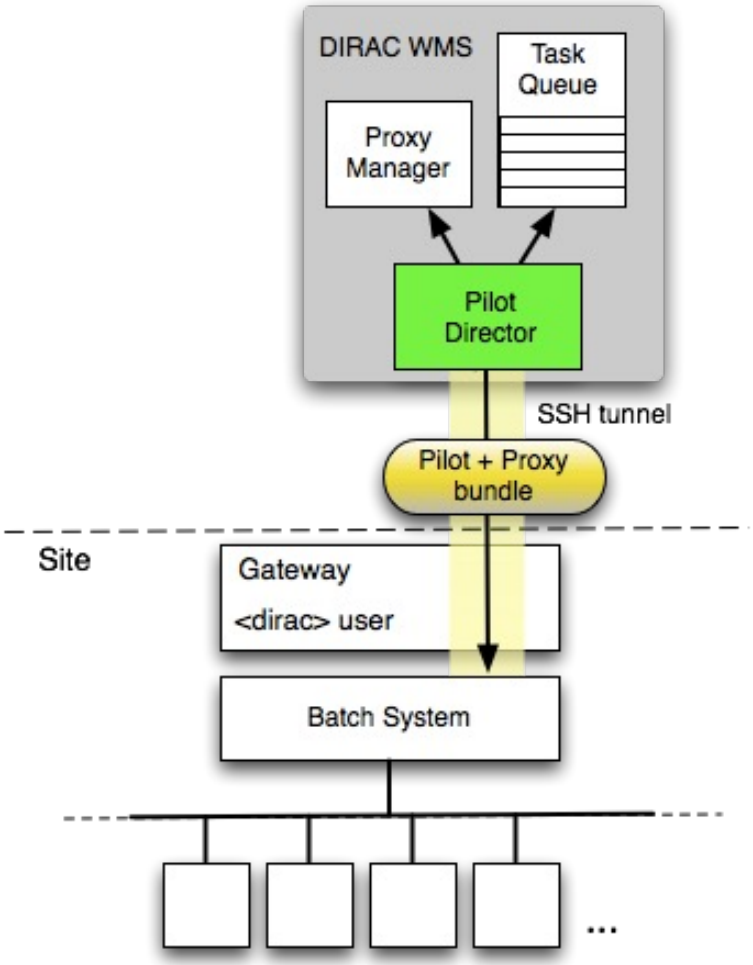
- Users submit jobs with precise description
  - Applications
  - Input/output data
  - Resources requirements
- Jobs requirements are analyzed and suitable sites are preselected
  - E.g. sites close to the input data storage service
- Bringing data online from archive storage can be triggered while the job is waiting in the Task Queue

Computing Grids

- DIRAC was initially developed with the focus on accessing conventional Grid computing resources
  - WLCG grid resources for the LHCb Collaboration
- It fully supports different middleware based grids
  - European Grid Infrastructure (EGI), WLCG, etc
    - DIRAC is an officially supported WMS service for the EGI infrastructure
  - Northern American Open Science Grid (OSG)
    - HTCondor middleware
  - Northern European Grid (NDGF)
    - Using ARC middleware
      - Direct submission to computing elements
- Other types of grids can be supported
  - As long we have customers needing that

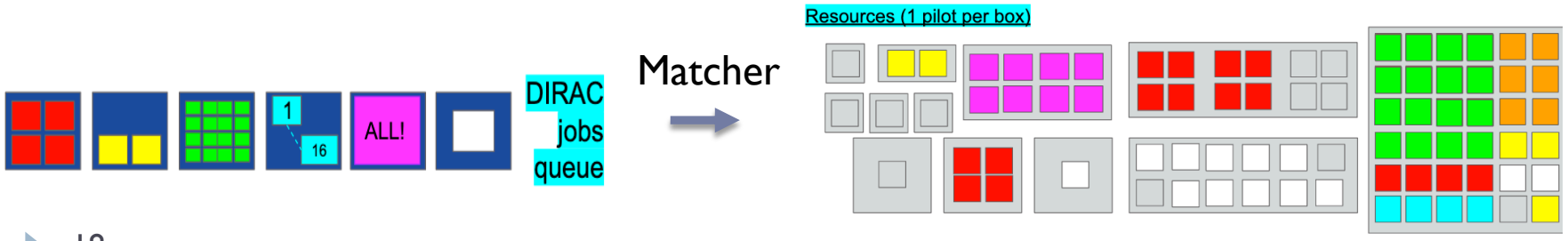
Standalone computing clusters

- Users can connect their own computing resources
  - Not making part of any grid infrastructure
- The user site can be:
  - a single computer or several computers without any batch system
  - a computing cluster with a batch system
    - LSF, BQS, SGE, PBS/Torque, HTCondor
      - Commodity computer farms
    - SLURM
      - HPC centers



Multi-core workers

- Pilots exploit multi-core workers using **PoolCE** "inner" Computing Element
  - On-WN batch system running jobs in isolated containers
  - Flexible strategy with prioritized job requests to the Matcher, e.g.:
    - First, ask for jobs requiring WholeNode tag
    - If none, ask for jobs requesting as many cores as available
    - If none, ask for jobs with MultiProcessor requirement
    - If none, ask for single-core jobs
  - The goal is to fill the nodes with payloads fully exploiting there multi-core capacity



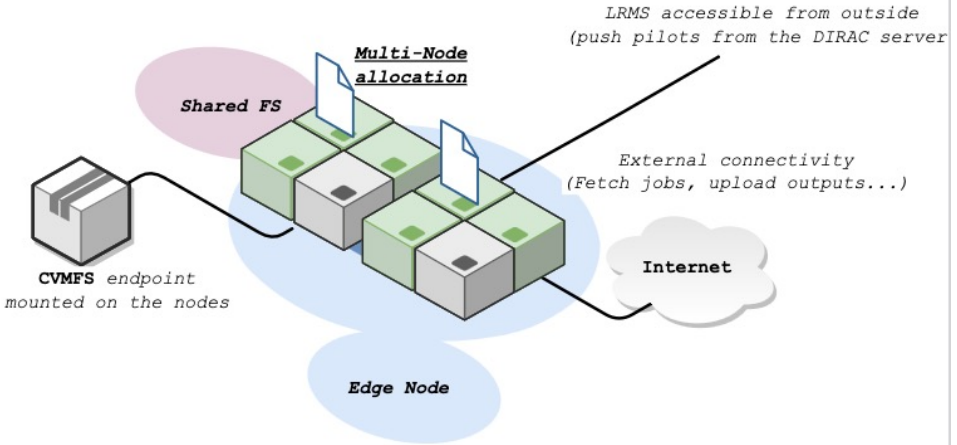
Muti-node allocations

- Fully exploit multi-node allocations
  - Natural for MPI-type jobs
  - Needs effort for grid-like "few-cores" jobs

Sub-Pilots

v7r3 DIRAC

- Use of `srun` to install 1 pilot-job per node in parallel.
- The pilot-jobs share the same identifier, status and logs.
- Possibilities to request elastic allocations (e.g. between 1 and 5 nodes).



Job Wrapper

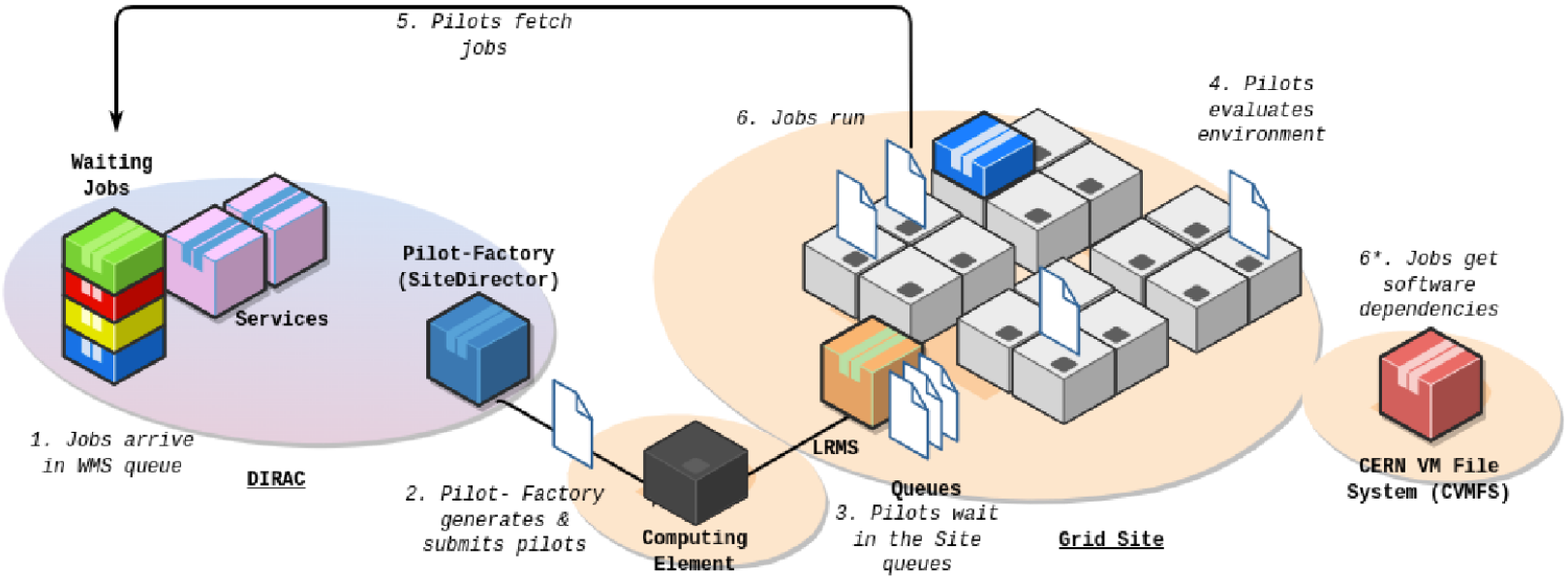
- Pilots create job wrappers to steer user jobs execution in the worker nodes
- Job wrapper – a watchdog process running in parallel with the user application
  - Initializes user application
    - Downloading input data, downloading software (CVMFS cache)
      I/O bound phase
  - Watches resources consumption by application
    - CPU, Disk, Memory, I/O operations
    - CPU bound phase
  - Sends job heartbeats and receives WMS commands
    - E.g. killing stalled jobs
  - Uploading results
    - Output data
    - Accumulated job parameters (resources usage, efficiency, etc)
    - I/O bound phase
- Job parameters as measured by the job wrapper are available to users
  - Allow for better description of subsequent jobs
- Job wrapper is part of the WMS
  - Can interact with the execution environment, e.g. CPU throttling while in I/O phase

HPC access

- DIRAC requirements to HPC
  - Standard user-level access to submit jobs to the local LRMS
    - E.g. ssh login with public key to the LRMS gateway
    - Or a standard CE interface, e.g. ARC at friendly sites
    - If no standard access possible:
      run pilot factory in the HPC gateway
  - Two main modes of operation
    - "Pull" mode similar to the grid environment
      - Rarely the HPC case
    - "Push" mode if no outbound connectivity is possible
  - Or a mixture of the two

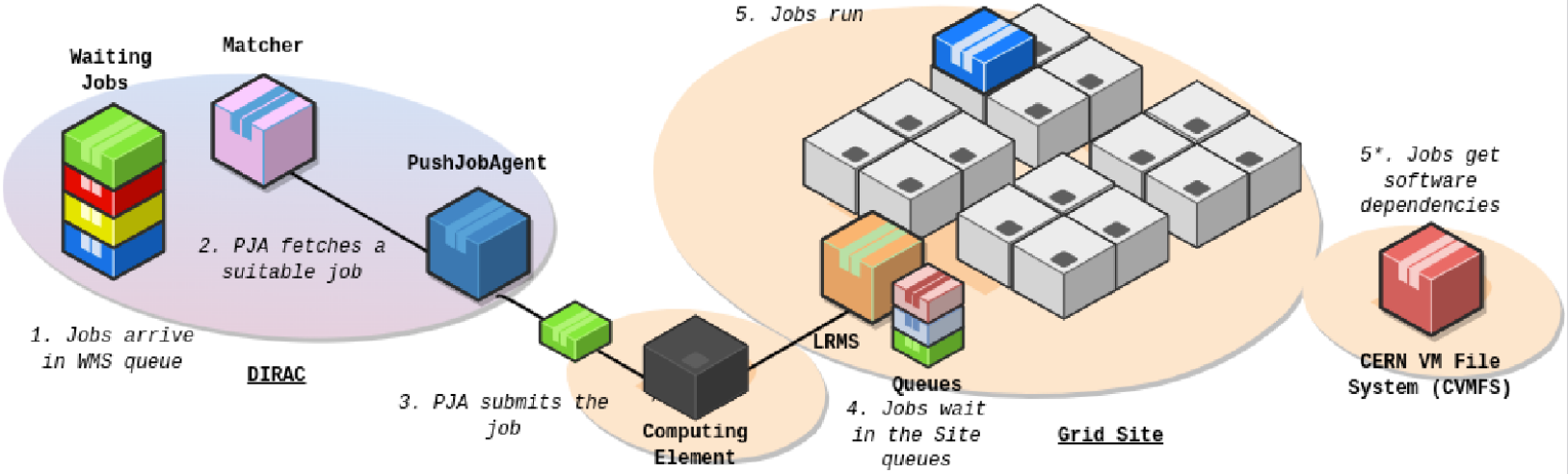
Resource allocation in “pull” mode

- Pilot mode “pull” (preferrable a la grid)
- LRMS jobs are standard DIRAC pilot jobs
- Preinstalled DIRAC software: preferably in a global FS, i.e. CVMFS
- Outbound connectivity to the WAN – central DIRAC services, storage services, CVMFS
- Job’s input sandboxes and data are brought in by the pilots
- Job’s output sandboxes and data are uploaded by the pilots



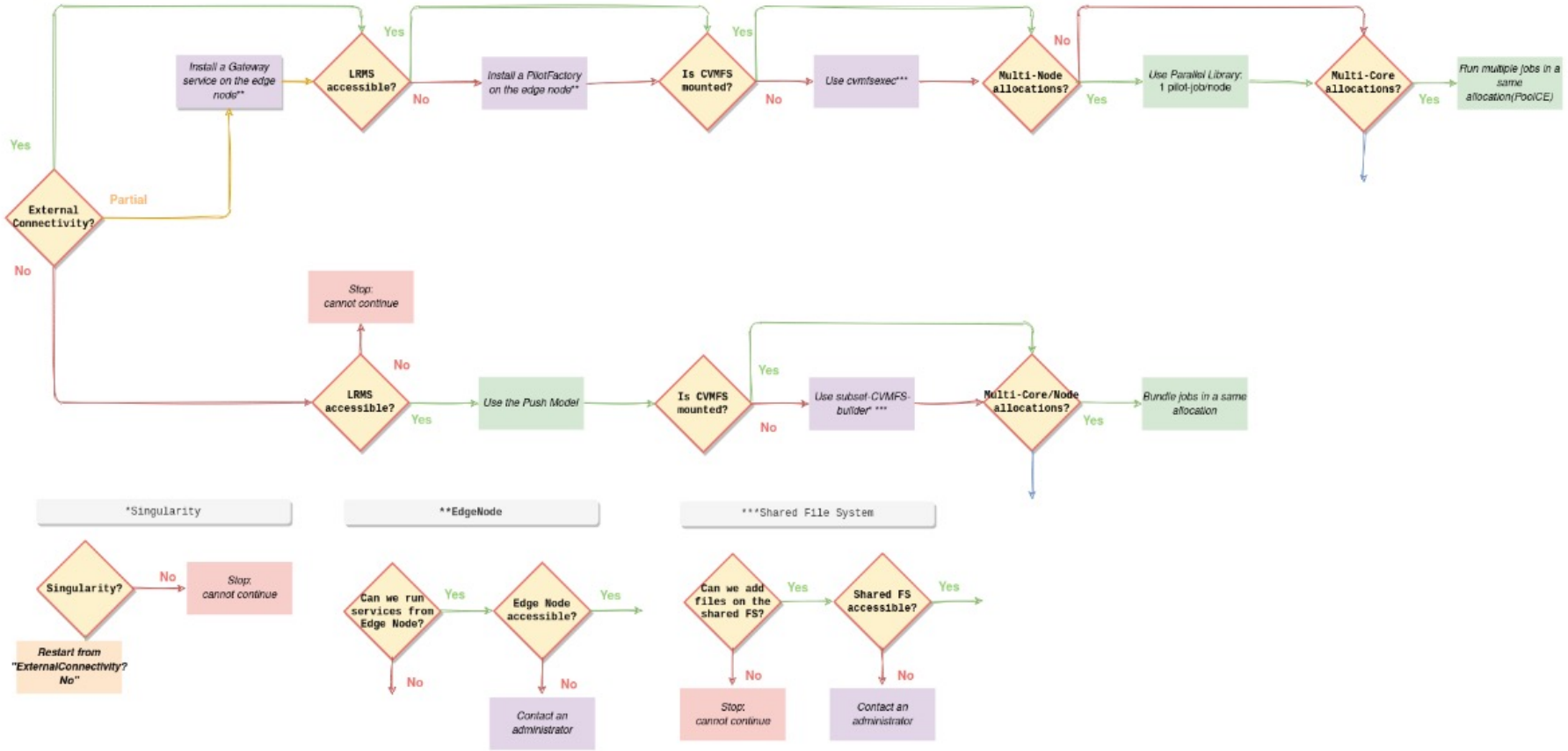
Resource allocation in “push” mode

- If no outputbound connectivity possible
- PushJobAgent: runs “pilots” in the central service pushing only user applications to LRMS
- Application and DIRAC software preinstalled in LRMS
  - E.g. CVMFS snapshot created in a CI pipeline and mounted as /cvmfs
- Job’s output sandboxes and data are extracted via the ssh tunnel
  - Or via the ARC interface if available



HPC cases

- A solution for a particular HPC is usually a compromise of getting the best of what is possible



DIRAC
THE INTERWARE

Conclusion

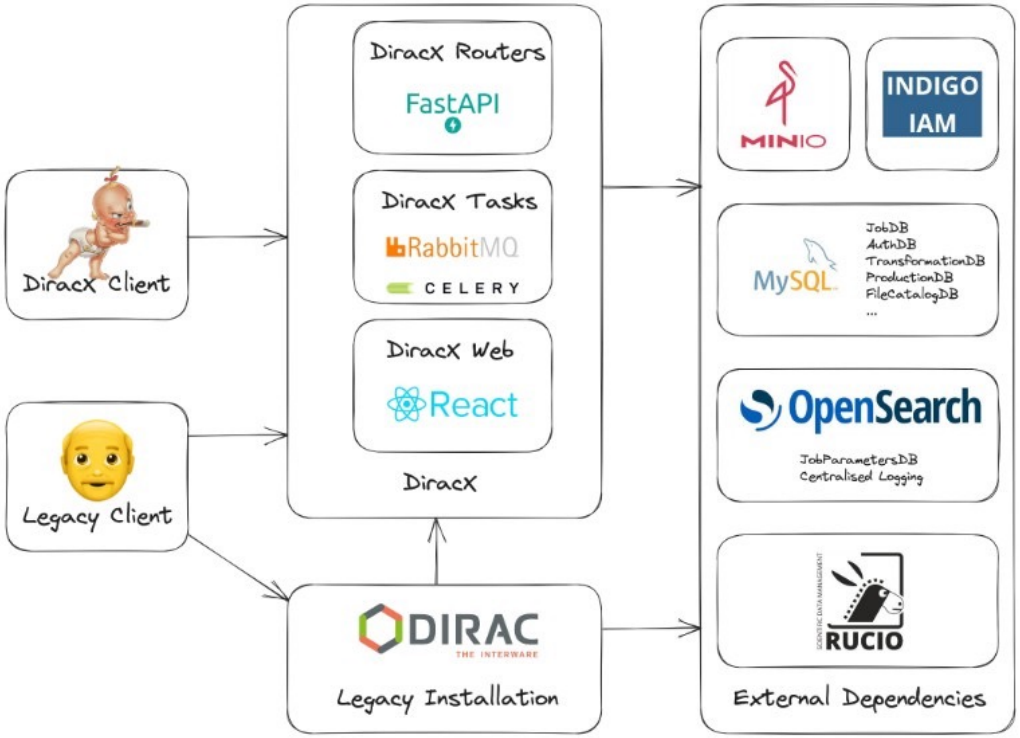
- DIRAC is an example of a product that evolved from a single experiment development to an open-source project exploited by multiple scientifique communities
- DIRAC offers a complete solution for all the computing and data management tasks for research communities
- DIRAC WMS is flexible enough to adapt to specific HPC infrastructures.
- DiracX is a major upgrade to meet the requirements of large user communities for the years to come. The ongoing developments take into account the HPC access requirements

DIRAC
THE INTERWARE

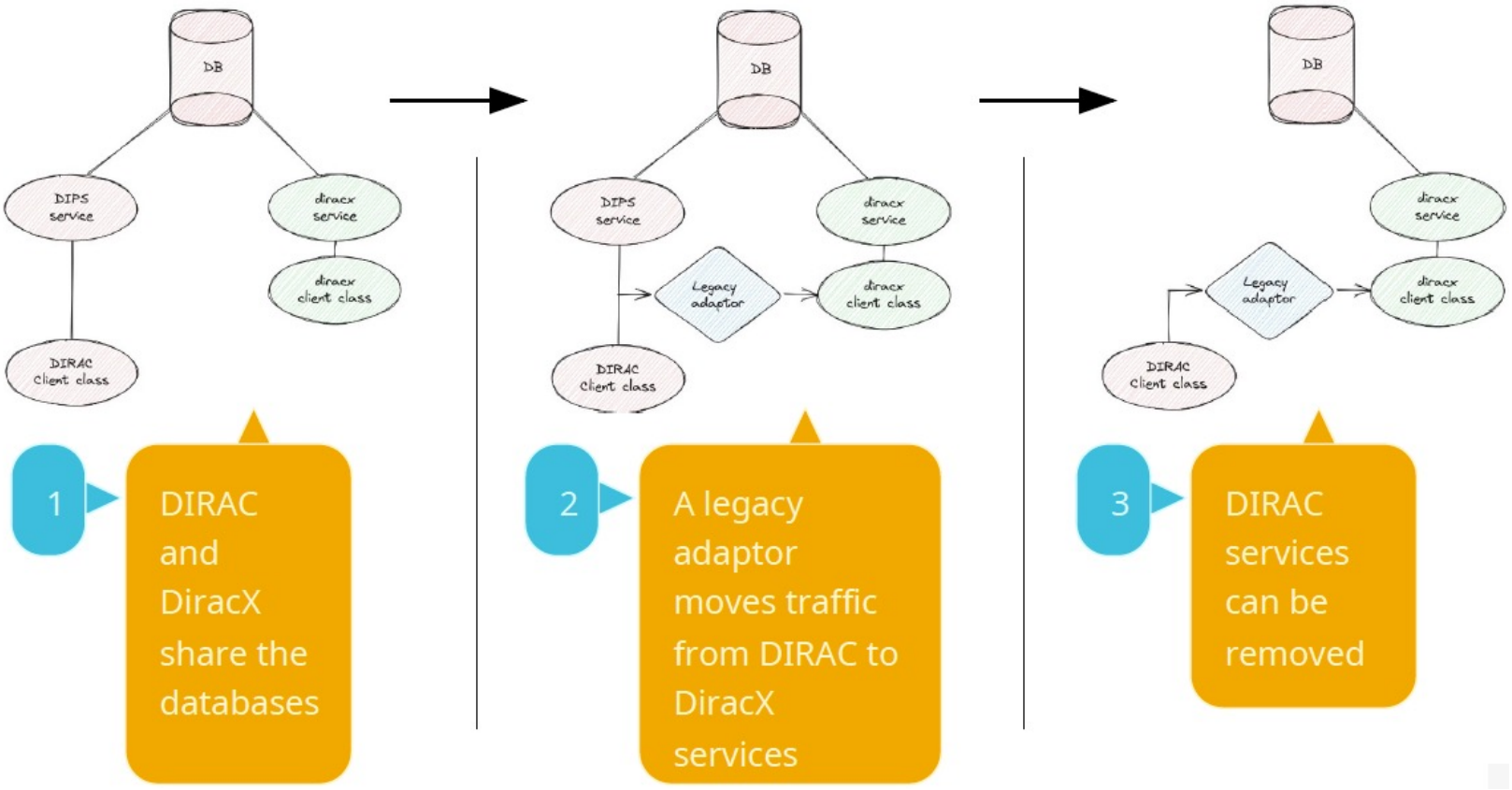
Back-up slides

DIRAC/DiracX architecture

- Running in parallel DIRAC and DiracX components
  - To ensure smooth transition of production installations
- Keeping common databases for DIRAC and DiracX components
  - **DIRAC 9.0**
  - **DiracX 0.1**
- Different installations can migrate at different pace
  - **DIRAC 8.0** will be supported up to 2027



DIRAC->DiracX migration

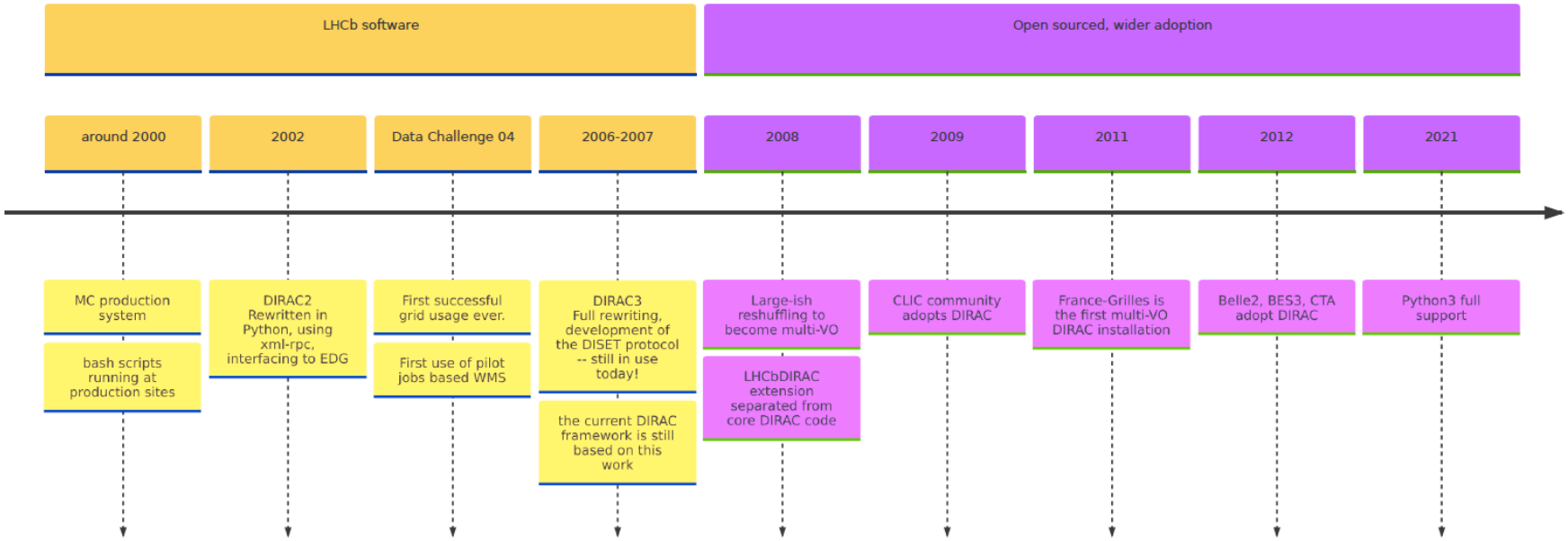


RucioFileCatalog

- Multiple requests to allow DIRAC to work together with the Rucio DMS.

- The RucioFileCatalog plugin developed by the Belle II experiment
  - Developments done both on DIRAC and Rucio sites
  - Implements the DIRAC File Catalog interface
  - Allows access of the DIRAC WMS and DMS components to get the file information from Rucio
    - Data aware job placement
- Now used by the CTAO Collaboration, offered by the GridPP DIRAC service
- DIRAC-Rucio joint workshops to elaborate common strategies towards better integration of the projects

Timeline



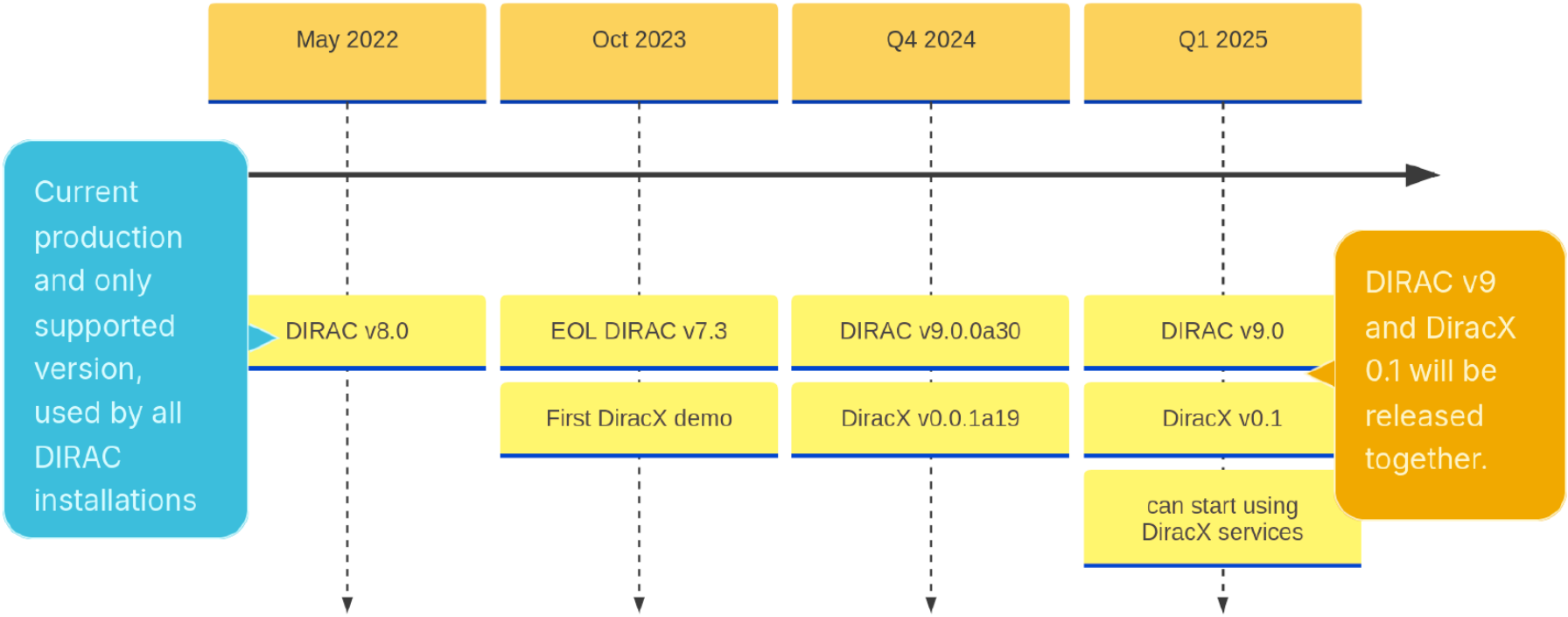
- Evolving from a bunch of shell scripts to a general purpose distributed computing framework
  - DISET secure protocol with data streaming support
  - Multi-VO, extendable
  - Adopted by several HEP communities and grid infrastructure projects

Upgrade: technologies

Major technologies :

- Services
  - DiracX Web APIs with
  - APIs documented with
  - Following the specification by
- Diracx Web Portal
  - NextJS
  - Material UI
  - TypeScript
- Deployment
  - Kubernetes + Helm
- Security
  - OAuth2/OIDC tokens

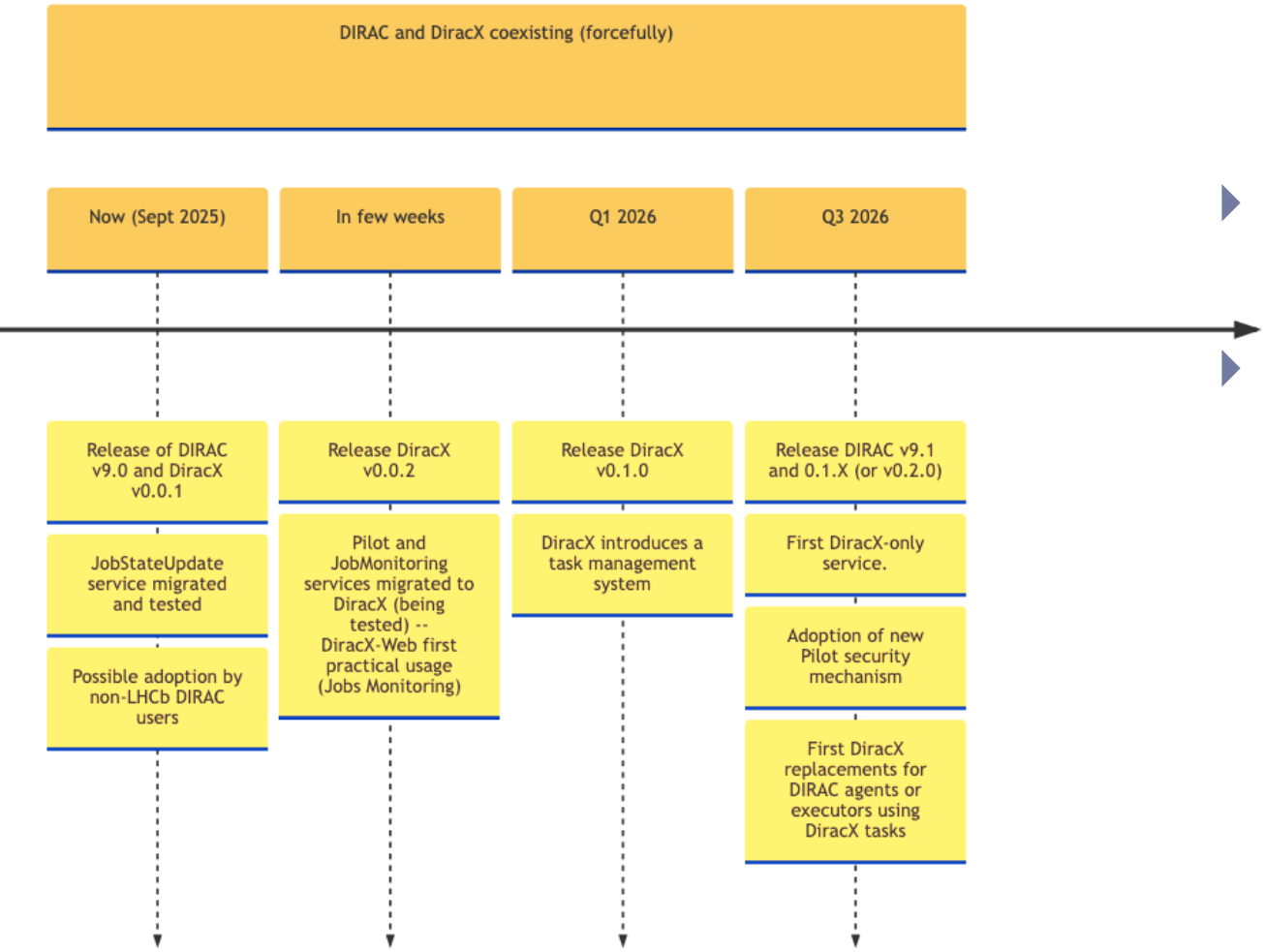
Releases



- LHCb deployed DIRAC9/DiracX0.1 in production in April'25
  - During the LHC technical stop, 10 days of the service shutdown
    - Together with several database optimizations
  - Successful restart after several fixes done
- Invaluable experience for other DIRAC installations to follow
- Gradular migration of all the DIRAC subsystems in the next 1-2 years
  - First Release DIRAC v9.0 + DiracX v0.1 in September'26

Roadmap

2026 Developments



- Q4'2026 EOL
  - DIRAC v8
- Q4'2027
  - DiracX v1.0
  - Full replacement of DIRAC v9
  - Very indicative guess