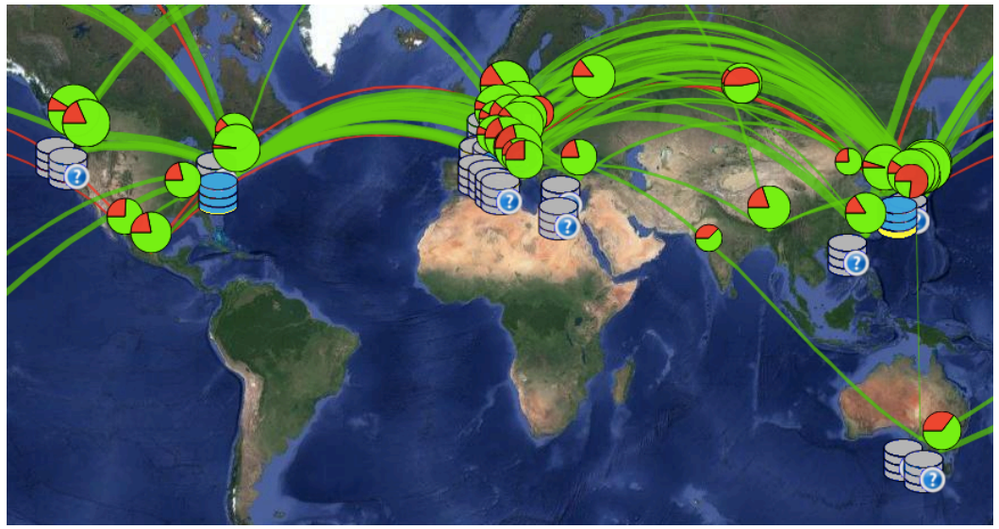
BelleDIRAC

The DIRAC extension for Belle II

Ueda I.

2018.May.23. DIRAC UWS

Belle II Resources



SEs

- Total: 26
  - DPM: 10, dCache: 7, StoRM: 8, BeStMan: 1
  - a few are retiring…

Sites (CEs)

- Total: ~55 (some sites with multiple CEs, some sites subdivided in config, …)
  - LCG (cream): 23, OSG (HTCondorCE): 2, ARC (ArcCE): 5
  - DIRAC (ssh+batch): 20 + some test,
  - DIRAC (local SiteDIrector+cloudscheduler): 1
  - CLOUD (VMDIRAC): 1 + (temporary use of AWS)
  - VCYCLE: 3

Distributed world-wide

- Heterogeneous and wide-spread
- Not every site has a SE, some sites are far from any SEs
- Not every SE is large and stable (in other words, not every SE is T1/T2D…)

Belle DIRAC Installation

Main servers

- 6 servers at KEK running most of the components
- 1 separate server to run WebApp at KEK
- 4 separate servers at KEK to run MySQL
- 1 server at BNL to run Belle II DDM components
- A few servers to run SiteDirector, CS slave and ReqProxy at other sites

Test servers

- “Certification” — to test new BelleDIRAC codes
- “Migration” — to test upgrade of base DIRAC, upgrade of BelleDIRAC components with big jumps, …

Admins

- Hideki (and myself partially) manages the installation at KEK
- Malachi and co. have been managing the installation at PNNL — moved/moving out to BNL
- Hiro Ito and John DeStefano now started managing the installation at BNL

BelleDIRAC

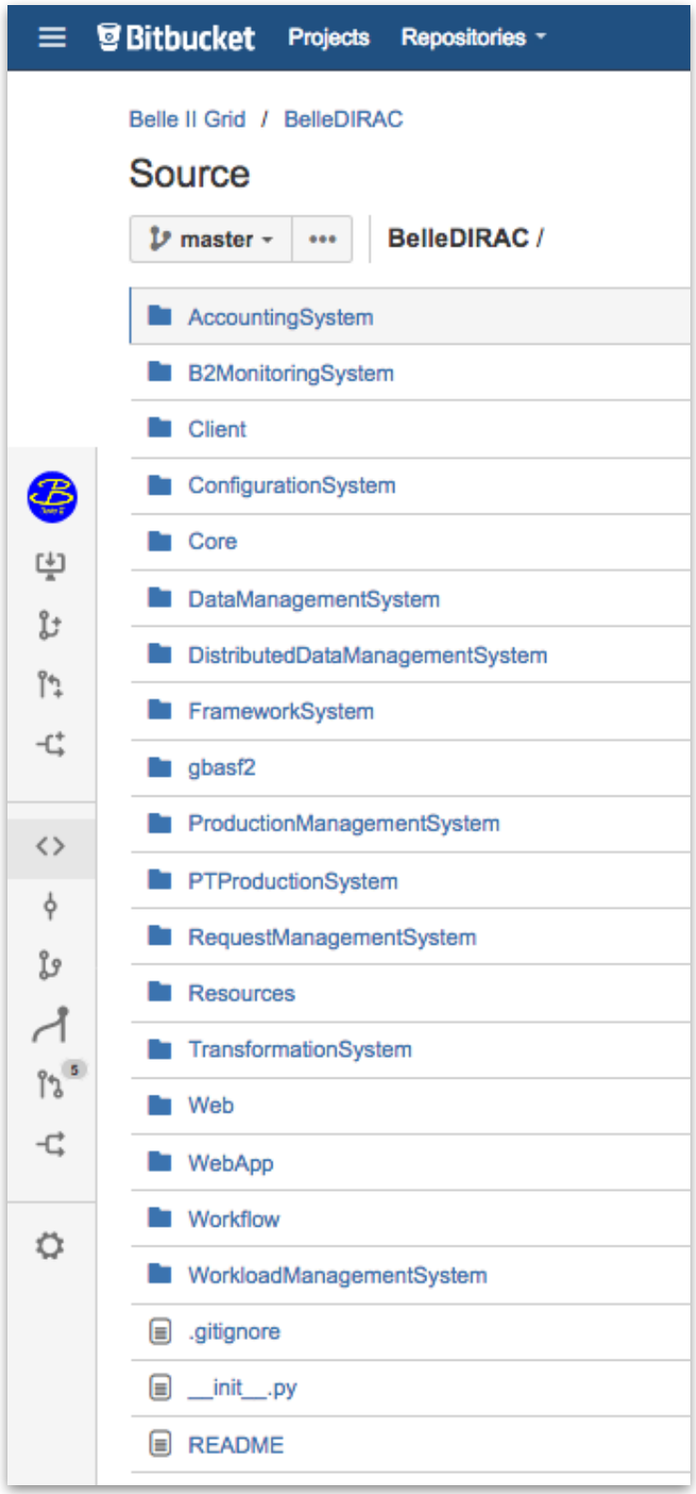
Currently based on DIRAC v6r17

Name:

- The Experiment name is “Belle II” (http://belle2.jp/)
- The VO name is ‘belle’
- Thus, our extension is “BelleDIRAC”

Contents:

- B2MonitoringSystem
- Client
- DistributedDataManagementSystem
- FabricationSystem
- PTProductionSystem
- ProductionManagementSystem
- gbasf2
- AccountingSystem
- ConfigurationSystem
- Core
- DataManagementSystem
- FrameworkSystem
- RequestManagementSystem
- Resources
- TransformationSystem
- Web/WebApp
- WorkloadManagementSystem



Extensions on vanilla Systems

(Some pick ups)

DataManagementSystem

- AMGA related implementations

FrameworkSystem

- Archiving log

TransformationSystem

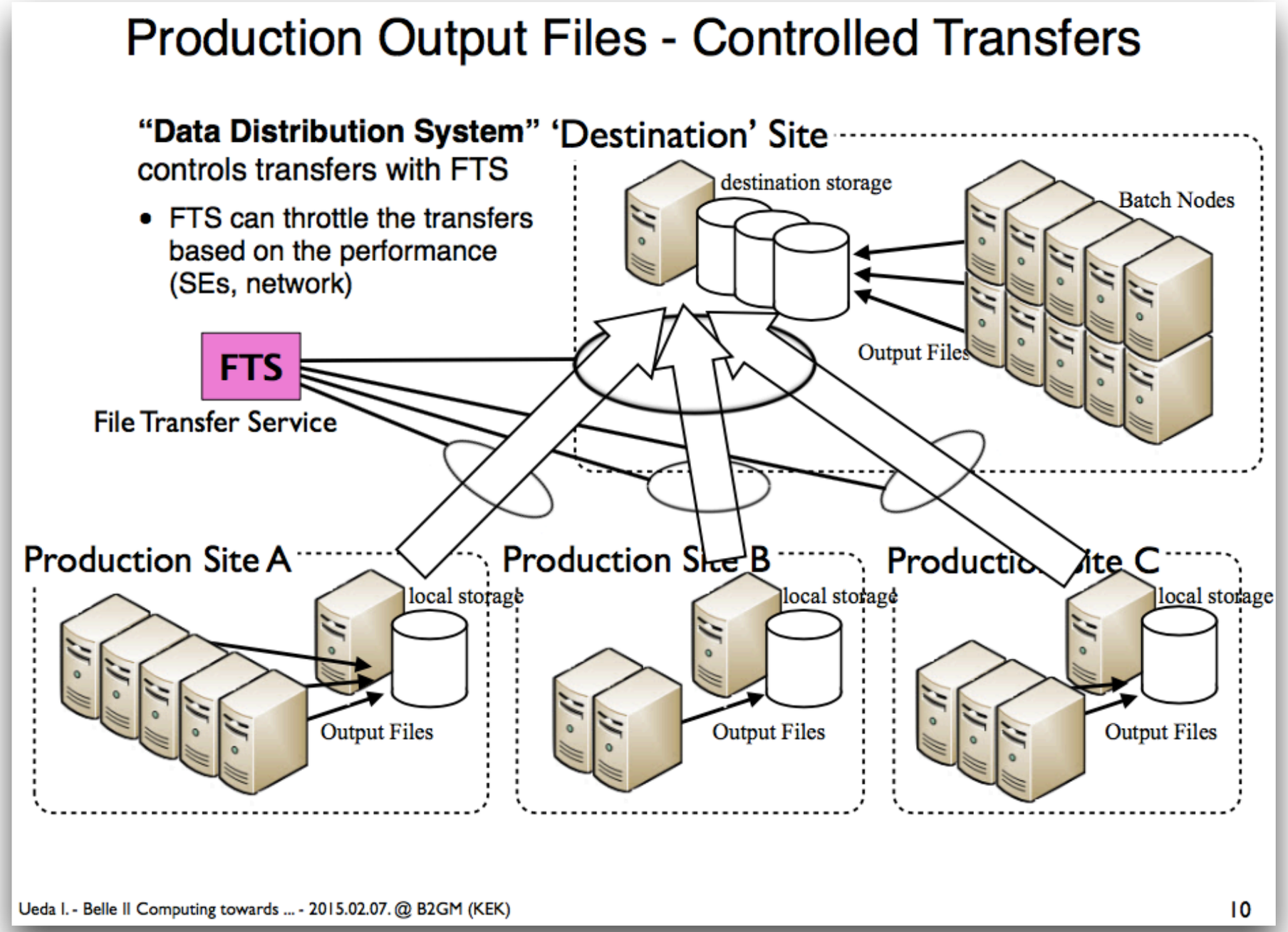
- Belle II specific plugins

WorkloadManagementSystem

- BellePilotCommands.BelleInstallDIRAC
  - Get files from CVMFS instead of downloading from Web
  - Currently only for Belle II with hard-coding, but possibly generally useful

FTS transfers to Primary SEs

The very first thing we implemented for a large scale production: to gather output data to a set of selected SEs.



Data Management Blocks

Reminder

https://indico.cern.ch/event/477578/contributions/2143193/

Datasets

- Belle II produces **various types of MC data**
  - Organised as “datasets” (defined as a part of LFN path)
- **“Runs” in real data** are also considered as “datasets”
- **Clustering files** of the same dataset onto the same SE, *to some extent*, would ease some workflows
  - jobs with multiple input — merge, analysis, … — can avoid remote downloads
- A dataset can contains O(100k) files — *too many* as a unit of data management
  - eg. 32k files is a limit to be placed under a directory

Data Management Blocks

- **“Data block”** as a unit of data management — to lower pressure to “file” catalog
  - max 1000 files as initial implementation, so far so good.
  - A key for scalability: O(1000) less look-up than per file
- **“Dataset”** is the unit of production, but *the system organises files in **“data blocks”***
  - Some parts of the system are (to be) implemented based on “data blocks”
- Subdirectories under “dataset” path: `LFN = /belle/…dataset…/subNN/file`

Production System

Reminder

https://indico.cern.ch/event/477578/contributions/2143193/

An extension of DIRAC

- Base DIRAC handles jobs and files
- Belle II Production System handles “productions” and “datasets/datablocks”
  - Auto-creates jobs based on “production definitions”, submits them to DIRAC, checks the status, recreates and resubmits jobs when necessary, manages datablock placement

Composition

- **Production Management**
  - **ProductionManagement System** to define and manage what to produce
  - **Fabrication System** to define and manage jobs
- **Dataset Management**
  - **Distributed Data Management (DDM) System** to gather files in datablocks to “Primary SEs”.
- **Monitoring** and **automatised operations**
  - **B2Monitoring**

BelleDIRAC components named avoiding those (possibly) in vanilla DIRAC

We had to rename “Monitoring” to “B2…”

Production Management

Production Management System

- Production Managers (human) define “productions” and register them to ProductionManagementSystem
- Production Management System defines “Fabrications” and “chain” them according to the definitions

Fabrication System

- Uses Transformation System to define tasks (and then jobs)
- Fills data blocks with the output files
- Triggers data gathering via DDM (next slide)

Developed by Hideki Miyake

More details discussed in the “Productions Management” session

Data Management

Distributed Data Management System (Belle II DDM)

- does not use TransformationSystem
- manages “datablocks”
- submits file transfers to RMS
- deletes files by itself, not via RMS
  - to avoid overload on LFC and to control “priorities”

TS could have been the solution, as in LHCb
Initially tried, but the developers abandoned the idea

Currently directly calling lcg/gfal2.
Should use DIRAC APIs, with extensions where needed

Use cases

- To gather output files to “primary” SEs by “data block”
  - move == replicate by RMS + delete source by itself
- To distribute products over the grid by “data block” (yet being implemented…)

Developed by PNNL team (M. Schram, V. Bansa, et al.)

- The responsibility has moved from PNNL to BNL… (A. Undrus, S. Padolski)

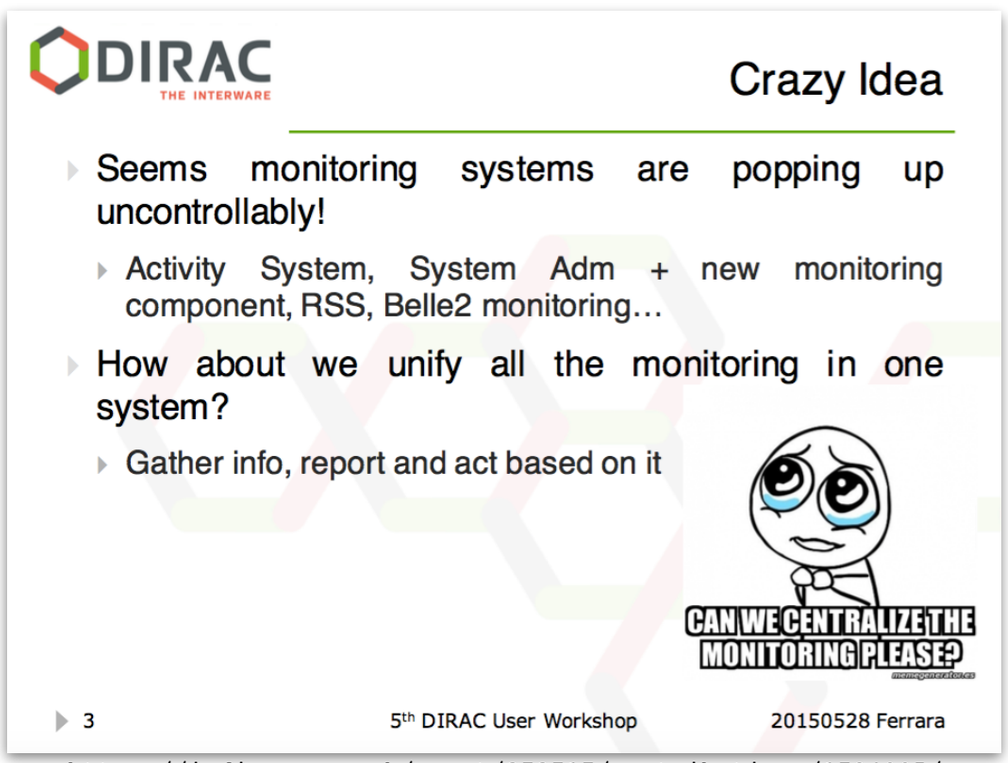
More details discussed in the “Distributed Data Management” session

Monitoring

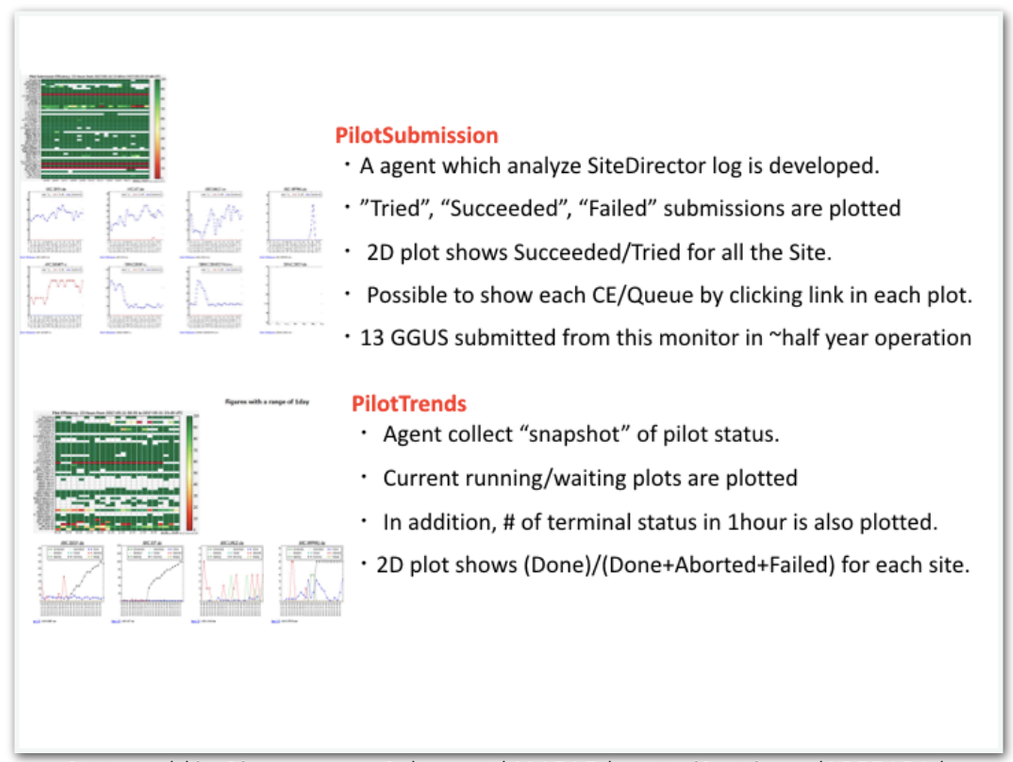
B2Monitoring

- initially developed “outside” of DIRAC
  - presented at the WS in Ferrara
- imported into “DIRAC” framework
  - presented at the WS in Warsaw
- being implemented in more “proper DIRAC” way
  - to be integrated into vanilla DIRAC (with the help by CH, ZM, FS, …)

Developed by “Nagoya” team (K. Hayasaka, Y. Kato)



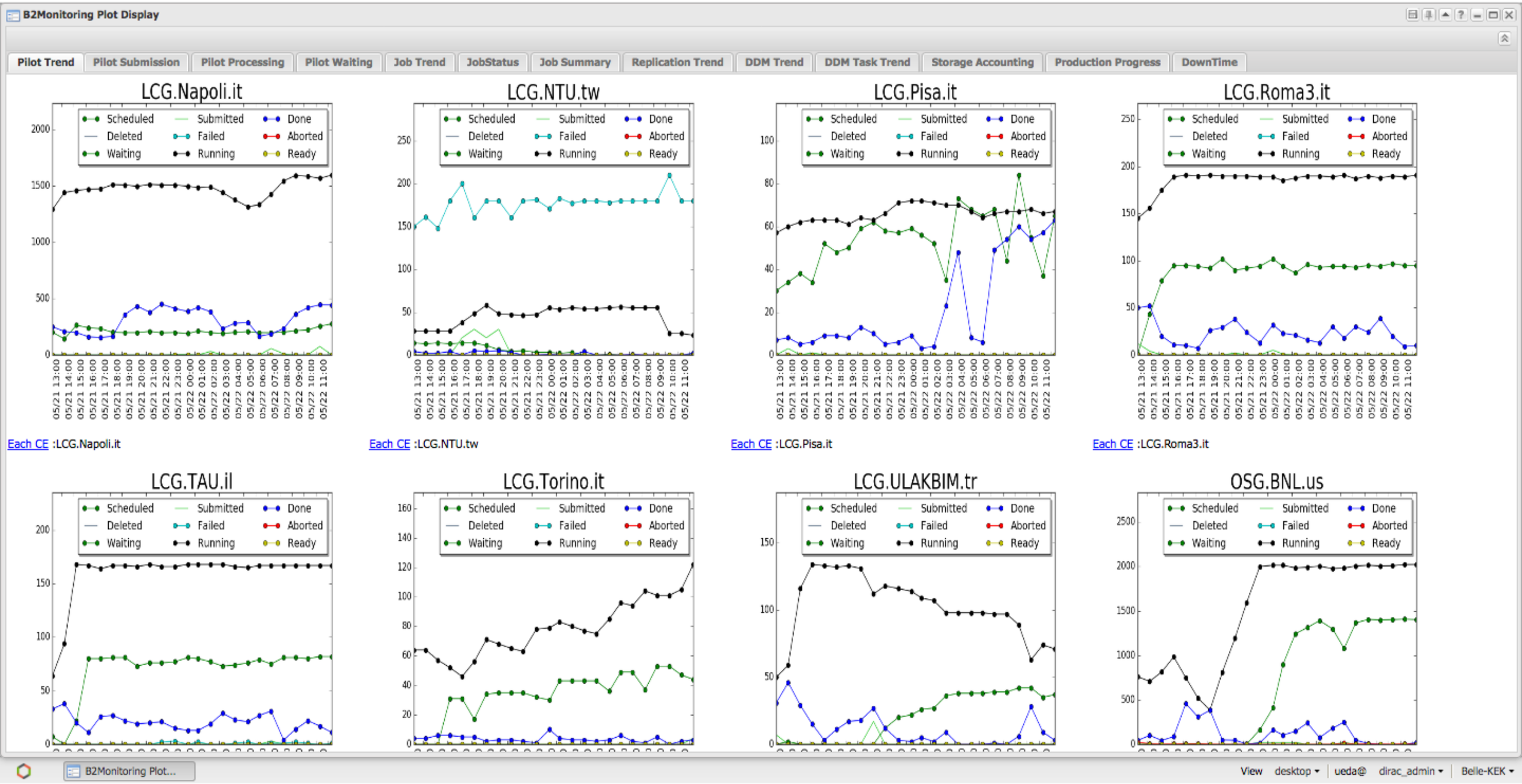
https://indico.cern.ch/event/372717/contributions/1794005/



https://indico.cern.ch/event/609507/contributions/2577154/

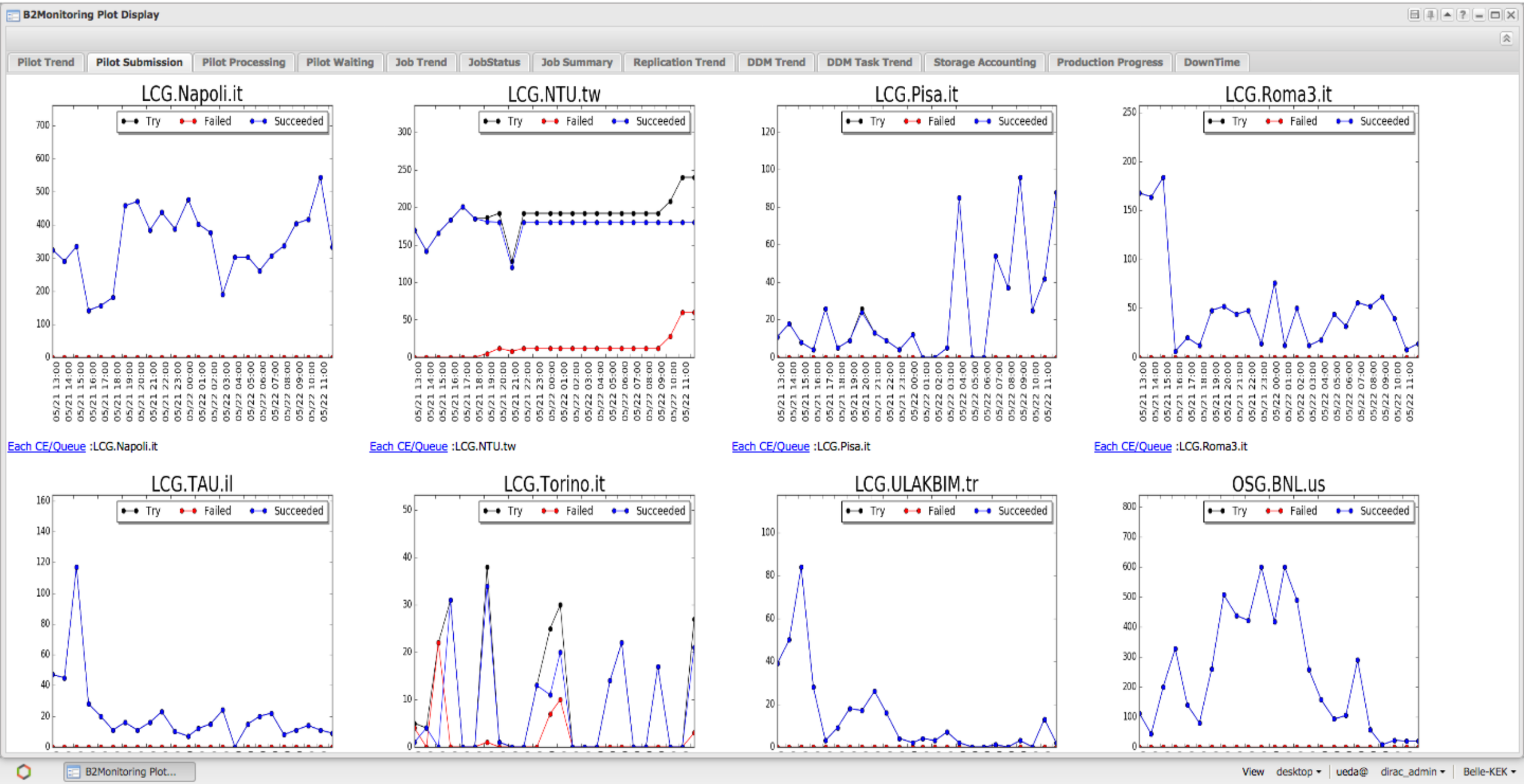
Pilot Status

Pilot stats in Accounting do not have “Waiting”



Pilot Submission Status

Pilot submission stats are not in Accounting



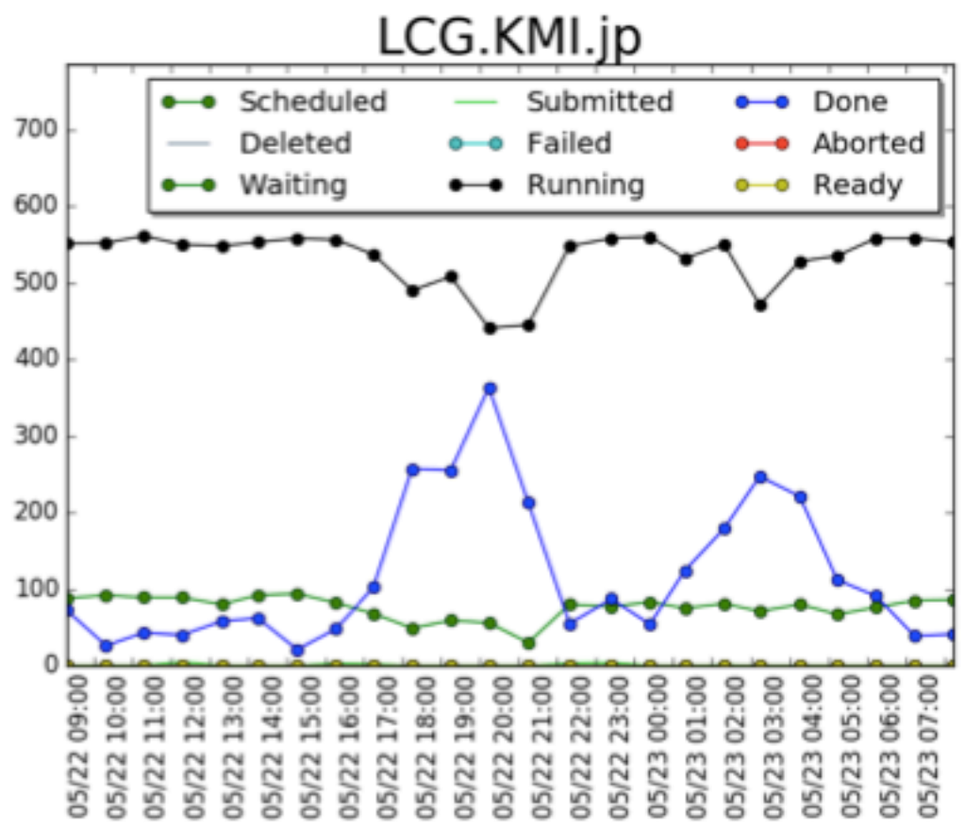
Plan to implement pilot submission monitoring on DIRAC

- Currently, the log file of SiteDirector is analyzed and plots are made by BelleDIRAC original API and stored into Belle2 Monitoring database.
- Change in following strategy.
  - Change plotting API to incorporate with base DIRAC.
  - Implement PilotSubmission category in the AccountingSystem.
  - Change SiteDirector to fill the pilot submission statistics into Accounting.

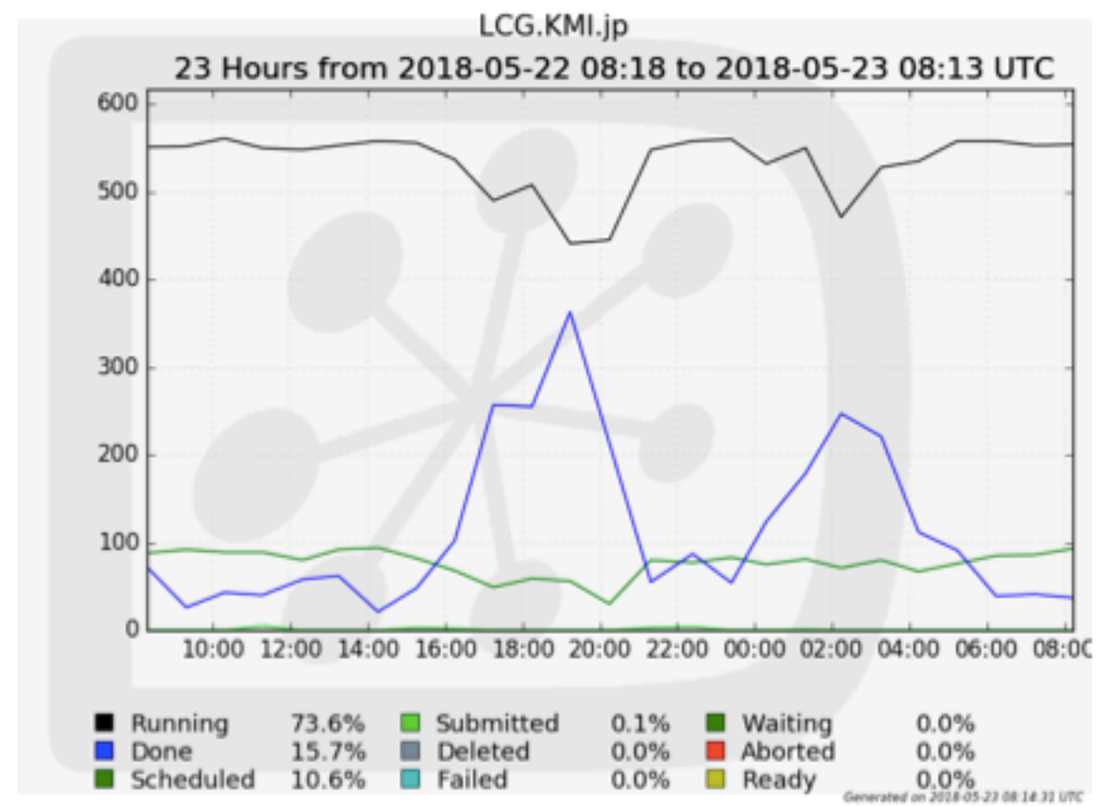
Yuji Kato

Moving plot API based on original DIRAC

Plots currently used for BelleDIRAC monitoring

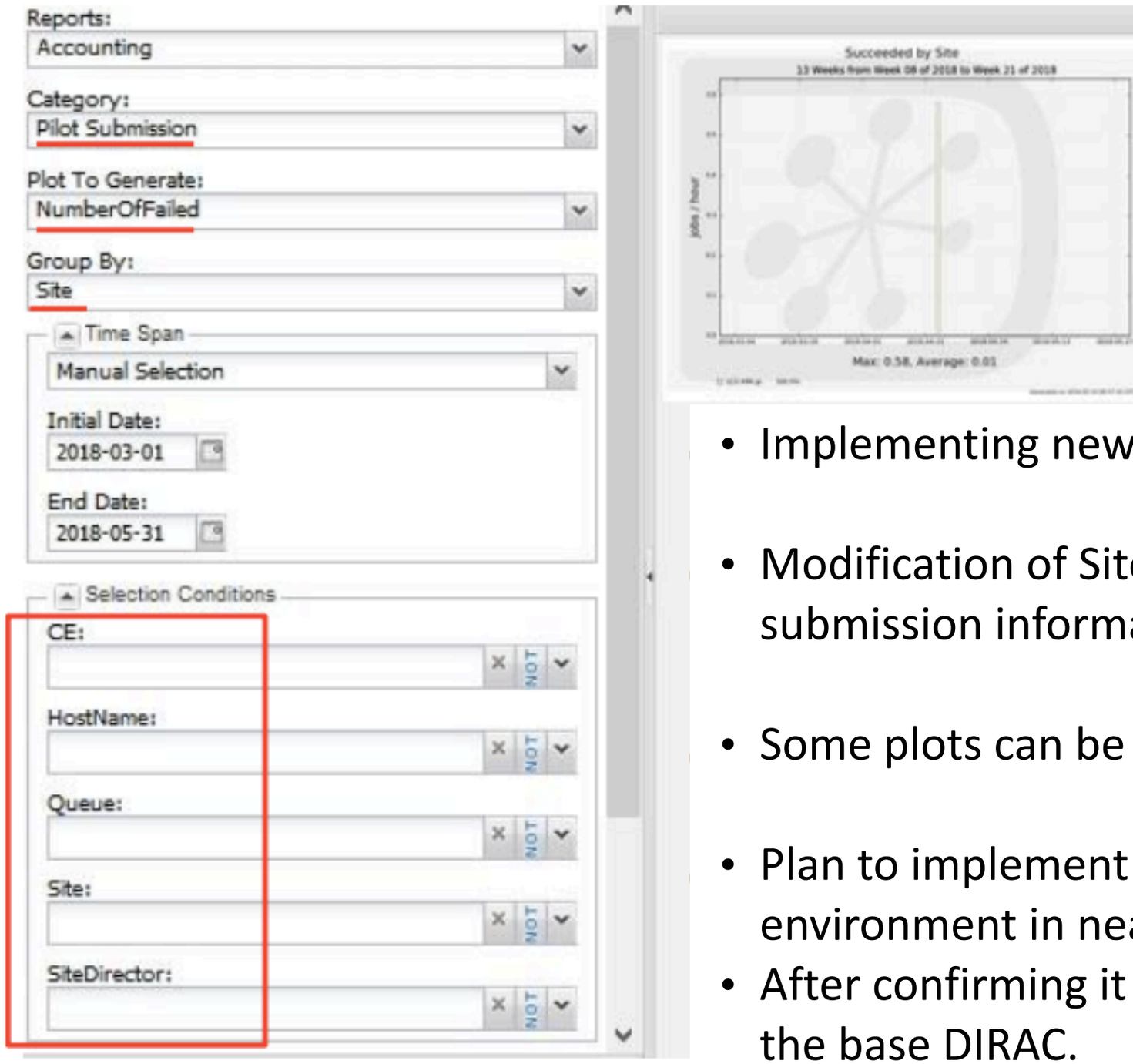


The same type of plot based on DIRAC API



Yuji Kato

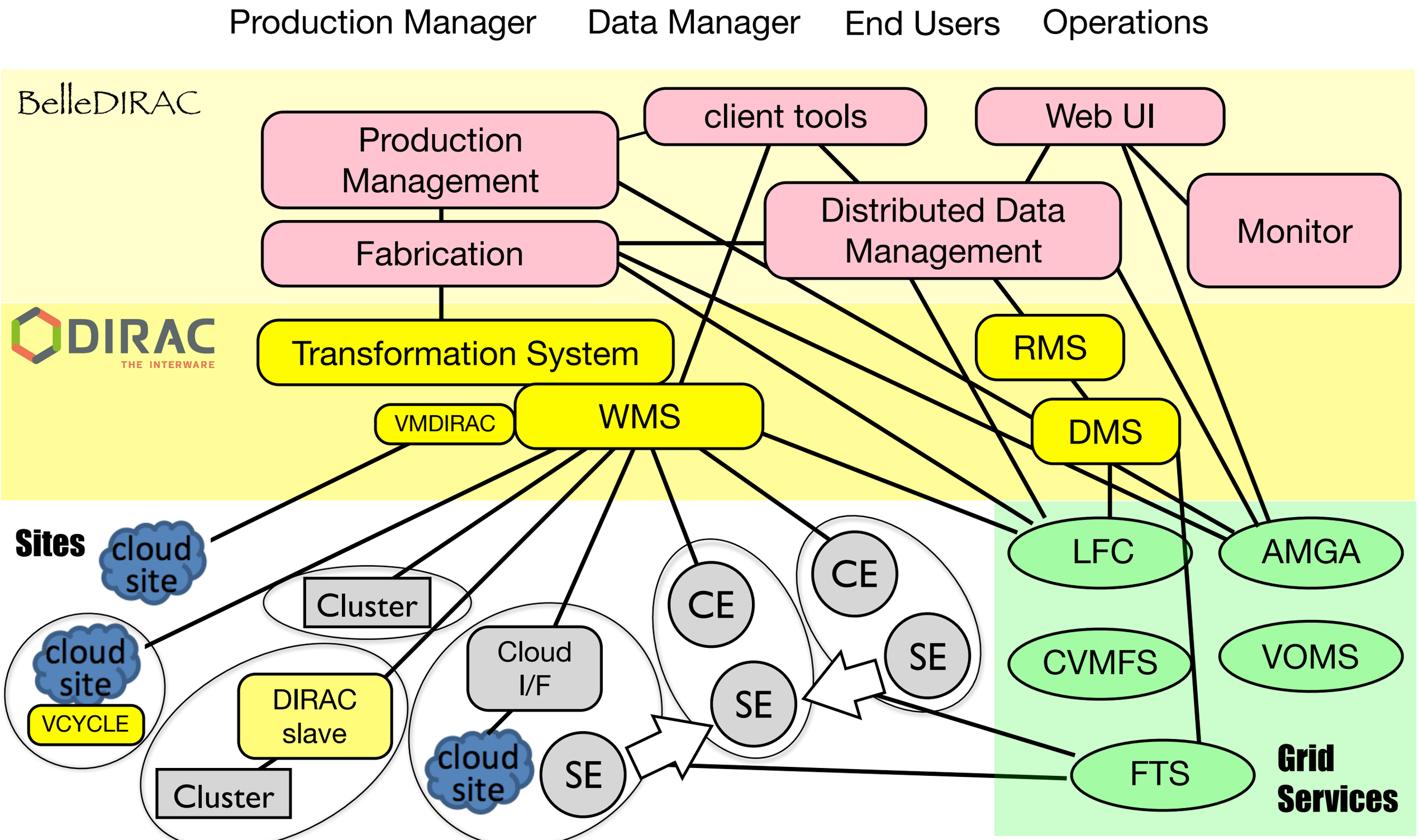
Implement PilotSubmission on Accounting



- Implementing new Category is done.
- Modification of SiteDirector to fill pilot submission information is done.
- Some plots can be made.
- Plan to implement in Belle II production environment in near future.
- After confirming it is working, implement in the base DIRAC.

Yuji Kato

Belle II Distributed Computing System



Cloud Resources

VMDIRAC

- We used to use VMDIRAC v1
  - Rafał Grzymkowski as the expert
  - Some academic clouds and AWS as the resources
- We tried to use VMDIRAC v2
  - Basically it works (in tests), not used in production (yet),
    - got stuck in preparing a “host certificate” — Rafal has an idea to overcome this issue

VCYCLE

- We have been using VCYCLE since some time
  - Silvio Pardi as the expert
  - Small (test) cloud at Napoli and (large) HNSciCloud testbed as the resources

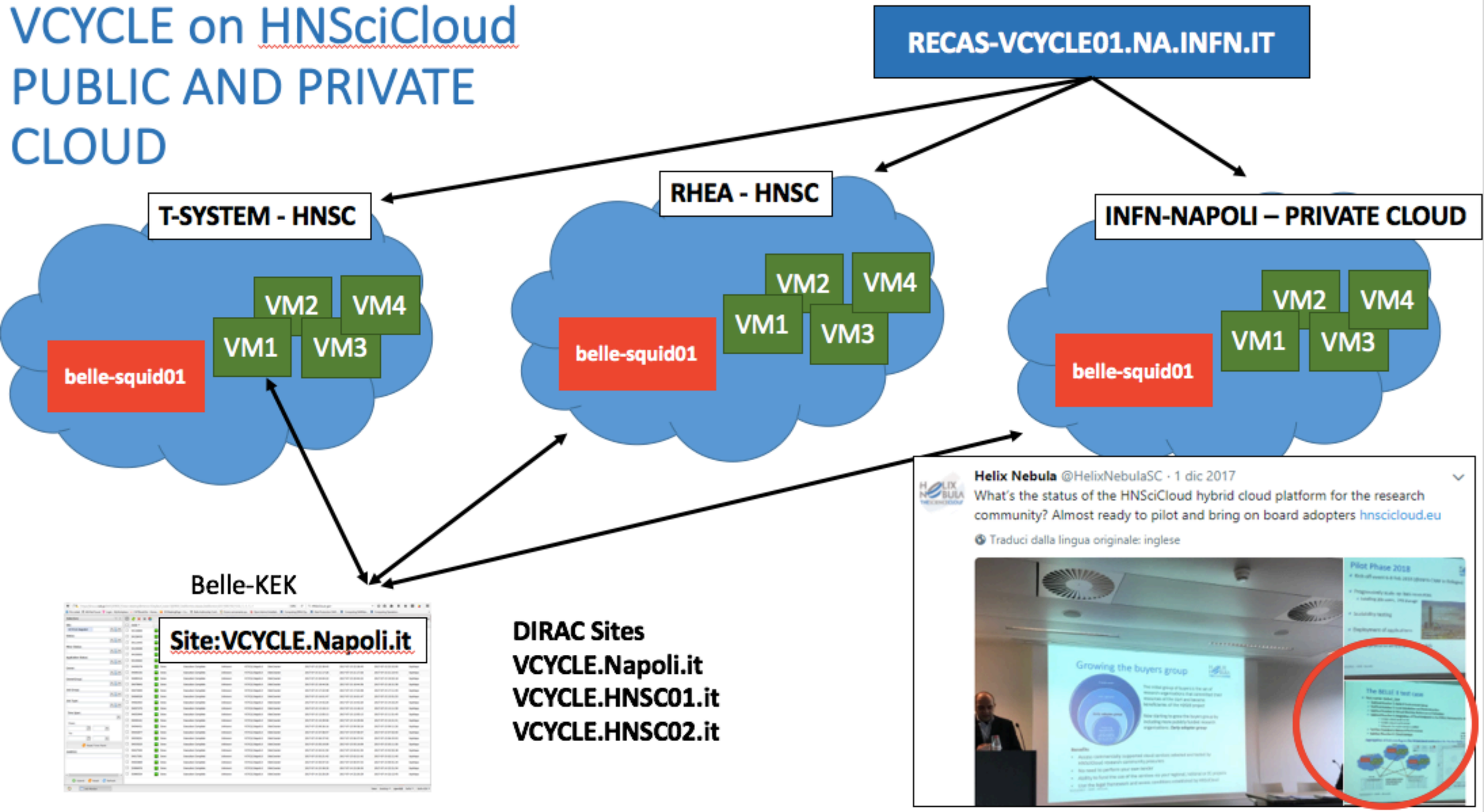
CE-like Cloud I/Fs

- Cloud Scheduler (UVic), Dynamic Torque (Melbourne/CoEPP)

Another implementation to use “cluster services” on cloud (in dev)

- Rafal presented his solution (Serverless CE)

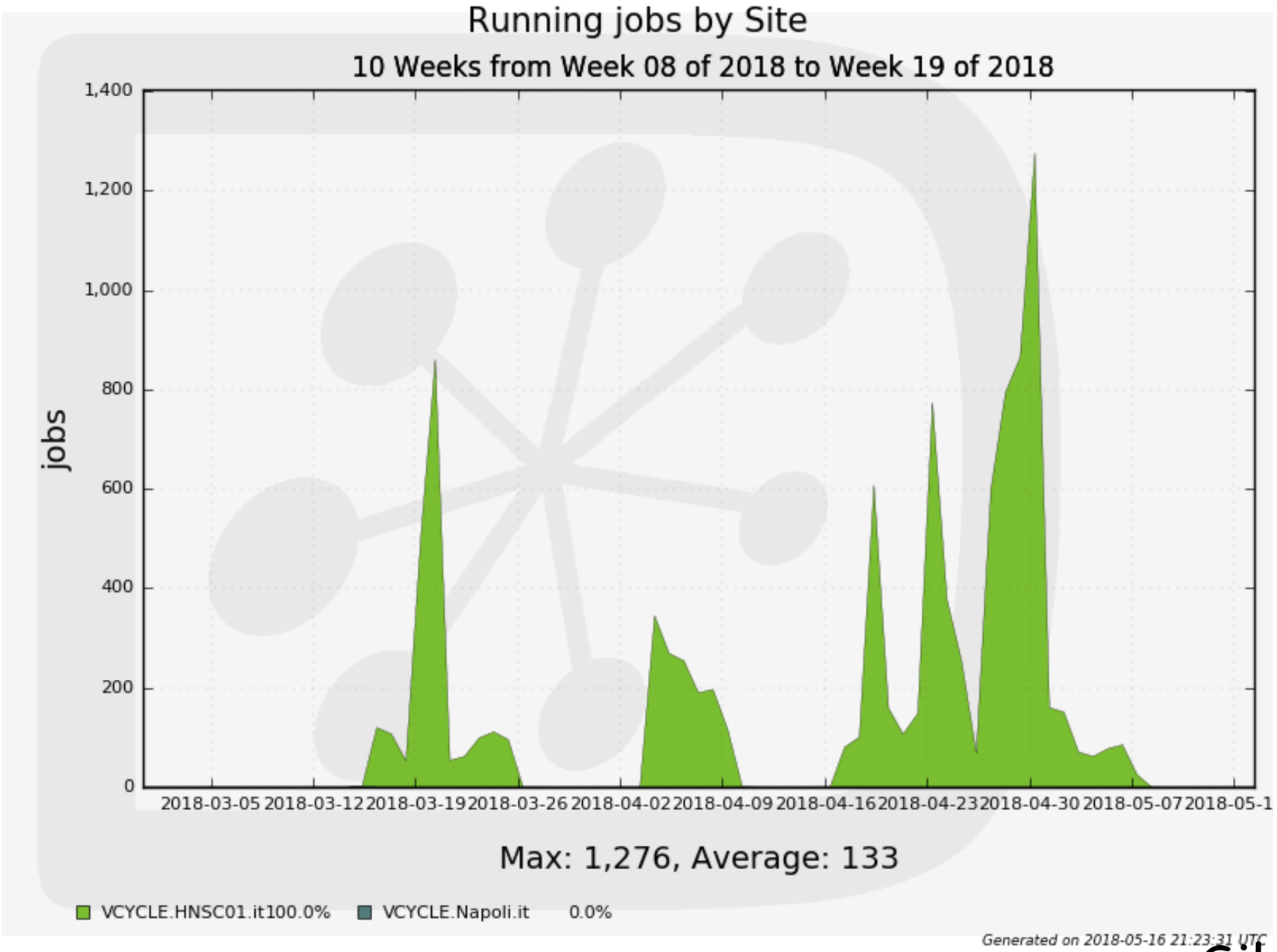
VCYCLE on HNSciCloud PUBLIC AND PRIVATE CLOUD



Silvio Pardi

Running jobs in HNSciCloud at a Scale

as a scalability test of HNSC

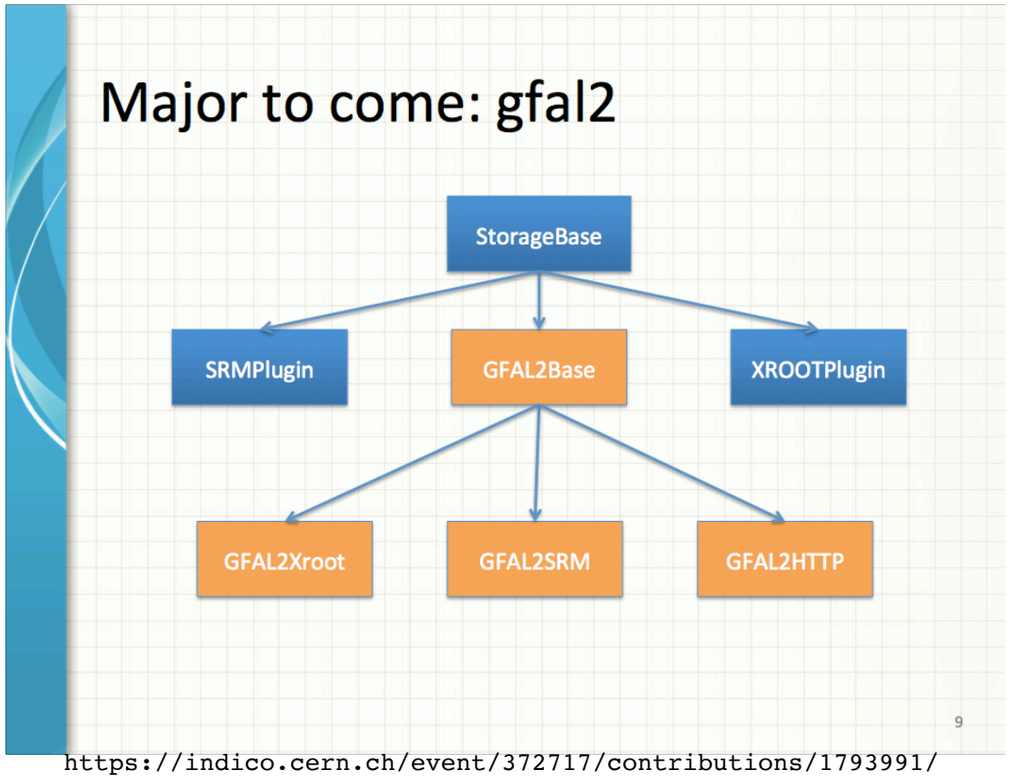


Silvio Pardi

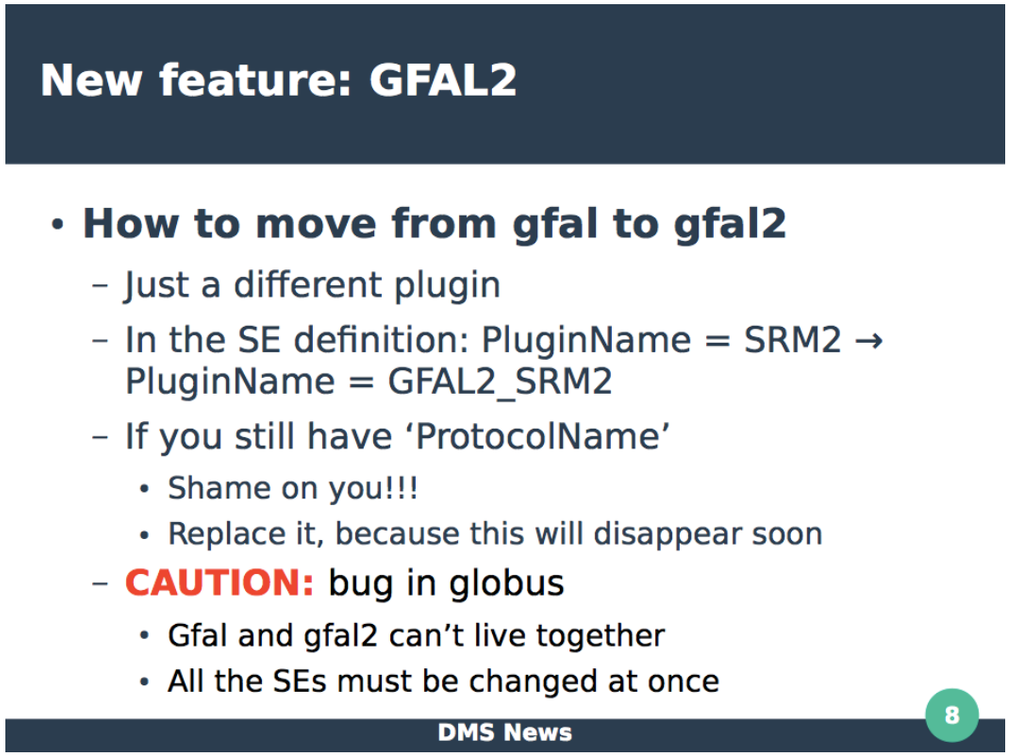
Data Management with HTTP

We are HAPPILY trying this out

- Only “trying” for now… not in production, yet…
  - … for we found our DDM was not implemented to be compatible with this
- I have set up a few endpoints in a test environment (with a great help from CH)
  - and confirmed the basic functions
- Our colleague in Napoli (SP) is trying to use it
  - per SE and via dynafed
- Our colleagues in UVic are waiting for it
  - to use their dynafed with their clouds



https://indico.cern.ch/event/372717/contributions/1793991/



https://indico.cern.ch/event/609507/contributions/2577187/

HTTP Use Cases

SEs with HTTP I/F

- To read files (direct i/o) from non-grid offline software
- To get rid of SRM when needed, or where http is more efficient

Global Dynafed

- Can be used as a generic fail-over SE
- Can be useful for payload jobs to find “parent” of the input files
  - Pilot jobs run our “grid” jobs. A grid job is a wrapper to run “non-grid” offline software.
  - Users may access “parent files” of the input files, but the offline software does not know anything about grid… and cannot look up FC. So, either
    - our grid wrapper to look up the FC and give SURLs to the jobs
    - we give a generic non-grid URL for users to specify in their jobs == dynafed

UVic Dynafed

- UVic “cloud” is distributed world-wide. UVic SE is not necessarily close to the WNs. They can set up a storage in each cloud and organize them with dynafed.

Dynafed in BelleDIRAC config

```
Dynafed-Napoli-SE
{
    BackendType = dpm
    ReadAccess = Active
    WriteAccess = Banned
    RemoveAccess = Banned
    AccessProtocols = https
    AccessProtocols += davs
    AccessProtocols += davs+3rd
    WriteProtocols =
   AccessProtocol.0
    {
        Host = dynafed01.na.infn.it
        Port = 443
        PluginName = GFAL2_HTTPS
        Protocol = https
        Path = /myfed
        Access = remote
      }
    }
```

```
Dynafed-UVic-SE
{
    BackendType = dpm
    AccessProtocols = davs
    WriteProtocols = davs
    ThirdPartyProtocols = davs
    ThirdPartyProtocols += davs+3rd
    ThirdPartyProtocols += https
    AccessProtocol.0
    {
        Host = dynafed02.heprc.uvic.ca
        Port = 8443
        PluginName = GFAL2_HTTPS
        Protocol = https
        Path = /
        Access = remote
        DisableChecksum = True
      }
    }
```

No more need for "davs+3rd"

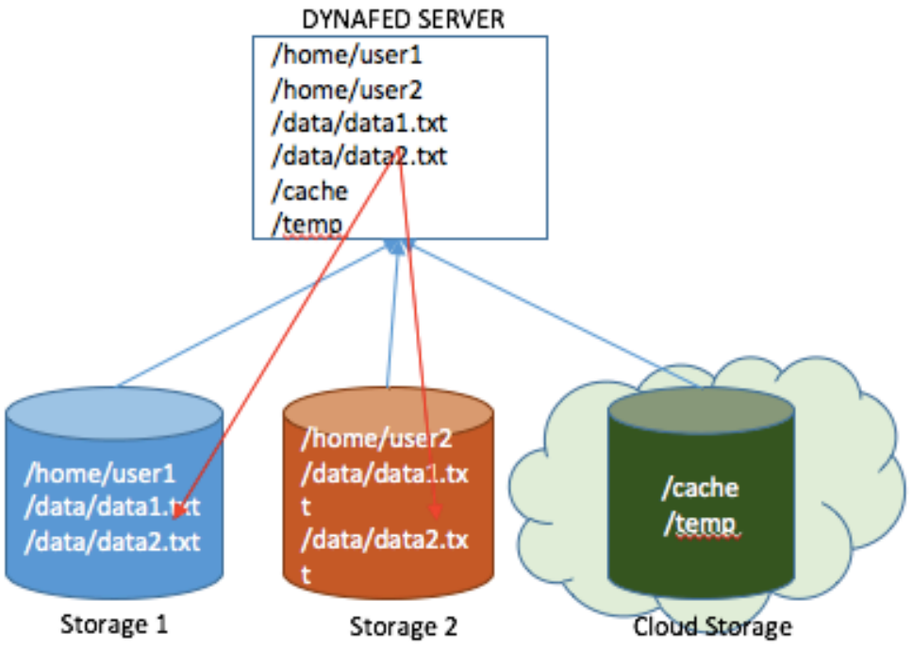
Global Dynafed

Dynafed Server for Belle II

#	STORGE NAME	HOSTNAME	TYPE
1	DESY-DE	dcache-belle-webdav.desy.de	DCACHE
2	GRIDKA-SE	f01-075-140-e.gridka.de	DCACHE
3	NTU-SE	bgrid3.phys.ntu.edu.tw	DCACHE
4	SIGNET-SE	dcache.ijs.si	DCACHE
5	UVic-SE	charon01.westgrid.ca	DCACHE
6	BNL-SE	dcbldoor01.sdcc.bnl.gov	DCACHE
7	Adelaide-SE	coepp-dpm-01.ersa.edu.au	DPM
8	CESNET-SE	dpm1.egee.cesnet.cz	DPM
9	CYFRONNET-SE	dpm.cyf-kr.edu.pl	DPM
10	Frascati-SE	atlasse.lnf.infn.it	DPM
11	HEPHY-SE	hephyse.oeaw.ac.at	DPM
12	Melbourne-SE	b2se.mel.coepp.org.au	DPM
13	Napoli-SE	belle-dpm-01.na.infn.it	DPM
14	ULAKBIM-SE	torik1.ulakbim.gov.tr	DPM
15	IPHC-SE	sbgse1.in2p3.fr	DPM
16	CNAF-SE	ds-202-11-01.cr.cnaf.infn.it	STORM
17	ROMA3-SE	storm-01.roma3.infn.it	STORM
18	KEK-SE	Kek-se03.cc.kek.jp	STORM
19	McGill-SE	gridftp02.clumeq.mcgill.ca	STORM

Dynafed is a lightweight federation services able to aggregate multiple Http/WebDav/S3 endpoints showing a single namespace

A Dynafed server is running in Napoli aggregating 19 belle II storage endpoints
https://dynafed-belle.na.infn.it/myfed



Silvio Pardi

Studies at Napoli

DAVS protocol in a gbasf2 analysis

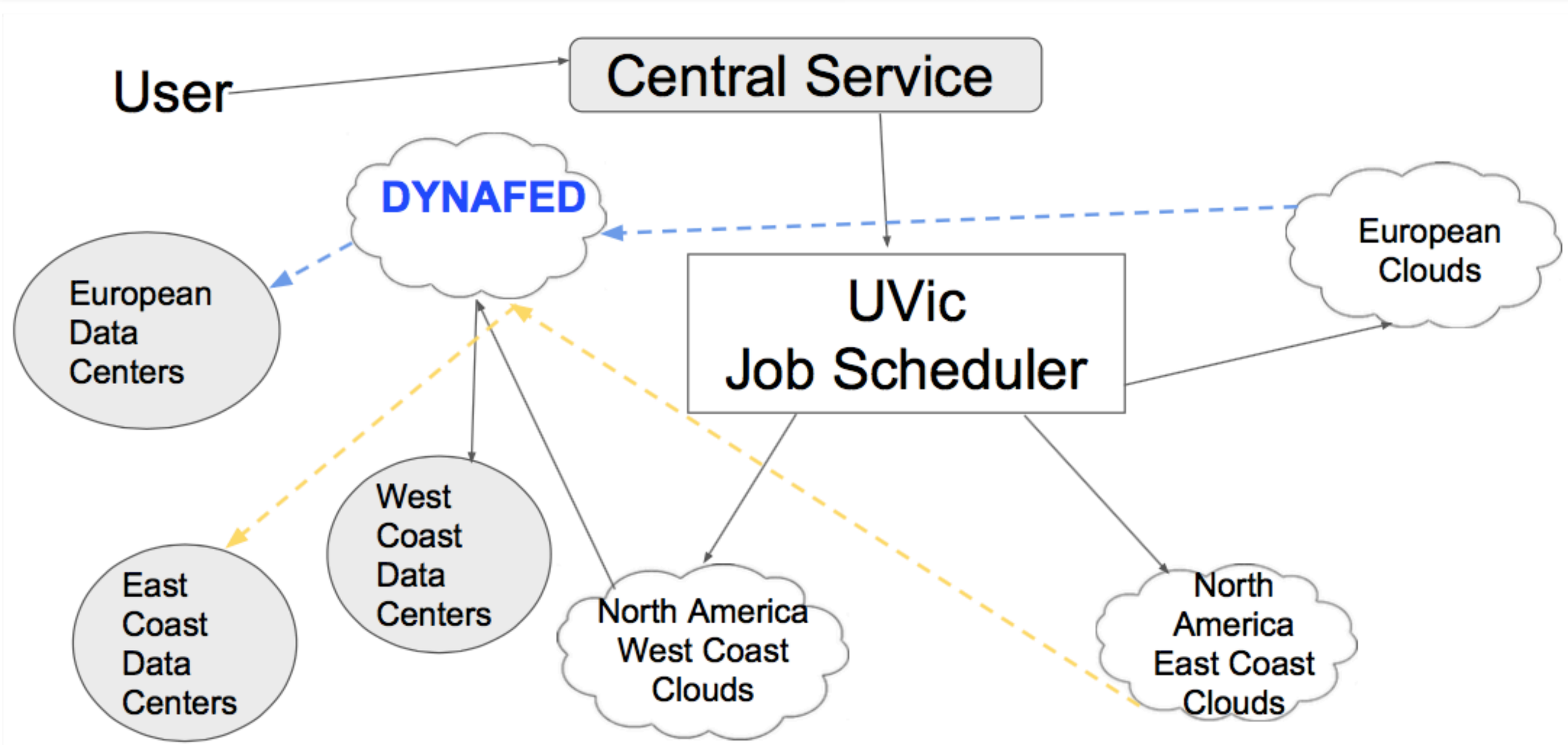
Ongoing test are focussed on three possible use-cases:

- DAVS protocol in DIRAC
- DAVS + Dynafed + DIRAC
- DAVS + Dynafed + DPM Volatile Pool (Cache) + DIRAC

All previous use-cases has been tested with success with local jobs using basf2 running in a user-interface.
Next stage is check the possibility to use in DIRAC with gbasf2

Silvio Pardi

Dynafed for “Cloud”



https://kds.kek.jp/indico/event/25459/session/5/contribution/50 M. Ebert

Victoria Dynafed for Belle-II

- Endpoints behind Belle-II Dynafed:
  - Compute Canada East (Minio via S3 API)
  - Compute Canada West (Minio via S3 API)
  - Amazon (S3)
  - Chameleon Cloud (Minio via S3 API)
  - Victoria Tier-2 SE (dCache, Victoria SE for Belle-II)
  - Victoria HEPRC Ceph (CephS3)

https://indico.cern.ch/event/651353/contributions/2993097/ F. Berghaus

Victoria Dynafed for Belle-II

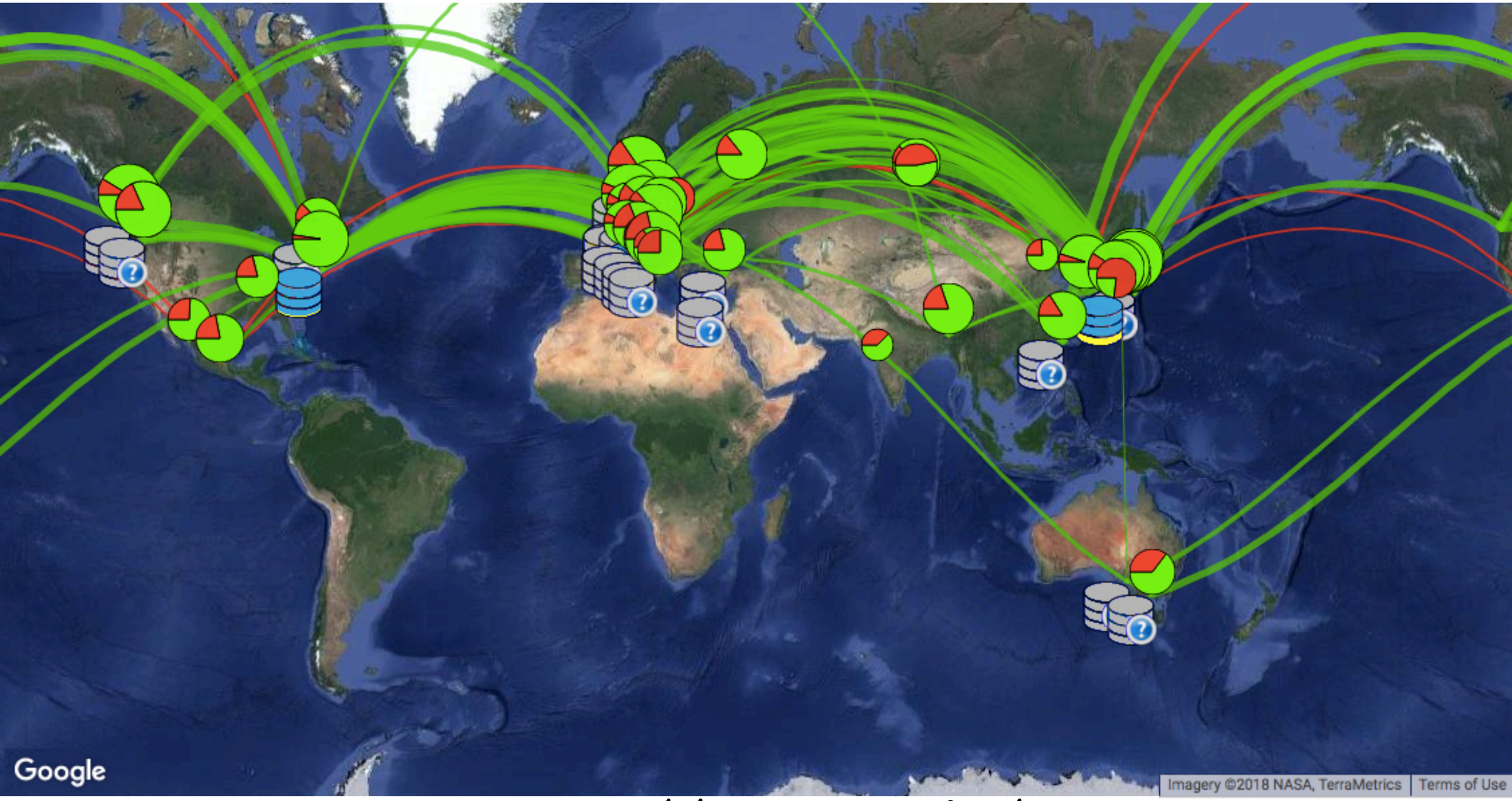
- Belle-II *is developing* gfal2 support for their DDM and WMS
  - Will allow direct usage of Dynafed as SE in the future
- Workaround:
  - Belle-II allows job configuration to access locally mounted volume
  - gfalFS provides fuse mount within Linux directory tree:

```
gfalFS -s ${HOME}/b2data/belle davs://dynafed02.heprc.uvic.ca:8443/belle
```

  - Jobs access Belle-II data from “local” directory ~/b2data/belle
- gfalFS only needs Dynafed as access point for any cloud
  - Dynafed redirects gfalFS to closest endpoint via GeoIP

https://indico.cern.ch/event/651353/contributions/2993097/ F. Berghaus

Outreach



http://belle2.jp/

Rucio?

Some gentleman mentioned about Belle II and Rucio at a BiLD meeing?

Rucio?

Belle II

- No official decisions made for Rucio
  - We have been wondering whether it is usable for us with our BelleDIRAC
  - I have unofficially asked some people for comments
  - We have our Belle II DDM, working and used in production, and planned to evolve further

Background

- PNNL
  - was responsible for the development of Belle II DDM
  - The responsibility has been moved to BNL… (by DOE)
- BNL
  - New to Belle II. Still learning about Belle II computing and DDM
  - They will establish Rucio as a service for the experiments <u>they host</u>

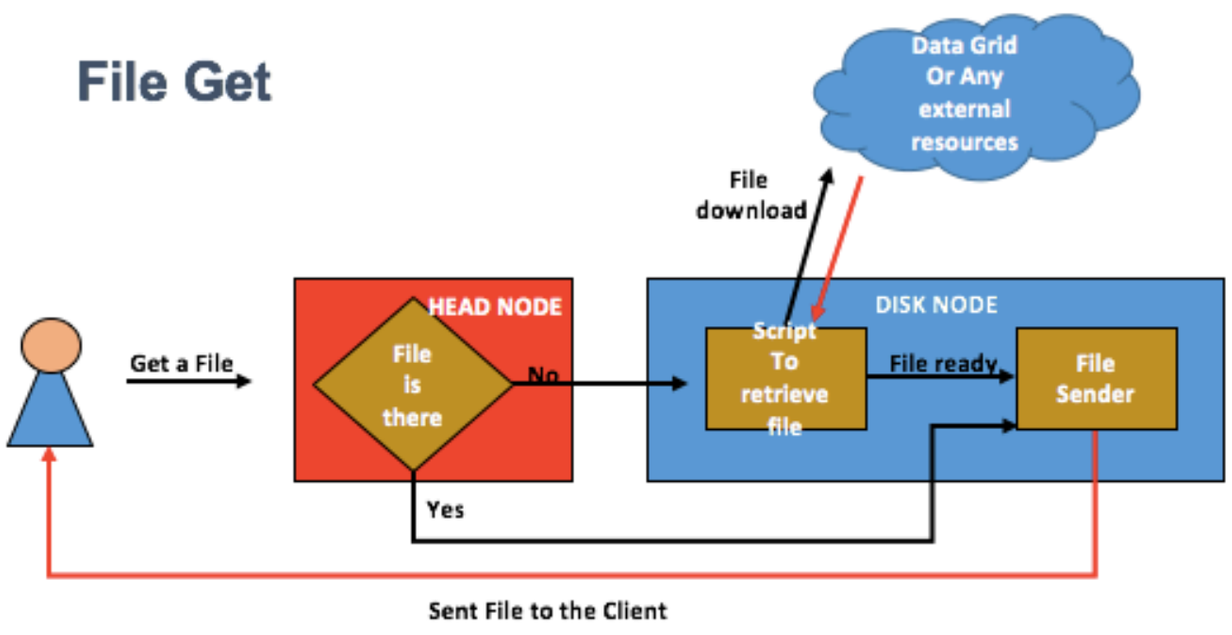
backup

Concept of Volatile Pool

A Pool in DPM is a collection of File Systems managed as a single storage area.

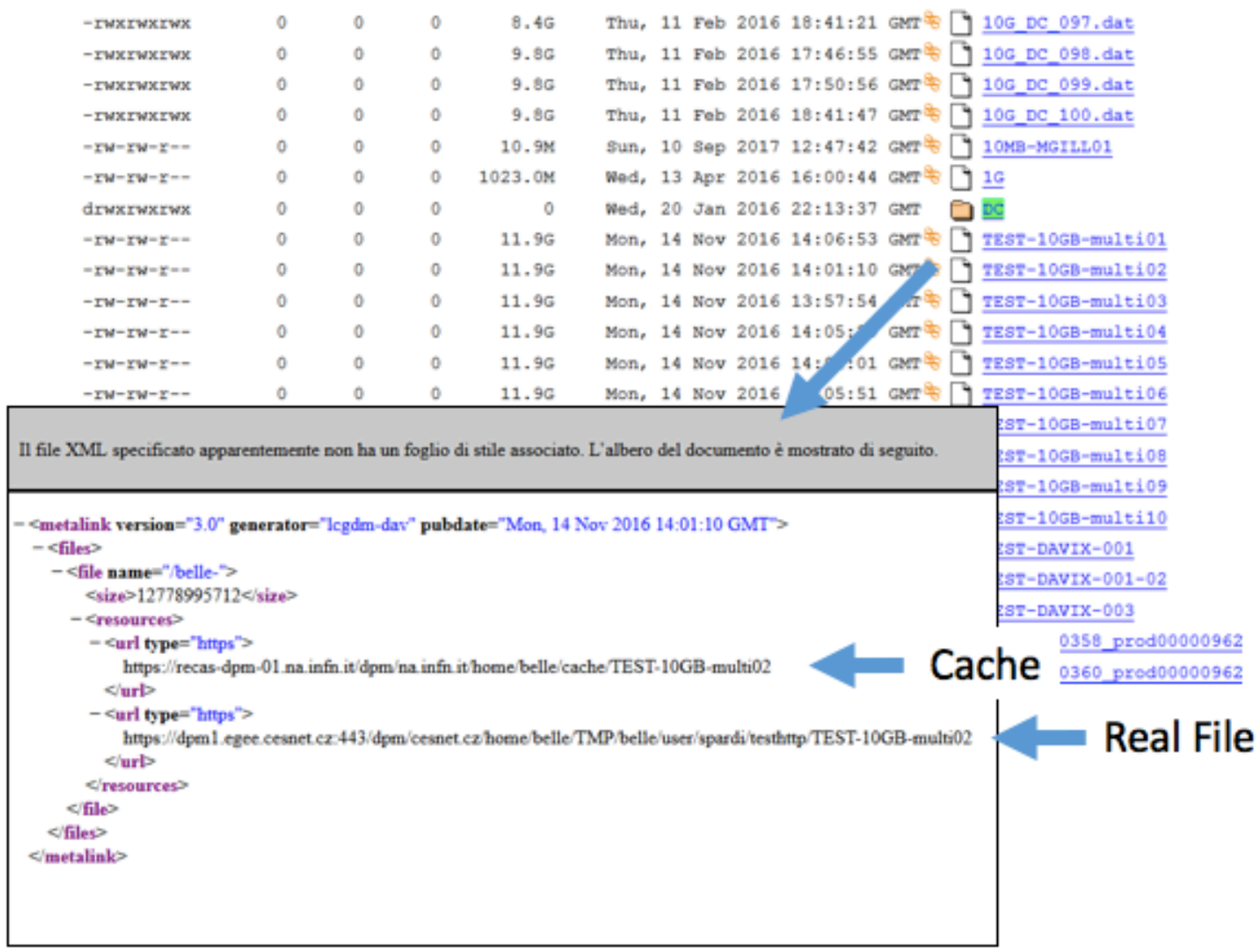
A **Volatile Pool** is a special pool that can pull files from external sources.

When an **User get a file** from the Volatile pool, the **Disk Node** providing the file system of the pool, send the file to the client if ready, otherwise a script is locally run in order to retrieve the file from some external source.



Silvio Pardi

Dynafed + Volatile Pool



What happen if we aggregate a Webdav endpoint with a DPM Volatile Pool?

When Dynafed stat files inside the real webdav endpoint, it receive always a reply even from the Volatile Pool.

So that the metalink representing a file in Dynafed, included always the real URL and the corresponding virtual copy in the cache (even if the latter does not exist yet)

Moreover thanks to the GeoPlugin, Dynafed prioritize the cache copy if the Volatile Pool is local to the Client or close to it.

This combination allow to create a cache system

Silvio Pardi