Distributed Data Management & MC production system in CMS

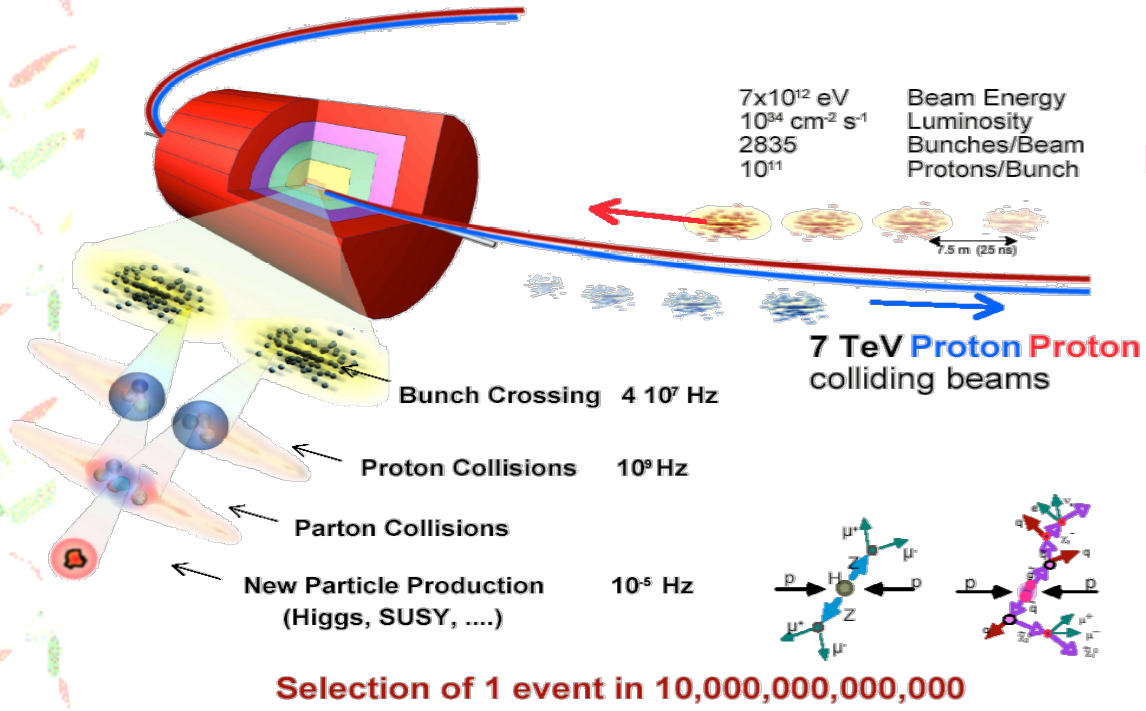
Outline

- Introduction
- Computing model overview
- Production system
  - Processing workflow
- Available resources@ CCIN2P3

Farida Fassi

Introduction

- Monte Carlo (MC) production is crucial for detector studies and physics analysis
- Event simulation and reconstruction typically done in computer farms of a large amount of computing, storage and network resources

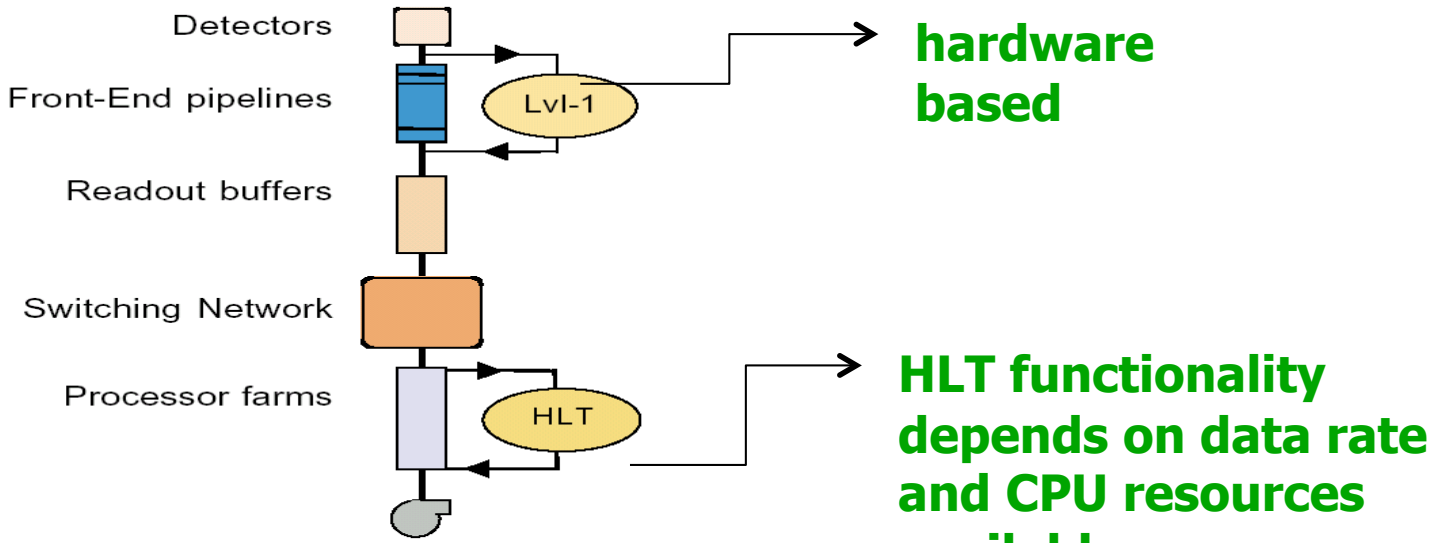


- The LHC collisions will occur @ 40 MHz, while the offline system can stream data to disk only at 150-300 Hz

CMS Trigger Strategy

- CMS has chosen a trigger sequence in which, after a L1 response,
  - reducing the events from 40 MHz to 100 kHz,
  - the offline reconstruction code runs to provide

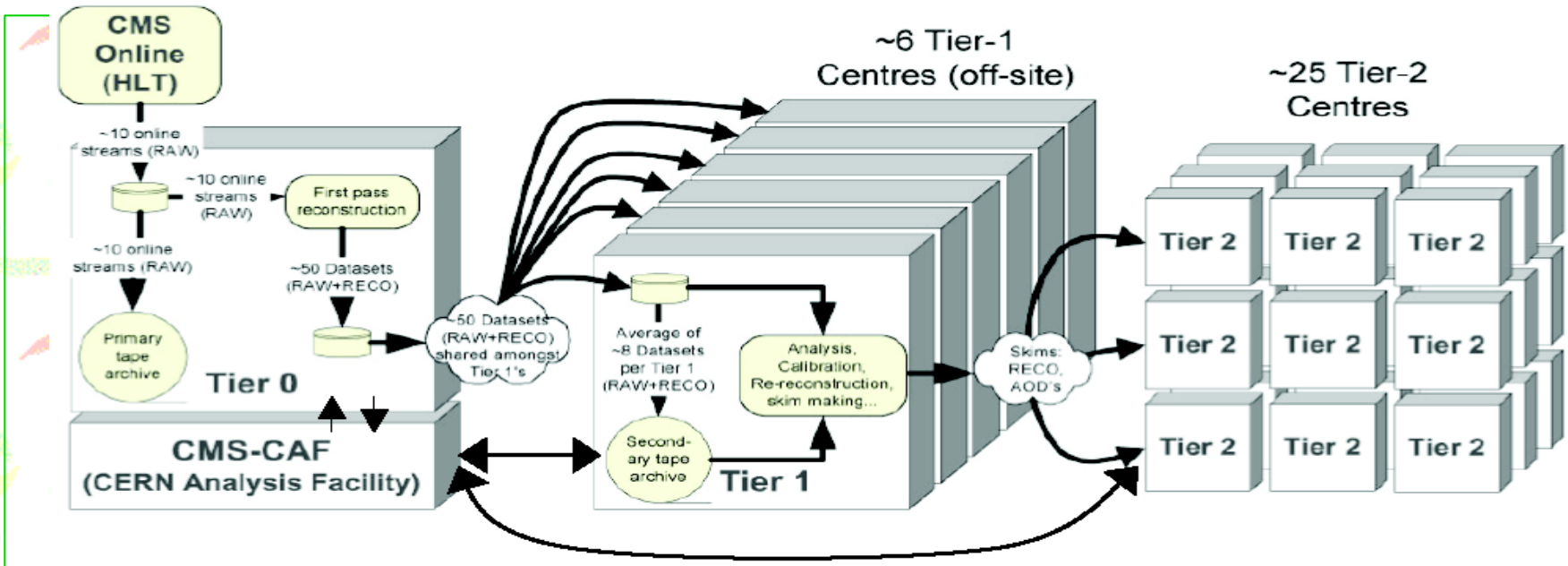
  the factor 1000 reduction to 150-300 Hz



CMS Computing model

- Aim to Cope with computing requirements for storage, processing and analysis of data
- Distributed model for all computing including the serving and archiving of the RAW and RECO data
- **Streaming**
- Classifying events early allow data access optimization
- RAW events are classified in primary datasets depending on their trigger history
- Propose O(50) 'primary datasets' are grouped into O(10) online streams
- O(2PB)/yr raw data split into O(50) (40 TB) trigger-determined datasets

Tiered Architecture



Tier-0:

- Accepts data from DAQ
- Prompt reconstruction
- Data archive and distribution to Tier-1's

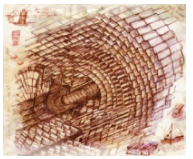
Tier-1's:

- Real data archiving
- Re-processing
- Skimming and other data-intensive analysis tasks
- Calibration
- MC data archiving

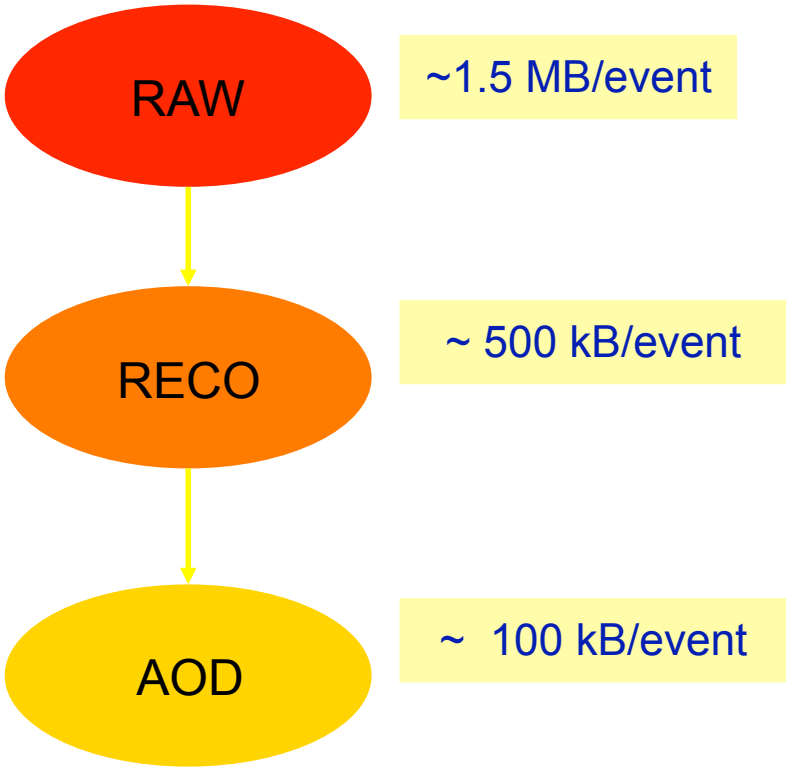
Tier-2's:

- User data Analysis
- MC production
- Import skimmed datasets from Tier-1 and export MC data
- Calibration/alignment

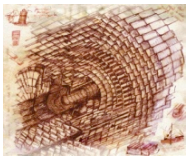
CMS Data Tiers



- CMS plans to implement a hierarchy of Data Tiers
  - Raw Data: as from the Detector
  - Full Event: contains Raw plus all the objects created by the Reconstruction pass
  - RECO: contains a subset of the Full Event, sufficient for reapplying calibrations after reprocessing
    - "Refitting but not re-tracking"
  - AOD: a subset of RECO, sufficient for the large majority of "standard" physics analyses
    - Contains tracks, vertices etc and in general enough info to (for example) apply a different b-tagging
    - Can contain very partial hit level information

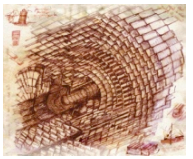


Data organization



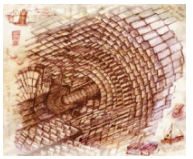
- CMS expects to produce large amounts of data (events)
  - O(PB)/year
- Event data are in files
  - average file size is kept reasonably large (≥ GB)
    - avoid scaling issues with storage systems, catalogues when dealing with too many small files
    - foresee file merging
  - $O(10^6)$ files/year
- Files are grouped in Fileblocks
  - group files in blocks (1-10TB) for bulk data management reasons
  - 10^3 Fileblocks/year
- Fileblocks are grouped in Datasets
  - Datasets are large (100TB) or small (0.1TB) : size driven by physics

Data tiers production

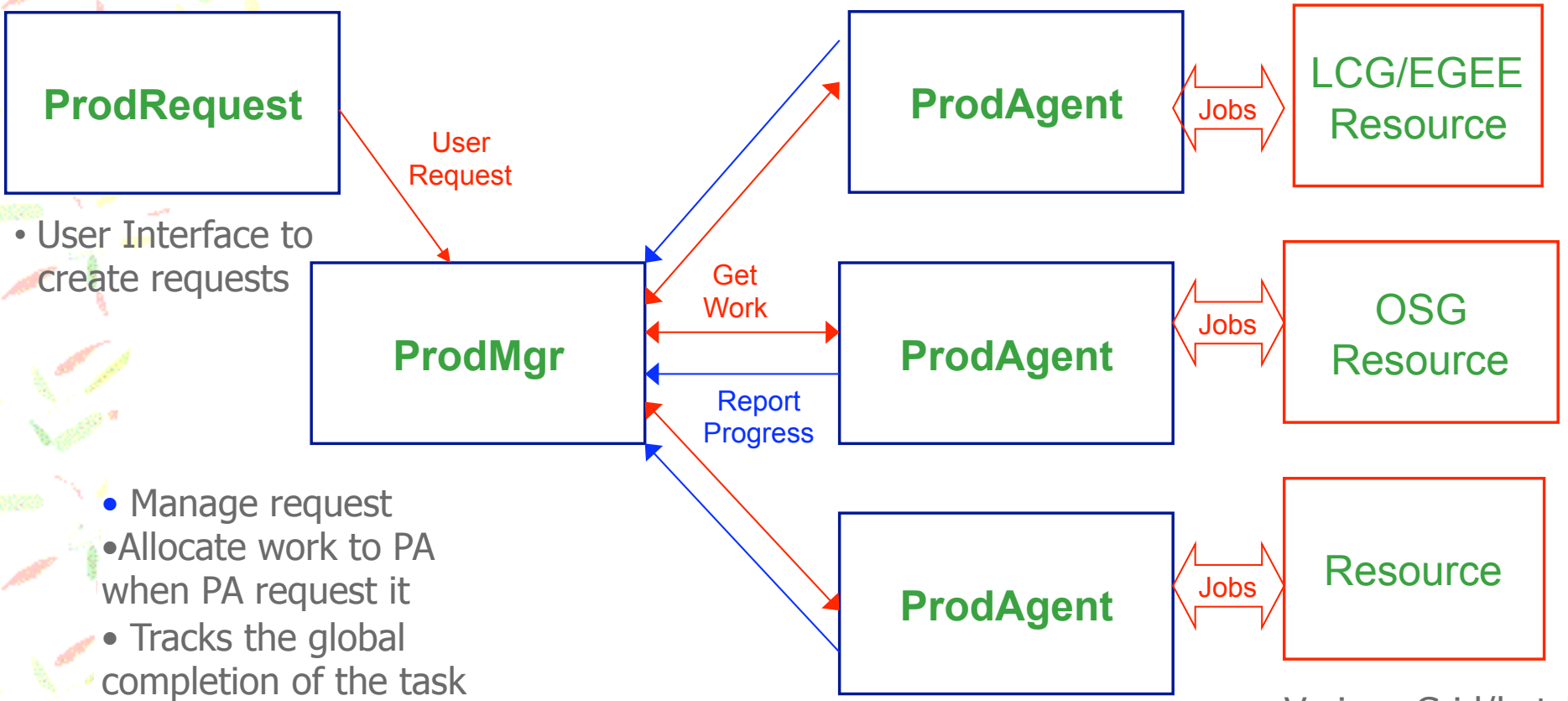


- Generation:
  - no input, small output (10 to 50 MB ntuples)
  - pure CPU: few minutes, up to few hours if hard filtering present
- Simulation (hits): GEANT4
  - small input
  - CPU and memory intensive: 24 to 48 hours
  - large output: ~500 MB in three files (EVD files), the smallest is ~ 100 KB !
- Digitization:
  - lower CPU/memory requirements: 5 to 10 hours
  - I/O intensive: persistent reading of PU through LAN
  - large output: similar to simulation
- Reconstruction:
  - even less CPU: ~5 hours
  - smaller output: ~200 MB in two files

Production System Overview



Aim at automating as much as possible, easy maintenance



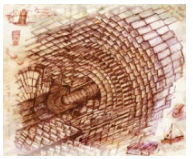
- User Interface to create requests

- Manage request
- Allocate work to PA when PA request it
- Tracks the global completion of the task

- Ask for work
- Convert work into processing jobs
- Create, submit, track jobs
- Manage the merge, failures, resubmit, local cataloguing, etc..

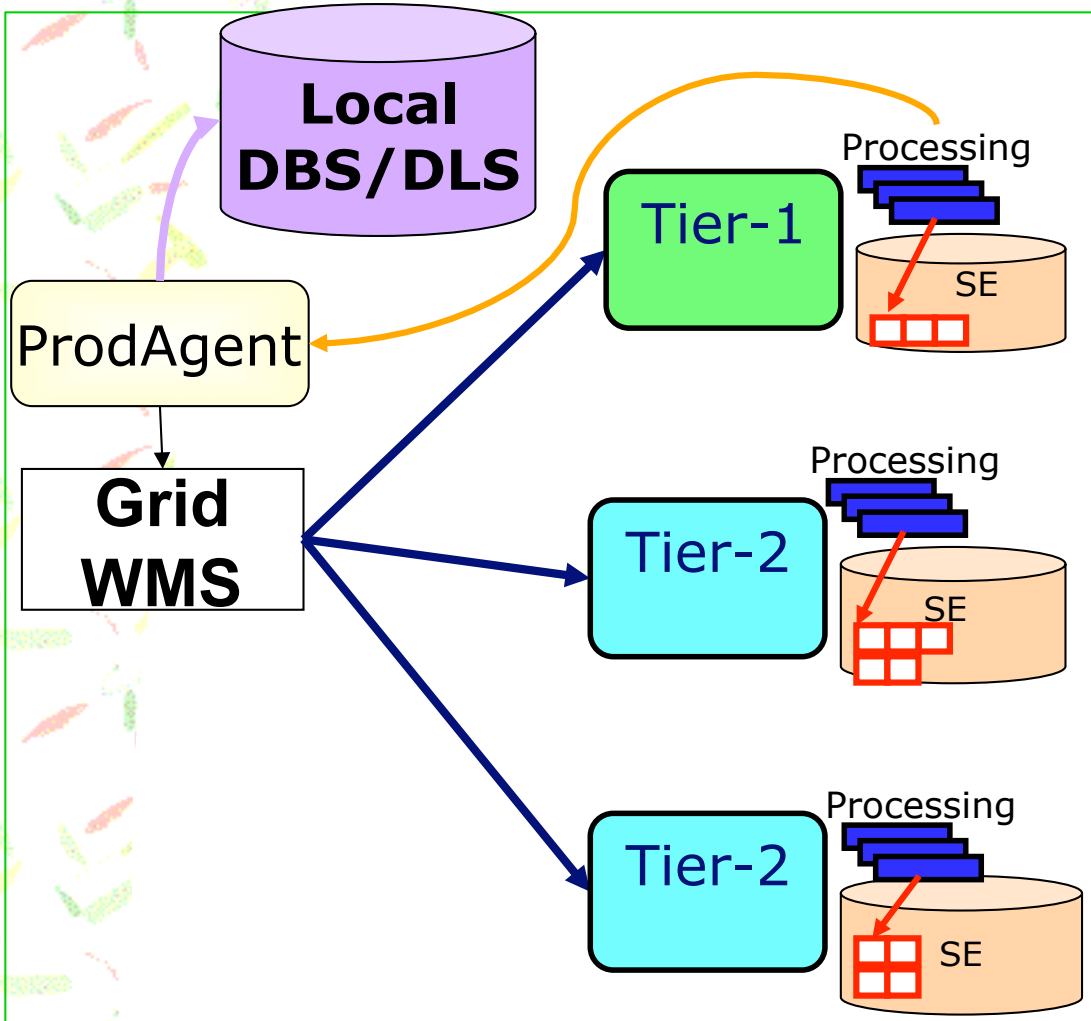
- Various Grid/batch Middleware to support

ProdAgent Workflow



- Send Processing jobs to sites
- Drop of data at the sites
- Report back to ProdAgent
- Merge data at site
- Catalog in Data Management catalog
- Transfer data
- Automate retries on errors

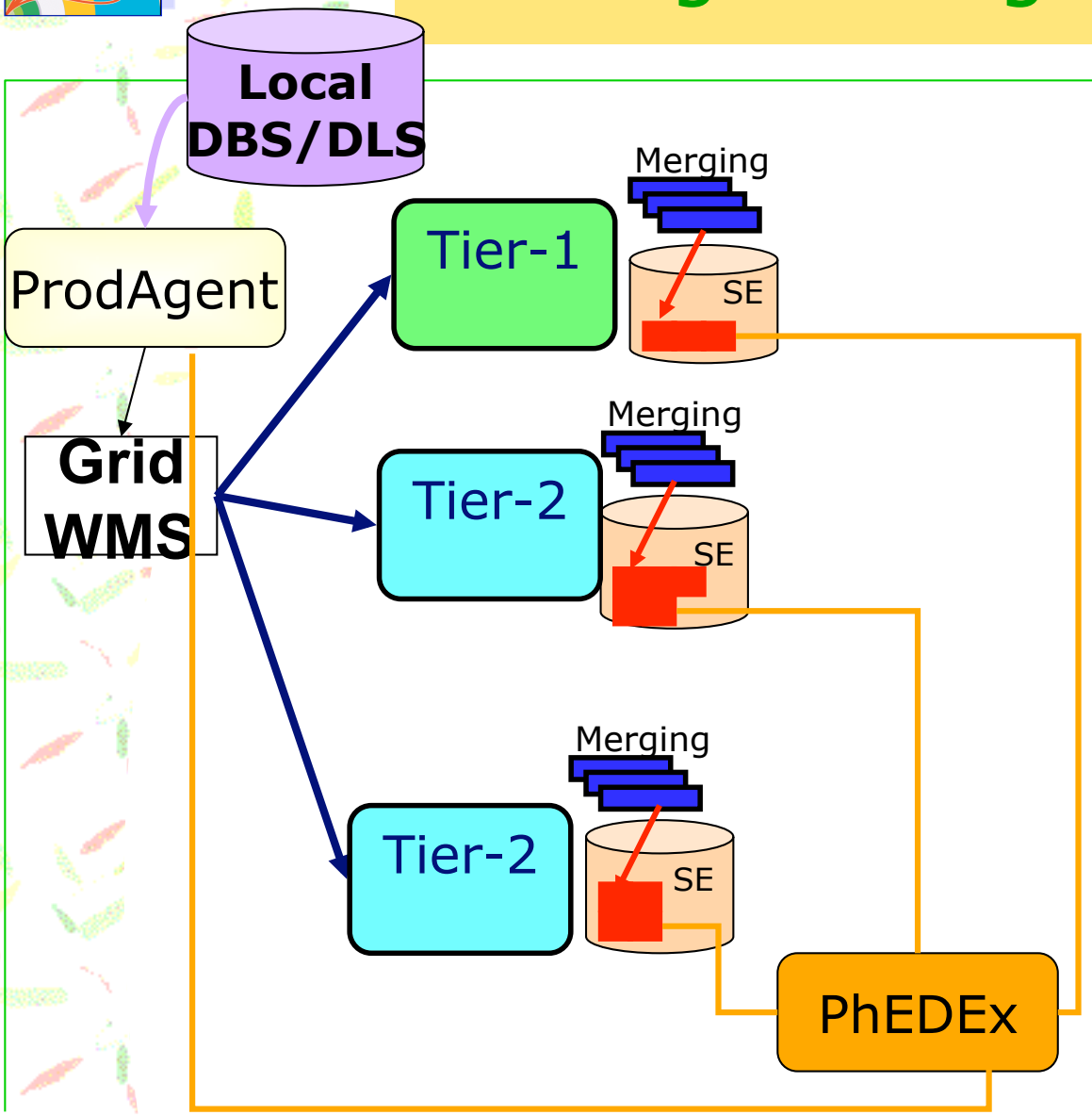
ProdAgent Processing Workflow



- Processing jobs sent to sites
- Output data left in local SE
- Report back to ProdAgent
- Data management cataloguing (registration in local DBS/DLS)
- Failed jobs handled automatically

- Small output file from Processing job

ProdAgent Merge Workflow

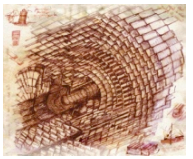


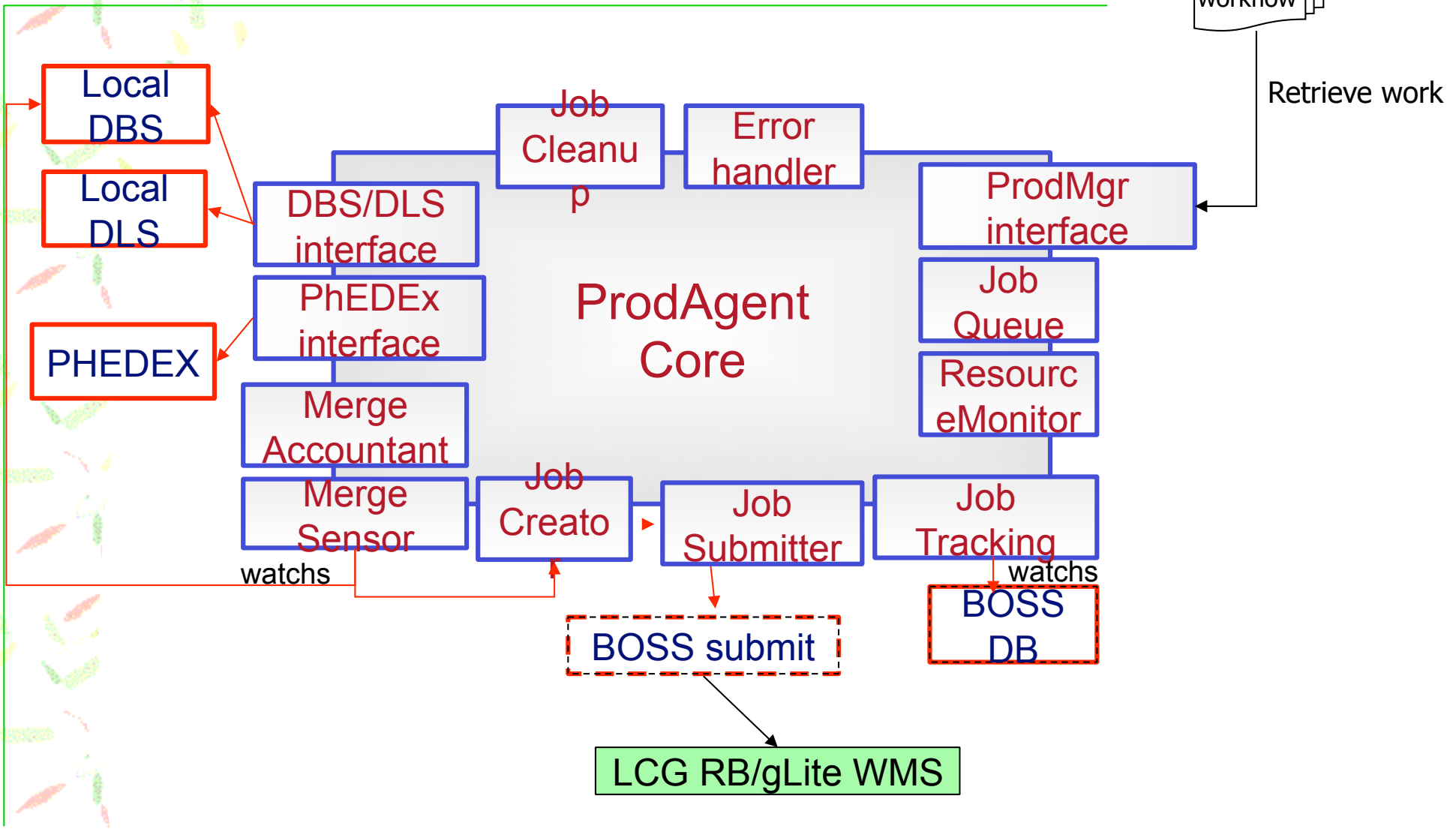
- Merge data at site
  - Watch DBS/DLS for produced unmerged data
  - send merge job at sites hosting data
- Transfer data
  - PhEDEx injection

Large output file from Merge job

PhEDEx transfer invoked by PA

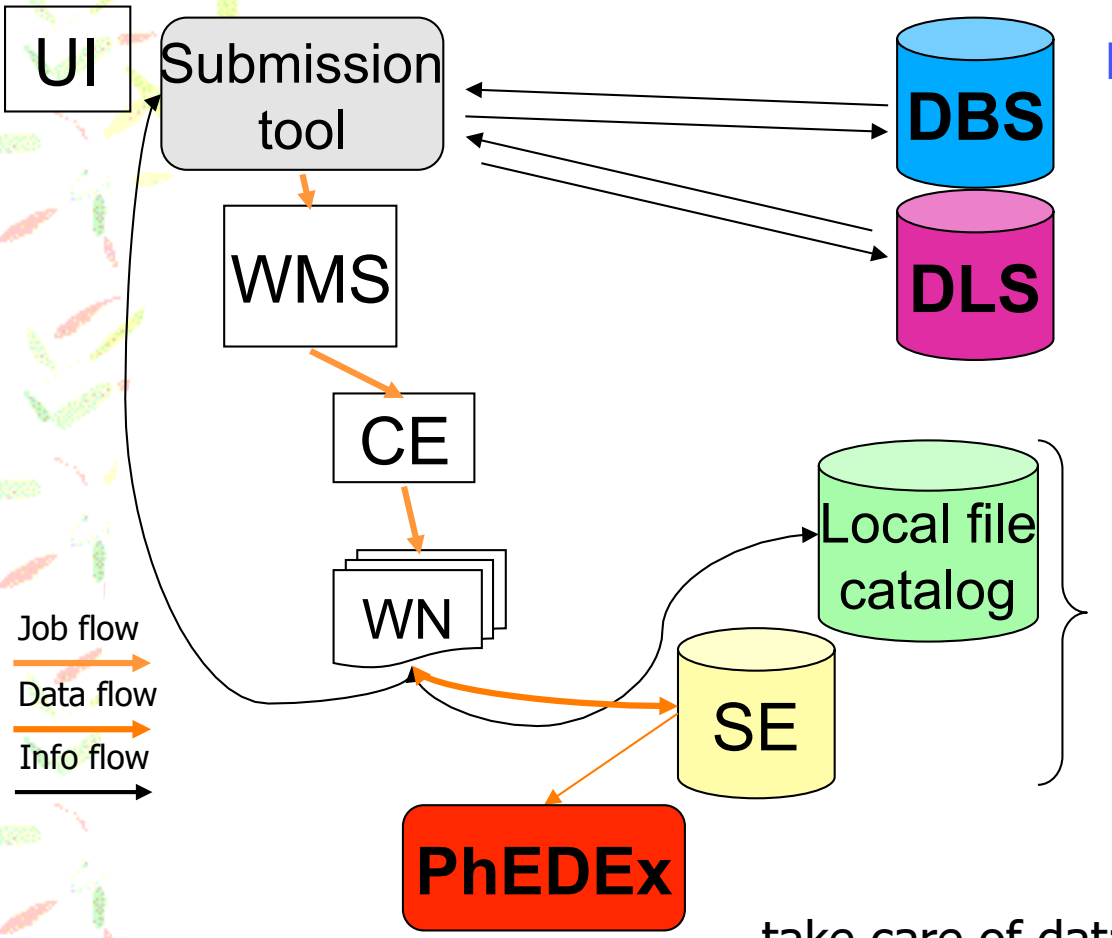
Production Agent components





Data processing workflow

- Data Management System allow to discover, access and transfer event data in a distributed computing environment



Dataset Bookeeping System:
What data exists?

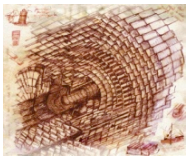
Data Location Service:
Where does data exists?

Site Local catalog:
physical location of files at the site

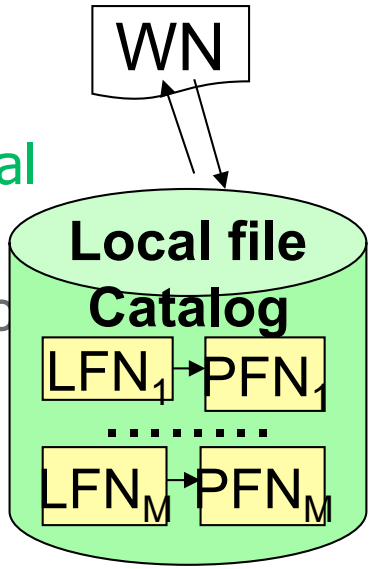
Storage System

PhEDEx:
take care of data transfer to reach destination site

Local data access



- CMS application read and write files at a site
- DLS has names of sites hosting the data and not the physical location of constituent files at the sites
- Local file catalogues provides site local information about ho to access any given logical file name
  - Baseline is to use a "Trivial File catalogue (TFC)"
  - Need to sustain very high-scale performance

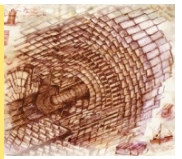


- Site local discovery mechanism : discover at runtime on WN the site-dependent job configuration to access data
- CMS application interface to storage with a POSIX-like interface (dcap, rfio. etc)
- Storage System with SRM (disk, mass storage)

CCIN2P3:TFC

```
<!-- production stage out : unmerged -->
  <lfn-to-pfn protocol="direct" path-match="/+(store/unmerged/.*)"
    result="/pnfs/in2p3.fr/data/cms/prod/$1"/>
  <pfn-to-lfn protocol="direct" path-match="/pnfs/in2p3.fr/data/cms/prod/+(store/unmerged/.*)"
    result="/$1"/>
<!-- LoadTest transfers -->
  <lfn-to-pfn protocol="direct" path-match="/+(LoadTest/.*)"
    result="/pnfs/in2p3.fr/data/cms/import/$1"/>
  <pfn-to-lfn protocol="direct" path-match="/pnfs/in2p3.fr/data/cms/import/+(LoadTest.*)"
    result="/$1"/>
<!-- jobs access protocol - default -->
  <lfn-to-pfn protocol="jobs" chain="dcap" path-match="(.*)"
    result="$1" />
  <pfn-to-lfn protocol="jobs" chain="dcap" path-match="(.*)"
    result="$1" />
<!-- default - production and Protocol chains -->
  <lfn-to-pfn protocol="direct" path-match="/+(.*)"
    result="/pnfs/in2p3.fr/data/cms/data/$1"/>
  <lfn-to-pfn protocol="srm" chain="direct" path-match="/+(.*)"
    result="srm://ccsrm.in2p3.fr:8443/srm/managerv1?SFN=/$1" />
  <lfn-to-pfn protocol="srmv2" chain="direct" path-match="/+(.*)"
    result="srm://ccsrm.in2p3.fr:8443/srm/managerv2?SFN=/$1" />
  <lfn-to-pfn protocol="dcap" chain="direct" path-match="/+(.*)"
    result="dcap://ccdcapcms.in2p3.fr:22125/$1" />
  <lfn-to-pfn protocol="root" chain="direct" path-match="/+(.*)"
    result="root://ccxroot.in2p3.fr:1094//$1" />
  <pfn-to-lfn protocol="direct" path-match="/pnfs/in2p3.fr/data/cms/data/+(.*)"
    result="/$1" />
  <pfn-to-lfn protocol="srm" chain="direct" path-match=".*\?SFN=(.*)"
    result="$1" />
  <pfn-to-lfn protocol="srmv2" chain="direct" path-match=".*\?SFN=(.*)"
    result="$1" />
  <pfn-to-lfn protocol="dcap" chain="direct" path-match="dcap://ccdcapcms(.*)"
    result="$1" />
  <pfn-to-lfn protocol="root" chain="direct" path-match="root://ccxroot.in2p3.fr:1094/(.*)"
    result="$1" />
</storage-mapping>
```

CCIN2P3-Job-config file



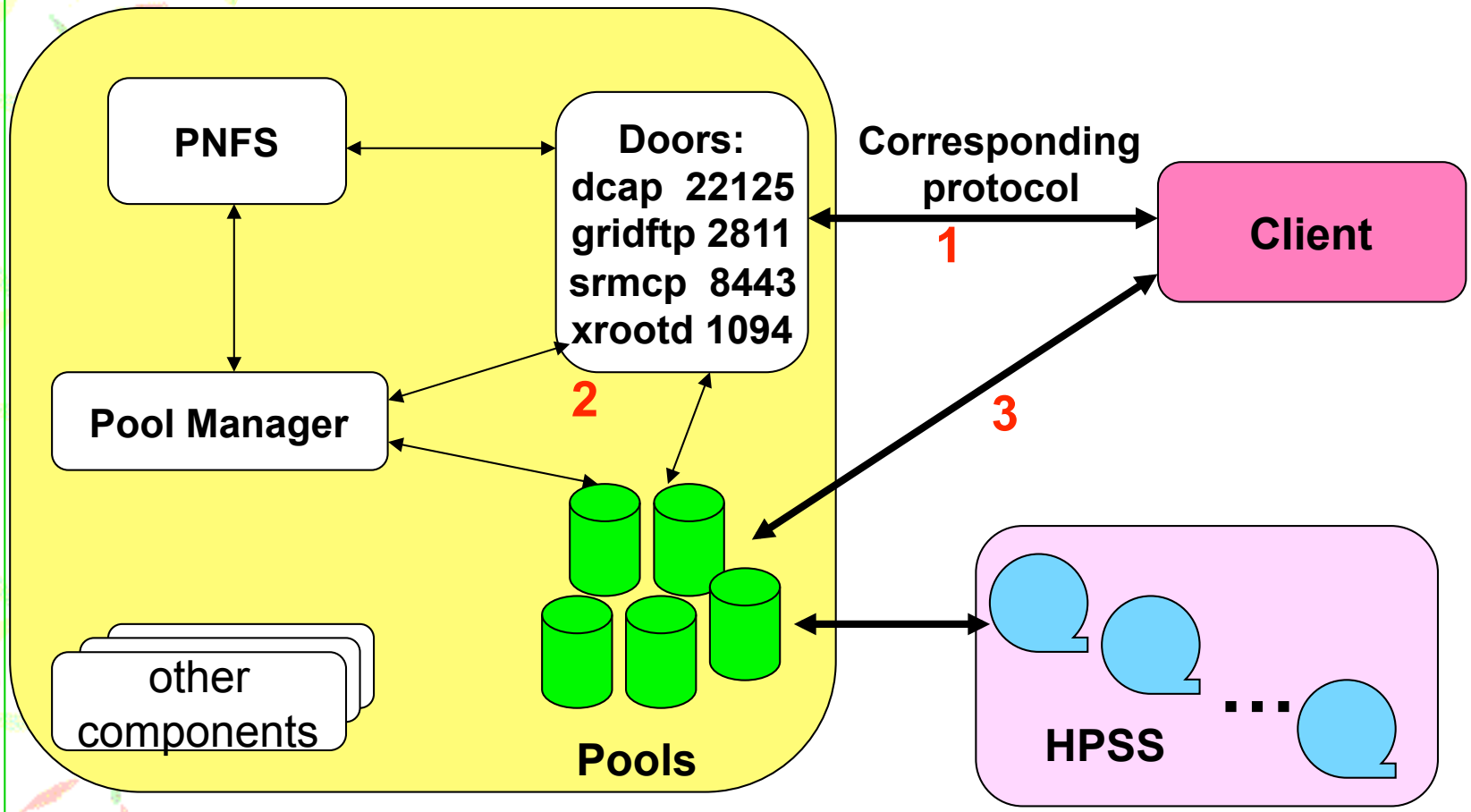
```
<site-local-config>
 <site name="T1_FR_CCIN2P3">
    <event-data>
      <catalog url="trivialcatalog_file:/afs/in2p3.fr/grid/toolkit/cms2/SITECONF/local/PhEDEx/storage.xml?protocol=dcap"/>
    </event-data>
    <local-stage-out>
      <command value="srm"/>
      <option value="-debug"/>
      <catalog
url="trivialcatalog_file:/afs/in2p3.fr/grid/to      ms2/SITECONF/local/PhEDEx/storage.xml?protocol=srm"/>
      <se-name value="ccsrm.in2p3.fr"/>
    </local-stage-out>
    <calib-data>
      <frontier-connect>
       <load balance="proxies"/>
        <proxy url="http://cclcgcms03.in2p3.fr:3128"/>
        <proxy url="http://cclcgcms01.in2p3.fr:3128"/>
        <server url="http://cmsfrontier.cern.ch:8000/FrontierInt"/>
        <server url="http://cmsfrontier1.cern.ch:8000/FrontierInt"/>
        <server url="http://cmsfrontier2.cern.ch:8000/FrontierInt"/>
        <server url="http://cmsfrontier3.cern.ch:8000/FrontierInt"/>
      </frontier-connect>
    </calib-data>
 </site>
</site-local-config>
```

Stage-out

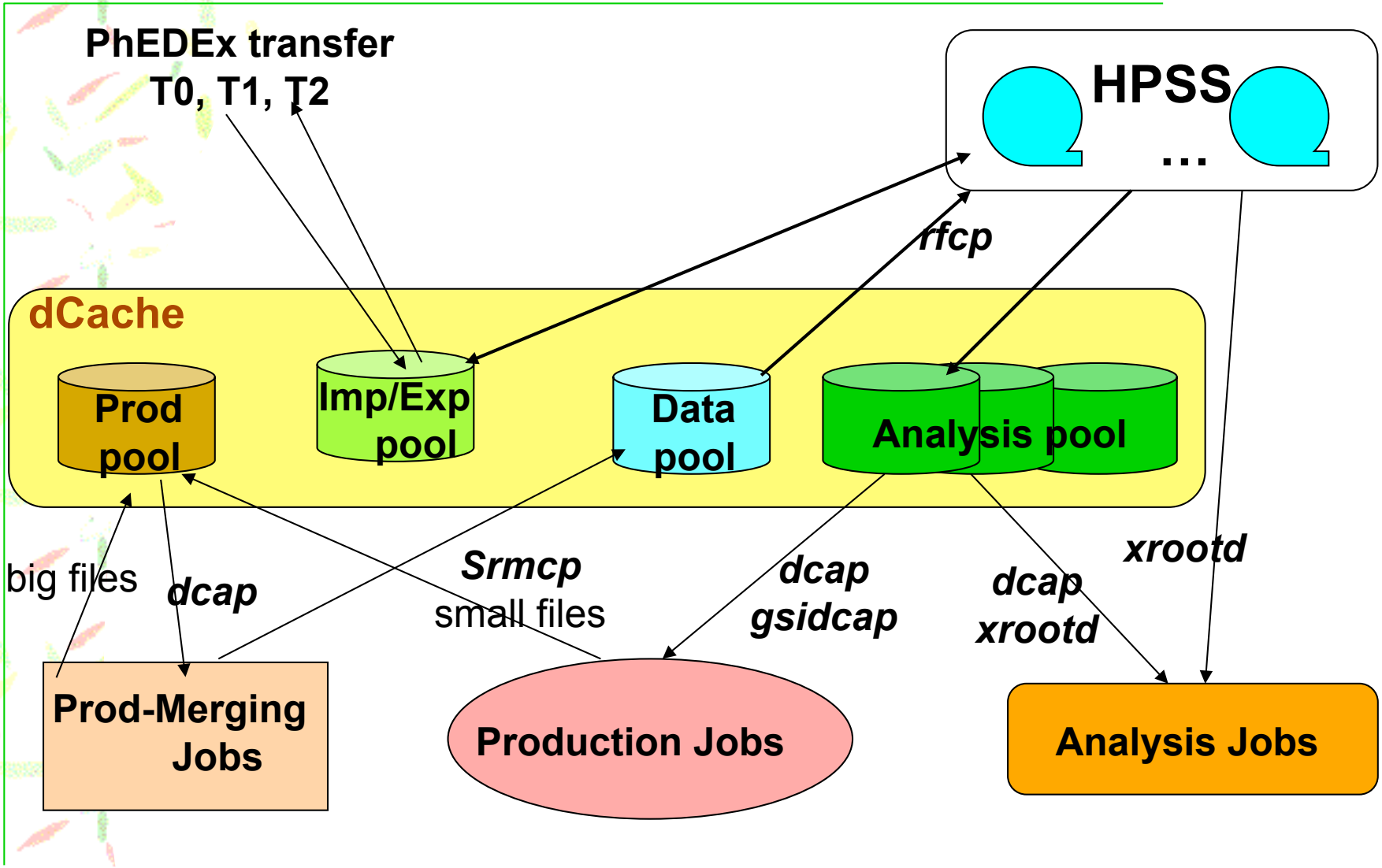
Squid

SE resources @CCIN2P3 (1)

dCache managed by a Storage Resource Manager (SRM)

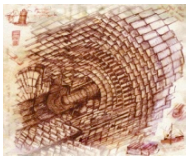


SE resources @CCIN2P3 (2)



backup

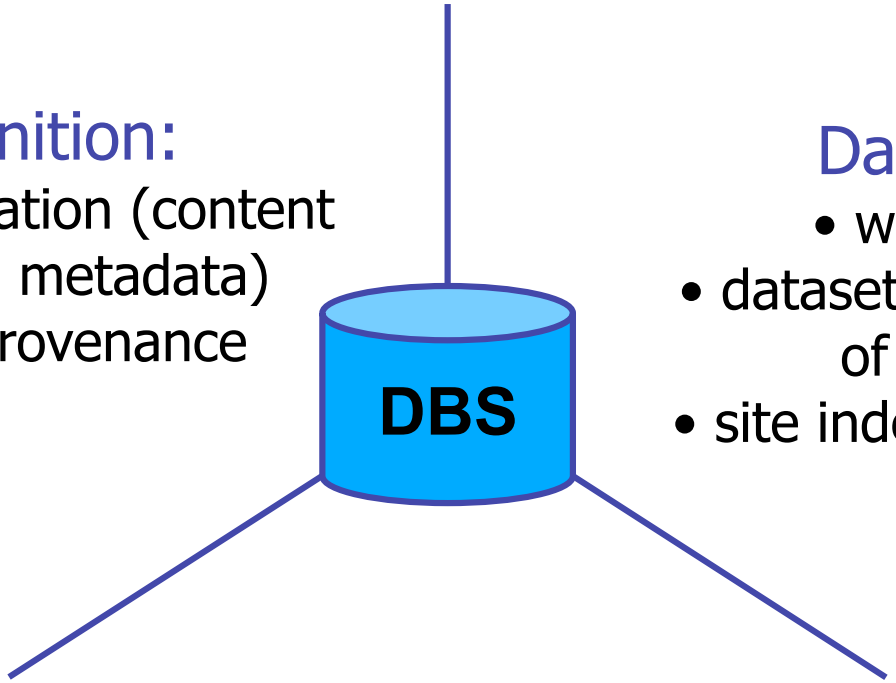
Dataset Bookkeeping System (DBS)



- The Dataset Bookkeeping System (DBS) provides the means to define, discover and use CMS event data

Data definition:

- dataset specification (content and associated metadata)
- track data provenance



Data discovery:

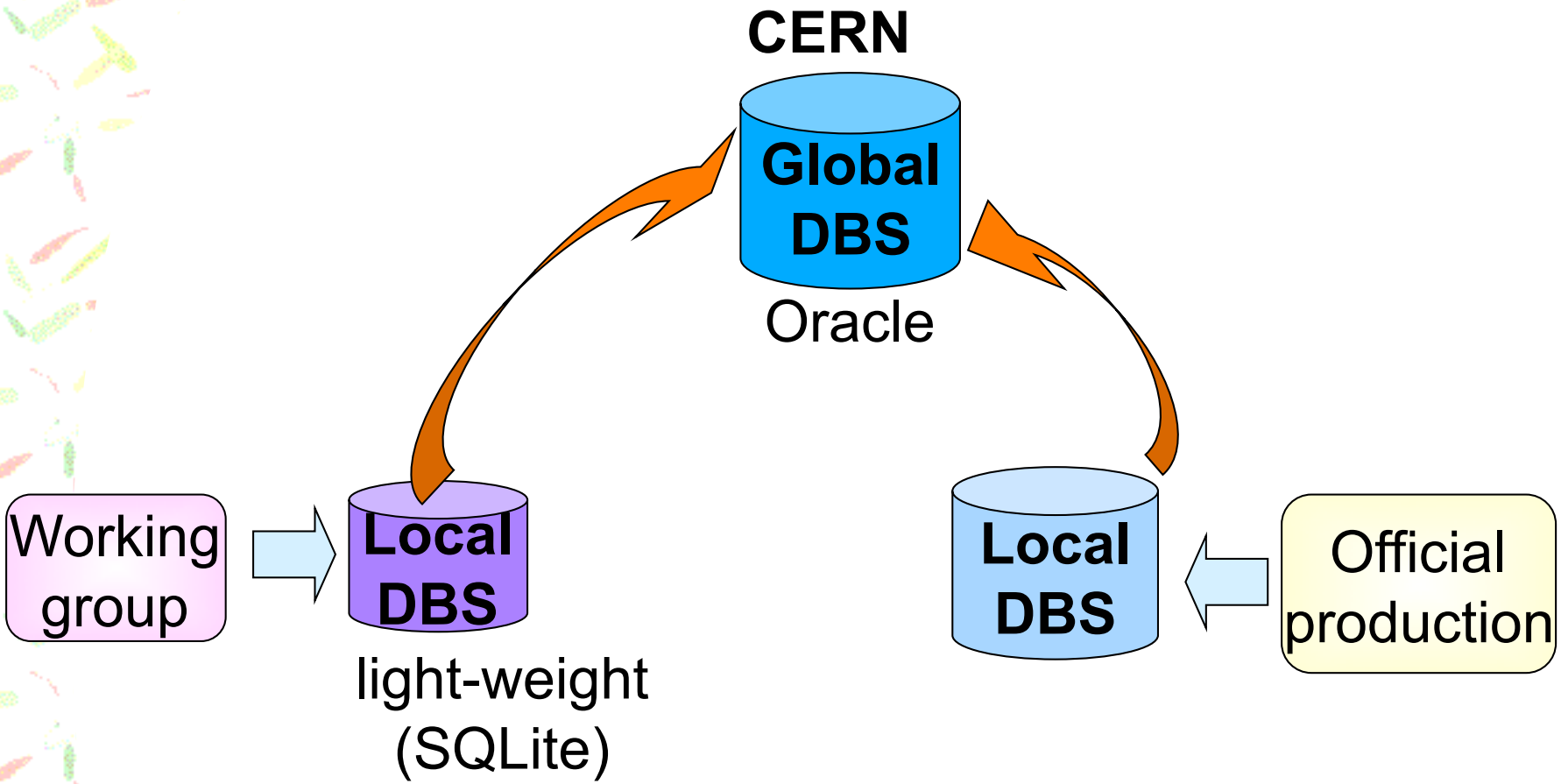
- which data exists
- dataset organization in term of files/fileblocks
- site independent information

Use:

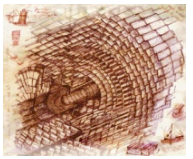
- Distributed analysis tool (CRAB)
- MC Production system

DBS scopes and dynamics

- A DBS instance describing data CMS-wide (Global scope)
- DBS instances with more "local" scope



Data Location Service (DLS)



- The Data Location Service (DLS) provides the means to locate replicas of data in the distributed computing system
  - it maps file-blocks to storage elements (SE's) where they are located
  - few attributes (*custodial* replica = considered a permanent copy at a site)
  - very generic: is not CMS-specific
  - global/local scopes