CMS: Workload management and CRAB

Stefano Lacaprara

Department of Physics
INFN and University of Padova

ARDA Workshop, 7/8 march 2005

Outline

Outline

Outline

Outline

Workload Management in CMS.

- Baseline solution for CMS
- Use (sometime abuse) only a fraction of Grid (LCG2) functionalities
- Does not even try to solve the *general* problem but focus on specific use case, the most common for CMS user
- Access to distributed data with batch jobs using CMS application
- Actual architecture based on following assumption:

- Data is already located on remote sites
- Local Pool catalogs available in remote sites
- CMS wm deployed and available on remote sites

- Simplify a lot Data Management!
- Data distributed on Dataset basis
- Dataset is atomic: complete and unbreakable
- Each dataset has different *data tiers*: Hit, Digis, DST, ...
- Each considered independently
- User input data is a dataset with given data tier(s)

Data discovery and location based on CMS specific services: RefDB and PubDB

- RefDB: central database knows of all produced datasets
- PubDB: remote database (one per site publishing data) contains local information about dataset access, including CE and local file catalog location
- RefDB knows about PubDB(s) publishing given dataset

CRAB CMS Remote Analysis Builder.

- CMS specific tool for Workload Management
- Perform all needed task to actual run user code on Grid environment
- User friendly interface for CMS user to grid services
- User is supposed to be able to develop and run her analysis code interactively

Gives directive to CRAB via configuration files:

- Dataset/Owner she want to access
- Type of data–tiers she needs (DST, Digis, . . .)
- Job splitting directives (# event per jobs)
- Name of Executable
- Configuration `.orcarc` cards: the one she uses locally!

CRAB CMS Remote Analysis Builder.

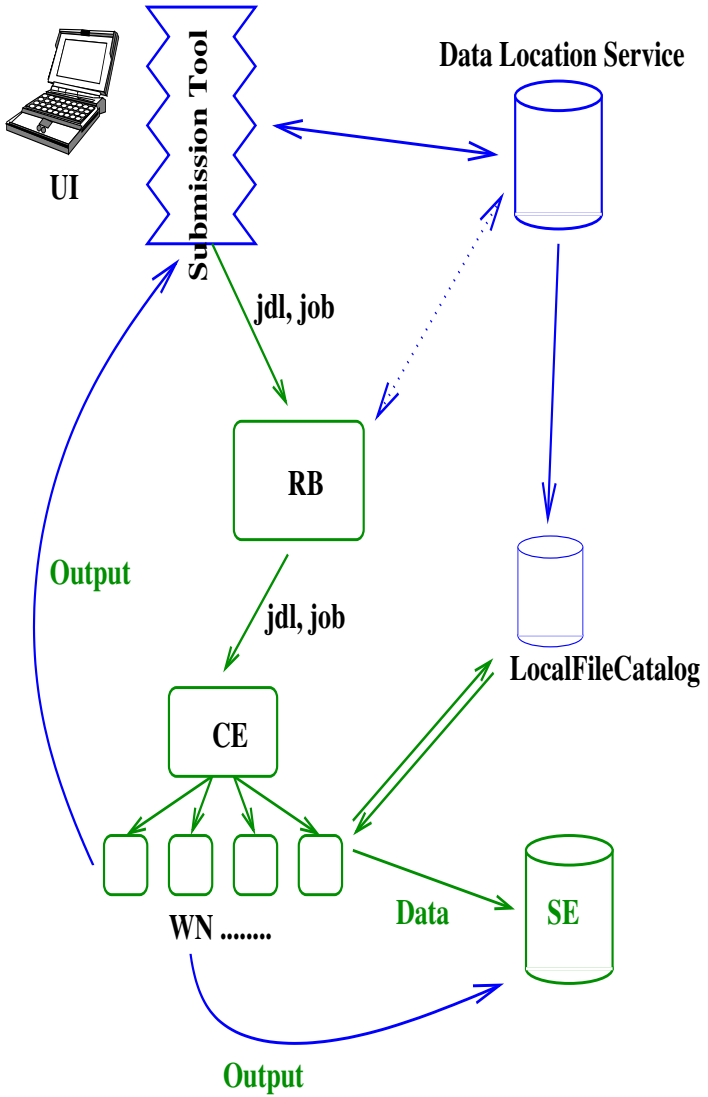
CRAB functionalities

- User job preparation: pack user private libraries and executable, prepare jdls, wrappers, etc . . .
- Dataset discovery and location
- Job splitting
- Changes of ORCA configuration files to run on remote site (including catalogs, splitting, etc . . .)
- Job submission, tracking
- Simple monitoring
- Automatic output retrieval at the end
- Or save it to SE or `gsiftp` server (*e.g.* `castor`!)
- Grid details are hidden to user

CRAB Status.

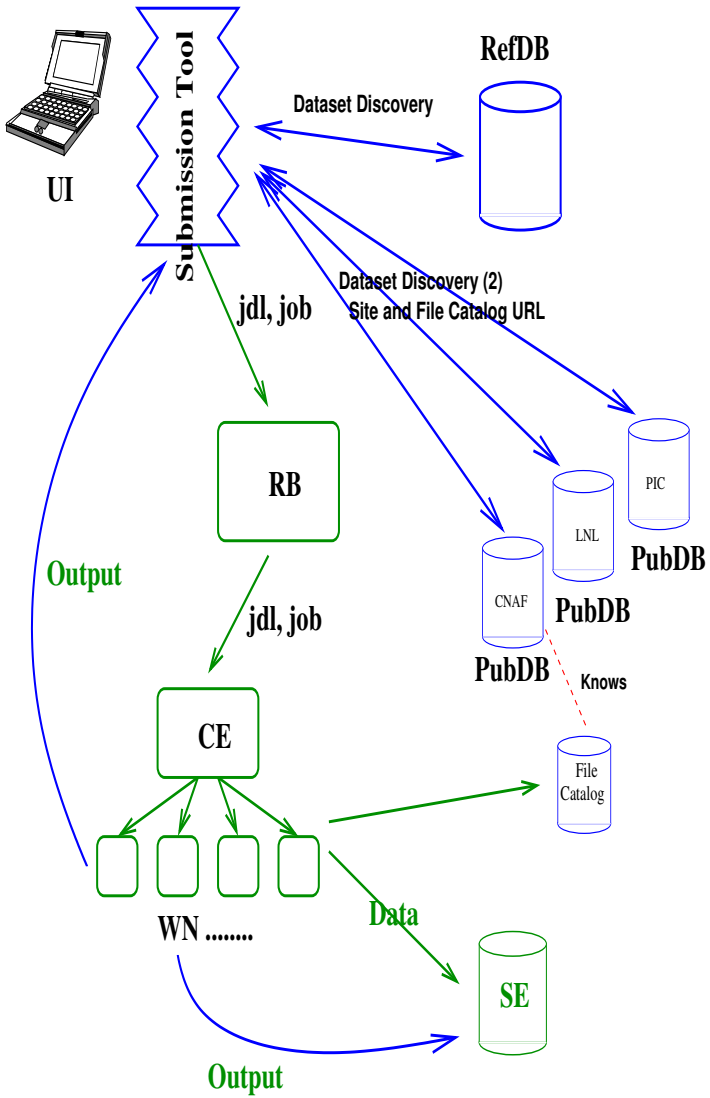
- Early stage of development (version 0_0_11)
- Actively developed to cope with (many) user requirements
- Actively used by many CMS end users $\mathcal{O}(10's)$, with little or no Grid knowledge
- Already several physics presentation based on data accessed via CRAB
- Successfully used to access from any UI data at Tiers-1 (and some T2)
  - CNAF (Italy)
  - PIC (Spain)
  - CERN
  - FNAL (US)
  - FZK (Germany)
  - IN2P3 (France)
  - RAL (UK): still working
  - Tiers-2: Legnaro, Bari, Perugia (Italy)
- Estimated grand total $\mathcal{O}(10^7)$ events

Workflow



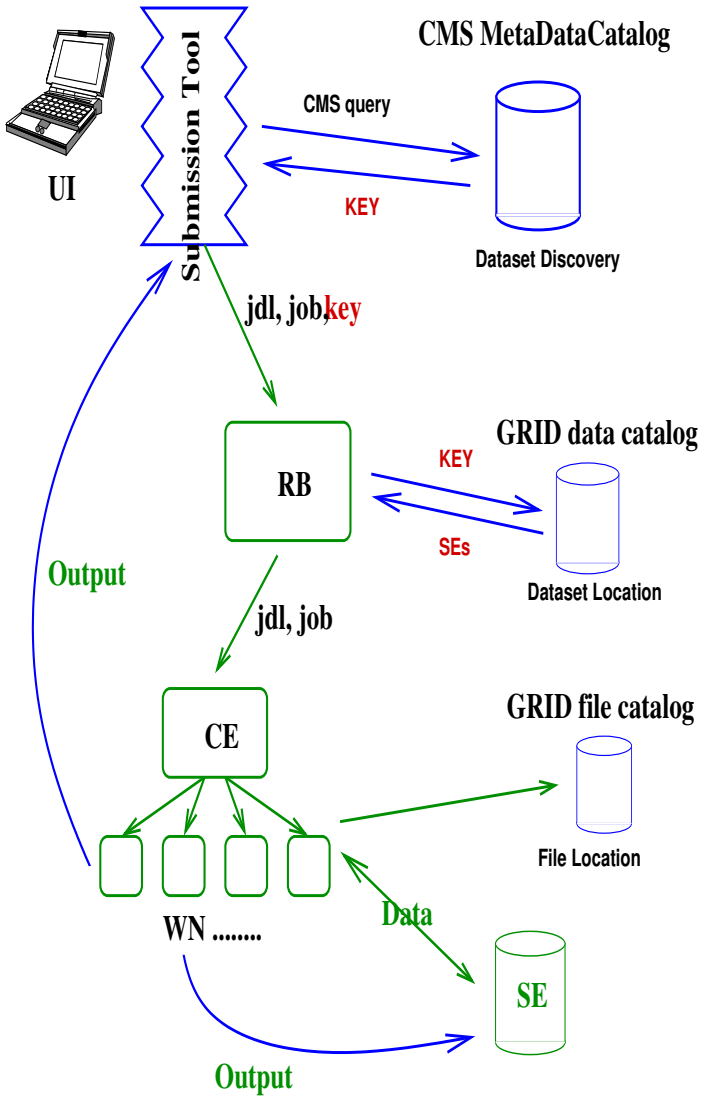
- User develops code on local UI
- Use CRAB for Grid submission
  - Input is Data to be accessed, code
  - Job preparation (private code, splitting, submission, . . .)
  - Create wrapper job to be submitted to Grid
- RB (or tool) uses Data Location Service to find good Site
- Job arrives to Working Node and runs against local Data using a local FileCatalog
- Output is retrieved or stored on Storage Element

Present



- Dataset Discovery: RefDB (CERN) and PubDB (one per site)
- RefDB knows which PubDBs publishing data
- Each PubDBs publish site (CE)
- Local PubDBs knows about Dataset details (# events, ...) and URL of local FileCatalog(s)
- Submission tool query RefDB & eligible PubDBs
- find Dataset location (CE) and tell the RB (as requirement)
- RB ship job to CE
- From WN, LocalFileCatalog (xml or mysql) knows file location (used by COBRA)

Data Location and Access: a proposal



- Dataset Bookkeeping Service (CMS)
  - higher level, interface to physicist
  - provide query mechanism
  - output is a “key(s)”, uniquely associated with *Data chunk(s)*
  - Data Chunk is an unbreakable unit (Atom). the granularity is defined by CMS (today is Dataset . . .)
- Grid Data Location Catalog
  - Given key identifying DataChunk $\Rightarrow$ list of SE(s) $\Rightarrow$ RB get CE(s)
  - Use only *abstract Data*, not files!
- Grid local file catalog
  - Available at local sites
  - Files location for CMS framework and DataManagement system

Outline

CMS SW deployment.

- User job runs against pre-installed CMS software
- CMS system to deploy release sw, rpm based, not root privileges needed
- Run installation with *ad-hoc* job run by cms SoftwareManager Grid user
- Has special privileges to write in shared area on CE
- Many problems with CE specific configuration (mainly afs)
- What of site (eg CERN) has already all sw installed? Need to made ad-hoc solution
- No automatic mechanism to deploy releases as soon as they are available
- SW tag (VO_CMS_ORCA_X_Y_Z) published by CE
- Match done by RB based on user requirement: if failed job *Aborted* but submission fine: need to check job status until *Scheduled*

Job clustering.

- Typical User job is splitted into several *subjobs* each accessing a fraction of total input data
- Subjobs are identical but for few bits
- Same Input Sandbox, same requirements, etc. . .
- Eventual common pre-job:
  - Stage-in (pinning) of input data from MSS
  - User sw compilation and linking
- Need job cluster (or bulk) seen as a single entity
- Allow bulk operations (submission, query, status, cancel, . . .)
- Also possible to get access to single sub jobs
- SubJob number available at WN level, used by job wrapper
- Several splitting logic possible
  - first iteration done at UI level
  - then at RB level, using Grid data location

Data Discovery and Location.

- Today completely done by CMS specific tools and services
- Data discovery is (and will remain) CMS specific
- Data location is not
- CMS choice is to avoid file-based data discovery
- User (and user application) does not access single files, but *data chunks*
- User does not need to know which are the files she will access from WN
- Need to know about files only at WN level, not before!
- CMS want to decouple data discovery from File access
- Data catalog *is not* file catalog.

Access to Local Data.

- Local Pool Catalog
  - Created by Production tools or by PhedEx
  - Supported xml (one per dataset/data tier), mysql
  - Maintenance by local site administrator
- Access to local files
  - From WN access to local data guaranteed by local site admin
  - Any Pool/Root compliant protocol is fine
  - Actual protocol defined inside local Pool Catalog
  - Used Posix (NFS), RFIO, dCache
  - Some problem with RFIO authentication: grid *user* and *group* copied into disk server `/etc/passwd`
  - Copy to WN working area of needed files not an option!
- Both should be provided by grid

Input sandboxes.

- Today sent via input sandbox:
  - Configuration files,
  - Job ancillary files,
  - User libraries and executable
- Size limit on InputSandBox $\mathcal{O}(10)$ *MB*
- Use SE for big input stuff: many problems.
  - Which SE?
  - Close to UI (not necessarily defined)
  - Close to CE, not known in advance
  - Must be sure to avoid name clashing (using what user want not some relic from past jobs)
  - Must cleanup everything at the end: when? data lifetime?
  - Should foresee a experiment specific service?

Output produced.

- User wants output on her computer or on a storage accessible from her computer (via posix or any usable protocol, eg RFIO)
- In general not interesting to have output on Grid
- Different for “production” use cases
- If output via output sandbox: user must ask when *Done*
- Query L&B every *x* seconds until job is *Done* scalability??
- Can user be notified when job is finished?
- If storage has the proper server installed (*e.g.* `gsiftp`) possible to just copy the output when done.
- What about ACL? Output written according proxy certificate ACL, which are different from storage ones
- `cms002` need to write on `/castor/cern.ch/user/s/slacapra/...`

User Support.

- Critical issue!
- Analysis is performed by generic users, with little or no specific knowledge about grid
- User does not want to became an expert also on Grid: there are already so many things she need to know to do analysis, too much!
- CMS specific support
- Grid support: how?
- GGUS, EIS, ROC ... which is the correct entry point?
- Should we have a CMS–oriented Grid support? (yes)
- Support on sites: Grid or(/and) CMS specific?

Outline

Summary

- CMS first working prototype for Distributed User Analysis is available and used by real users
- Proposal for catalog/data discovery/data location presented
- Pragmatic approach: many lesson learned, lot of feedback provided to Grid team