Workload Management

Stefano Lacaprara

Department of Physics
INFN and University of Padova

CMS Physics Week, FNAL, 12/16 April 2005

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 sw 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 to grid services for CMS user
- User is supposed to be able to develop and run her analysis code interactively

Directive given 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, total number of events, . . .)
- Name of Executable
- Configuration `.orcarc` cards: the one she uses locally!

CRAB CMS Remote Analysis Builder.

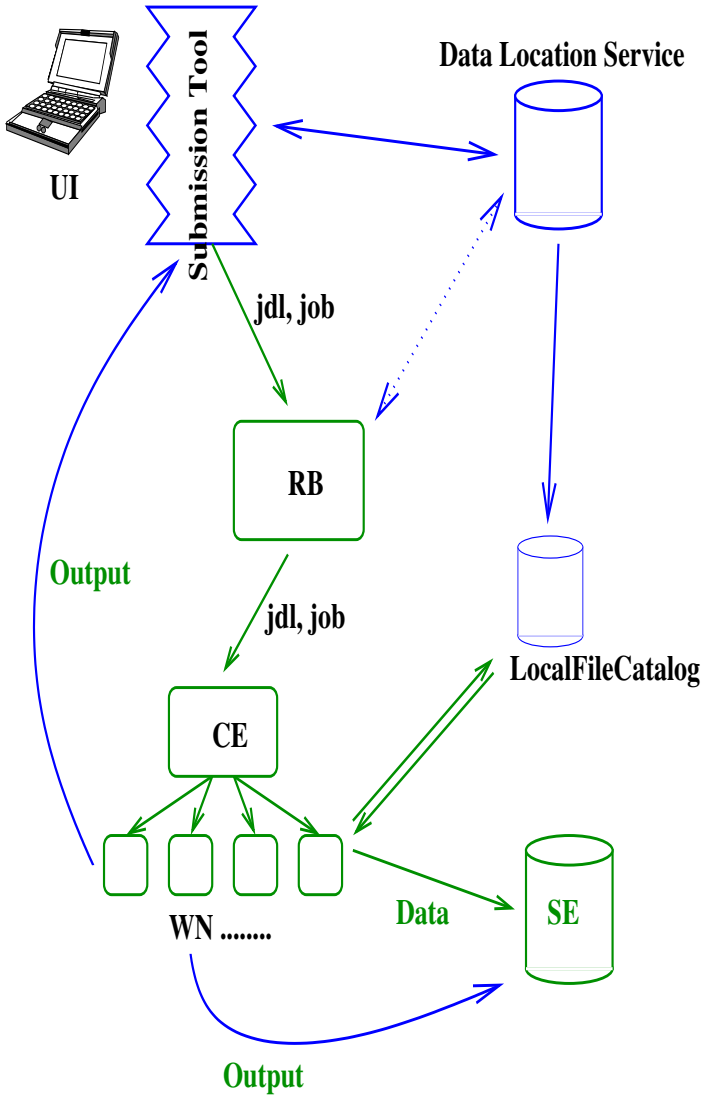
CRAB functionalities

- 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.

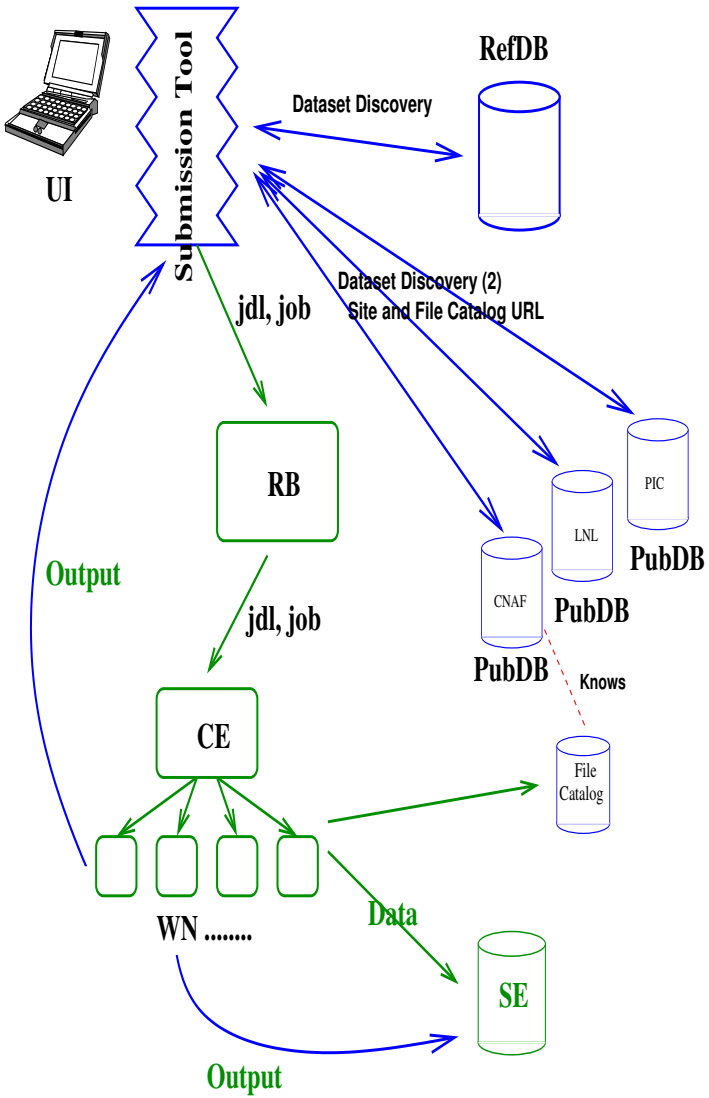
- Early stage of development (version 0_1_1)
- Actively developed to cope with (many) user requirements
- Actively used by many CMS end users $\mathcal{O}(10's)$, with little or no Grid knowledge
- Already several physics presentation based on data accessed via CRAB
- Successfully used to access from any UI data at Tiers-1 (and some T2)
  - FNAL (US) (yesterday $\mathcal{O}(200)$ CRAB jobs running!)
  - CNAF (Italy)
  - PIC (Spain)
  - CERN
  - FZK (Germany)
  - IN2P3 (France)
  - RAL (UK): still working
  - Tiers-2: Legnaro, Bari, Perugia (Italy)
- Estimated grand total $\mathcal{O}(10^7)$ events

Workflow



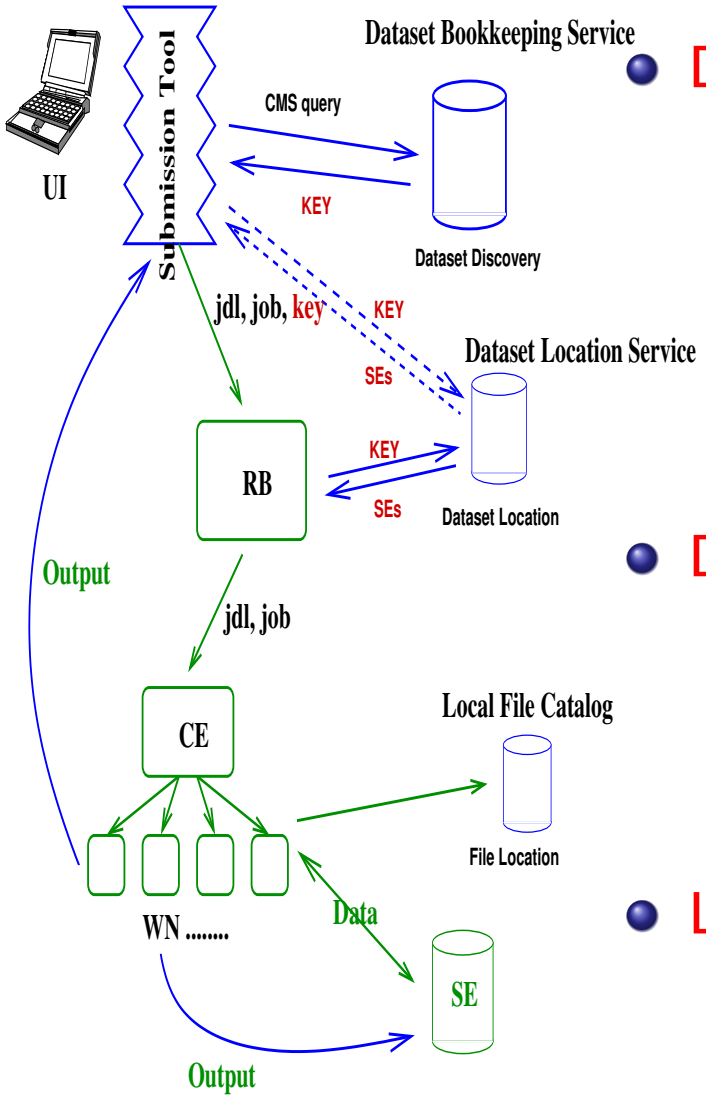
- User develops code on local UI
- Use CRAB for Grid submission
  - Input is Data to be accessed, code
  - Job preparation (private code, splitting, submission, . . .)
  - Create wrapper job to be submitted to Grid
- RB (or tool) uses Data Location Service to find good Site
- Job arrives to Working Node 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 (CEs) and tell the RB (as requirement)
- RB ship job to one of the possible CEs
- From WN, LocalFileCatalog (xml or mysql) knows file location (used by COBRA)

Data Location and Access: future



- Dataset Bookkeeping Service (CMS)
  - higher level, interface to physicist
  - provide query mechanism
  - output is set of *Data chunk(s)*
  - Data Chunk is an unbreakable unit (Atom). granularity defined by DM–WM (today is Dataset . . .)
- Data Location Service
  - Given key identifying DataChunk $\Rightarrow$ list of SE(s) $\Rightarrow$ RB get CE(s)
  - Can be done at UI or RB level
  - Use only *abstract Data*, not files!
- Local File Catalog
  - Available at local sites
  - GUID to PFN mapping

Outline

CMS SW deployment.

- User job runs against pre-installed CMS software
- Lot of childhood problems here.
- Installation tool (`xcmsi`) available and working;
- Installation done via *ad-hoc* job run by cms SoftwareManager Grid privileged user;
- Meeting last week with Operation (Nick) and xcmsi team to drive future
- Setup an automatic mechanism to deploy releases as soon as they are available;

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. . .
- Need job cluster (or bulk) seen as a single entity
- Allow for bulk operations (submission, query, status, cancel, . . .)
- Also possible to get access to single sub jobs
- Splitting can be done at UI or (possibly) at RB level, using Data Location and resource matching

Data Discovery and Location.

- Current implementation based on RefDB and PubDB will be re–factorized into DBS, DLS;
- Data discovery is (and will remain) CMS specific (DBS);
- Data location is not: DLS can be a common tool for LCG;
- 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;

Dataset access monitoring

- Analyze data access pattern.
- Which data have been accessed by users?
- Which datasets need to be replicated?
- How efficiently are we accessing remote resources?
- Action: not easy today.
- Need “central” monitoring, not trivial to setup.
- Will investigate LCG Logging and Bookkeeping service and Monalisa
- Could spoils CMS specific site access problems (eg problems with incomplete catalogs, etc...) or problem with specific dataset

Job Monitoring.

- Active discussion with BOSS team to useful integration with CRAB;
- Agreement with BOSS 4 architecture;
- Will be fully used by CRAB;
  - Job tracking;
  - Batch scheduler (local or grid) interface;
  - Job monitoring;
  - Application real time monitoring (*good if you can have it, life goes on if you can't* policy);
  - Job logging and bookkeeping;

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 storage has the proper server installed (*e.g.* `gsiftp`) possible to just copy the output when done.
- What about publishing user output so that other people (*e.g.* same analysis group) could use it?
- What about *promoting* private output (*e.g.* re-reconstruction) to “official”: provenance, etc . . . ?
- New DM–WM architecture should allow both.

User Support.

- Critical and time consuming issue: means CRAB actively used!
- 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!
  - Pure Grid problems;
  - CRAB support;
  - Data access support: problems with catalogs, missing files, problems with MSS, etc. . .
  - ORCA problems. . .
- First level *Triage*: inter–operation with Grid/sites support

Open Science Grid integration

- Very useful discussion begun with Frank Wuerthwein and his group
- General idea:
  - Provide just one interface (CRAB) to final user;
  - *back–end* for LCG and OSG could (will) be different due to differences of the two Grid architectures;
  - Many common tool and service will be shared: notably the Data Management infrastructure (DBS, DLS, etc . . .)

Summary

- CMS first working prototype for Distributed User Analysis is available and used by real users;
- Many lesson learnt from CRAB usage;
- First lesson: people is using it
- Second lesson: real effort must be put in deployment
  the more problems are not to be addressed by CMS directly, the better