CRAB tutorial

26/06/2009

Prerequisities

- We expect you know:
  - Howto run CMSSW codes locally
  - How to get a user proxy
  - You are registered to the CMS VO
  - You are registered to siteDB

- If at least one of the previous points is not true,please look at the CMS workbook
  - https://twiki.cern.ch/twiki/bin/view/CMS/WorkBook

CMS Computing Model

The CMS offline computing system is arranged in hierarchical Tiers geographically distributed.

Online system

Offline farm

recorded data

Data from DAQ are sent, stored and first step reconstructed at Tier-0, then spread over T1s

Tier 0

CERN Computer center

Tier 1

France Regional Center

Italy Regional Center

Fermilab Regional Center

T1s take care about
- calibration, skimming and reconstruction. They sent data to T2s

Tier 2

Tier2 Center Tier2 Center Tier2 Center . . .

T2s provide power for analysis and simulation

Tier 3

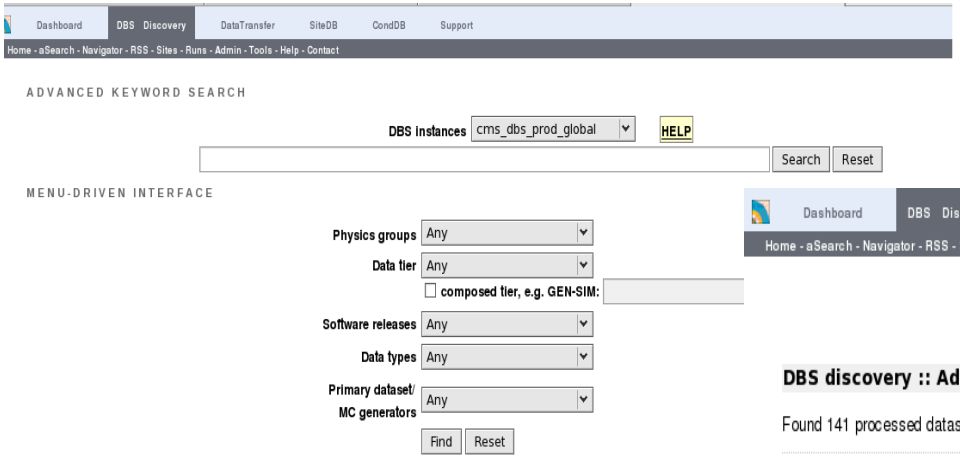
InstituteA InstituteB

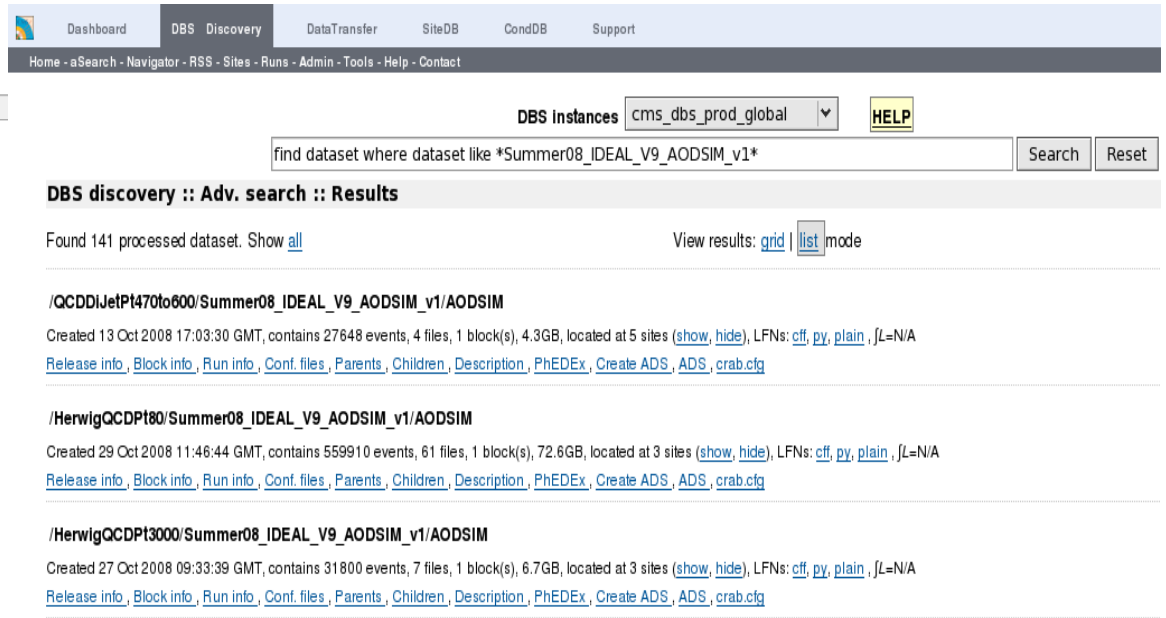
workstation

Remote data accessible via grid

How to discover where data are

- Dataset stored to remote sites are registered in the global DBS https://cmsweb.cern.ch/dbs_discovery/
- DBS query SQL-like





CRAB for newbies

- What is the CMS Remote Analysis Builder?
  - Is a user-friendly command line tool that let you interact with the Grids & the Local Resources (LSF/ CAF)
  - Takes care of hiding you most of the distributed model complexities
  - Lets you run CMSSW over whole datasets already distributed among the sites
  - Automate most of the analysis workflow (status tracking, resubmissions, …)

How CRAB works

- CRAB does not compile on the remote sites
  - What you have locally is what you get worldwide
- CRAB takes care of reproducing your whole working environment remotely
- The only relevant information CRAB needs
  - which version of CMSSW you refer to
  - the ParameterSet
  - the Dataset name you'r going to analyse

What CRAB does

- Create tasks preparing the jobs for the remote environment and split the load according your requests
- Submit the created jobs
- Monitor the progress of the jobs
- Handle user output moving them to your reference SE
- Resubmit failed jobs
- Notify about task progress via e-mail
- Publish the result to DBS

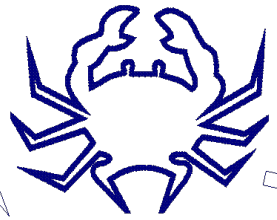
The CRAB flow

At the User Interface the user:

1)Writes local CMSSW analysis code
2)Decides the Dataset Name to analyze
3)Modifies the CRAB configuration with analysis info

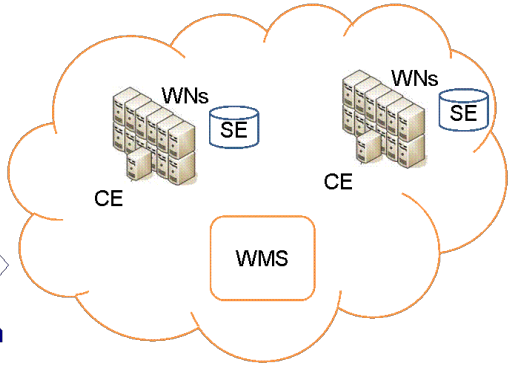
A) User parameter
DatasetName

CRAB installed on the User Interface

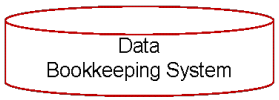


C) Job creation

D) Job submission



Data Discovery system knows:

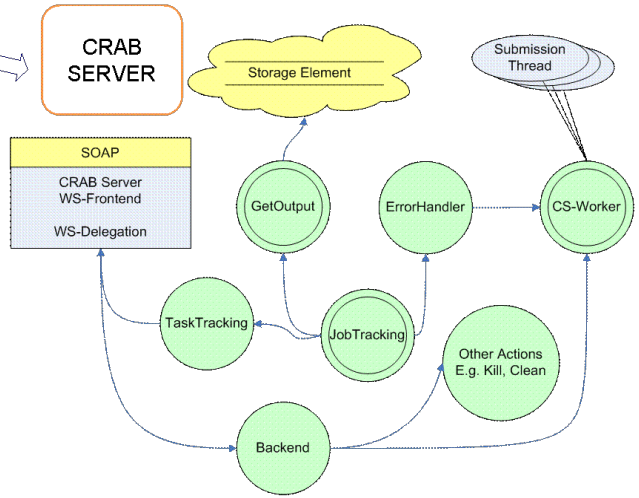


Which kind of data are available
How the dataset is split in event collection
Where event collections of a dataset are stored

Storage Element Name

B) Data Discovery
DatasetName
Storage Element Name

C1. Creates the user code wrapper and packs it to be run on the remote resources.
C2. Splits the analysis payload according to data location
D1. Submits created jobs directly to the Grid or to the CRAB Server. The SE name is used to drive the Workload Management System (WMS) to match remote resources
D2. Manages the produced output. Copies it to a remote SE or retrieves it to the UI
D3. Publish user results to be shared and analyzed further by the collaboration



CRAB functionalities fully automated with the server
•Monitoring of job status
•Kill and resubmission of jobs
•Retrieval and handling of user output
•User e-mail notification about task execution

How you can talk to CRAB

- Every configuration in CRAB is set through the directives reported in the crab.cfg file
  - Organized as key = value pairs
  - Grouped in macro-sections [CRAB], [CMSSW], [USER], …
- A minimal and a full template for crab.cfg are in $CRABPATH/crab.cfg, $CRABPATH/full_crab.cfg
- Inline documentation (crab –h) guides you to set attributes
- **At Purdue users may access CRAB at: /grp/cms/crab/CRAB**
- **source /opt/osg/setup.sh**
- **source /grp/cms/crab/crab.sh**

Some useful commands

- Create the CRAB project (by default crab.cfg)
  - crab –create
- Submit your jobs
  - crab –submit <all | n | rng > [–c <crab_prj>]
- Track the jobs progress
  - crab –status [–c <crab_prj>]
  - alternative use of the CRABSERVER web interface
- When jobs get done, retrieve data:
  - crab –getoutput <all | rng > [-c <crab_prj>]
  - output will store in <crab_prj/res>

Some useful commands

- If you need to kill some job
  - crab –kill <all | n | rng > [–c <crab_prj>]
- Get post-mortem infos (useful for abort jobs)
  - crab –postMortem <all | rng > [-c <crab_prj>]
- Resubmit
  - crab –resubmit <all | rng > [-c <crab_prj>]
- Publish your results, if you need to share them
  - crab –publish [-c <crab_prj>]
- Clean the obsolete CRAB project
  - crab –clean [-c <crab_prj>]

Copy of outputs to a SE

- Necessary step when big output are produced
- CASTOR isn't the best place where stored data. You should use T2 or T3 SEs.
- Storage elements are known as “CMS official SE” registered in the siteDB, and “not official”
  - Crab configuration is different for these cases.
    - “official”: it's necessary to know the “alias name” of storage as registered in the siteDB. CRAB discovers the correct path where to copy your data
    - “not official”: it's necessary to know the storage name and the path

Copy example

```
[USER]
return_data=0
copy_data=1
storage_element=T2_US_Purdue
user_remote_dir=neumeist
```

Using the SE "alias name" as registered in siteDB

```
[USER]
return_data=0
copy_data=1
storage_element=srm-dcache.rcac.purdue.edu
storage_path=/srm/managerv2?SFN=/pnfs/rcac.purdue.edu/data/store/user/neumeist/
user_remote_dir=neumeist
```

Using the SE real name

in the crab wrapper script these configurations are translated as:

```
SE=srm-dcache.rcac.purdue.edu
SE_PATH=/srm/managerv2?SFN=/store/user/neumeist/abc
```

Publication

- Essential step to re-run with Grid and CRAB
- Before to publish your data
  - You must know the local DBS instance where to publish
  - You must know a Tier2 StorageElement where to store data
  - For “not official SE” you have the responsability to know which directory allows data publication
- Moreover
  - crab.cfg must contain the publication directives BEFORE creation The .root must be an EDM file
- https://twiki.cern.ch/twiki/bin/view/CMS/SWGuideCrabForPublication

Publication example

datasetpath=/TauolaTTbar/Summer08_IDEAL_V9_AODSIM_v1/AODSIM

```
[USER]
publish_data=1
storage_element=T2_US_Purdue
publish_data_name=pubb_data_name
```

Using the SE "alias name" as registered in siteDB

```
[USER]
publish_data=1
storage_element=t2-srm-02.lnl.infn.it
storage_path=/srm/managerv2?SFN=/pnfs/lnl.infn.it/data/cms/
user_remote_dir=/store/user/
publish_data_name=pubb_data_name
```

Using the SE real name

in the crab wrapper script these configurations are translated as:

```
SE=t2-srm-02.lnl.infn.it
SE_PATH=/srm/managerv2?SFN=/pnfs/lnl.infn.it/data/cms/store/user/fanzago/TauolaTTbar/ \
        pubb_data_name/${PSETHASH}/
LFNBaseName=/store/user/fanzago/TauolaTTbar/pubb_data_name/${PSETHASH}/
```

Monitoring

- Use:

  http://dashboard.cern.ch/cms

CRAB documentation

- The inline manual
  - crab –h for crab configuration syntax
- HowTo and FAQ Twiki pages
  - https://twiki.cern.ch/twiki/bin/view/CMS/SWGuideCrabHowTo
  - https://twiki.cern.ch/twiki/bin/view/CMS/SWGuideCrabFaq
- The CRAB twiki page
  - https://twiki.cern.ch/twiki/bin/view/CMS/SWGuideCrab
- Get support from hn-cms-crabFeedback mailing-list (please add your stderr-stdout-log and crab.cfg)
- **Purdue specific information:**

  http://www.physics.purdue.edu/Tier2/content/view/58/87/

Copy to CASTOR user area

```
[USER]
  copy_data = 1
  storage_element=srm-cms.cern.ch
  storage_path=/srm/managerv2?SFN=/castor/cern.ch
  user_remote_dir=/user///whatever
```

```
user_remote_dir=/user/f/fanzago/test/fede
```