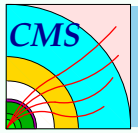

CCS Data and Workload Management

CERN, Monday 20 Sept 2004

Workload Management draft mandate and workplan

Stefano Lacaprara

Stefano.Lacaprara@pd.infn.it

INFN and Padova University

Outline

- Mandate and goals of WM,
- Draft workplan,
- Deliverables,
  - Data publication,
  - Access to resources,
  - Software deployed,
  - Job preparation,
  - Job splitting,
  - Monitoring and bookkeeping,
  - Output retrieval and publication,
  - Documentation and training,
- Status and Future,
- Lucas's task decomposition,

Goals and Mandate

- Coordinate and promote work to allow end user access to data
- Strongly end-user focused: locate data, prepare and run jobs at regional center

The goal of the WM is to allow physicists to perform analysis on a distributed way, that is accessing data wherever it is, as soon as it is produced, using efficiently all the resources available on all Tier-n, and all in a user transparent way.

- The tools and middleware developed by LCG and related projects (such as Grid3/EGEE) will play a major role. The purpose of the cross project is to develop tools, information sources, catalogs etc to integrate CMS framework with Grid.
- For an effective use of Grid resources, some CMS specific should be used/integrate in the grid workload management service. The identification and development of these components is responsibility of CMS WM.

Foreword on workplan

Proposed workplan strongly *pragmatic*

- Start with simple use case, try to address concretely it
- Learn from implementation of simple use case, and from user feedback in order to address *NLO* use cases
- Do not concentrate **now** no complex analysis scenarios (such as interactive, data movement *on-demand*, etc.) Next iteration.
- Analysis model depend strongly on data management strategies (and vice-versa)
- Initial phase: assume **quasi static**, **dataset based** data distribution
  - Datasets (as a whole) are available in one (or more Tier-n), fully validated and *finished*
  - Dataset movement is strongly coordinated and done off-line wrt analysis
  - Datasets come with local, fully specified Pool file catalog

Workplan

Actions need to start a user analysis:

- Produce and publish data
  - Prod responsibility
  - Publish data: put the information about the produced data where a physicist/tool can found it.
- Guarantee access to CPU's close to the data
- Install proper analysis software on local resources
  - publish the information that the sw/version is actually installed
- Create a job(s) that can be submitted to remote resources
  - Including job splitting
  - User provided code (to be compiled locally) or user library
  - Smart resource broking in order to find the best site(s) where to run
- Monitor the status of the jobs
  - Resubmission on failure
  - Allow debug of user code, data access,...
- retrieving of the job output
  - publication of results (such as DB, trees, etc...) for group wide usage

Deliverables

- Dataset catalog: Define information strategies in order to allow users/tools to know which datasets are available, where they are and how to access them (in collaboration with DM and Prod)
- Software: Distribution of software and coherent publication of installation info
- Job preparation and submission tool: User friendly tool able to deal with job preparation, job splitting, job submission and output retrieval
  - Including use of private user code
  - Also (but not only) GUI and/or Web front end
- Monitoring and bookkeeping: effective, light and user friendly: the user is non–expert!
- Private data: Allow for user-data publication for group-wide usage
  - Places where user data can be put and retrieved by him or group (SE, afs, web, ...)
  - Way to notify users that *private* data is available, how to get to it, what contains and how it was produced

Data publication

- Agreement on publication schema with Prod
- PubDB: Dataset catalog focused on end user analysis
- Linked with RefDB (Production), where information about dataset definition, status, etc are stored
- Decomposition of publication info
  - Dataset catalogs
    - Information to be used by the RB to find where to submit the job
    - what to publish? Site (CERN, PIC, CNAF, ...)
    - Or SE, so that RB can find the closes CE
    - This CMS component should interact directly with RB, LCG component (work going on)
    - Should solve issues about file vs dataset catalog
    - Do we need/want a LCG generic dataset catalog?
  - local POOL file catalogs
    - to be used by job once landed on CE
    - infos are CE specific
    - all the info needed by COBRA to actually access data
    - to be used in user `.orcarc`

PubDB: issues

- local POOL file catalogs now in xml
- Foresee also MySQL/ORACLE catalogs: how to access them locally?
- Today xml explicitly depends from the site: how to made them more “abstract”?
- Integration with DM
  - knowledge of file location is inside local POOL catalogs
  - What if DM moves files? How to keep local catalogs uptodate?
- Where to locate PubDB? Centrally, replicated, distributed?
- PubDB allows great flexibility, need to understand use cases
- Two kinds of information: do we need two different DBs

  oh, my, not an **other** DB! What about a DB non prolification treaty?
  - Local PubDB on each site publishing data
  - Information about dataset location (for RB) replicated on central (CMS wide, PRS wide, T1 wide, ...) PubDB, to be used by RB
- Responsibility: who?
  - Development: user is WM, should also take responsibility? Expertise mostly on Prod.
  - Maintenance: if PubDB in each Tn, need to identify resources on Tn to install, update and maintain it. Need Tn commitment!

Access to resources

Guarantee access to CPU's close to the data

- Using LCG tools (in near(?) future gLite): each Tn should provide some resources (mostly CPU) installed with LCG middleware.
  - Which LCG version?
  - Integration with VOMS is highly desirable, in order to allow US user to use resources.
  - Crucial: for farm installation, management and upgrade, human resources are needed on every Tn. These human resources should be identified, acknowledged and their task to be defined: whether "just" infrastructure maintenance or also higher level task, such as those related to production and/or data management.
- Foresee jobs policy and priority
  - Within LCG, possible CE level (via local batch scheduler configuration)
  - We do need also a CMS/PRS-wide policy? How to do it? Requirement for EGEE.

Software deployment

Install proper analysis software on local CPU

- Analyst will use CMS official packages (libraries) plus private ones.
- Need to distribute CMS sw coherently in all sites
- Not only binaries but also source, allowing compilation/linking
- Software environment must be identical to the one found on interactive resources (eg `lxplus`): `scram` based installation
- CMS environment ($PATH, utilities, etc...) must be automatically available on remote site. Not the case today, user must source configuration file: requirement for EGEE
- Installation policy: when and where to install new releases? Everywhere, on-demand, selected sites? Removal?

Software deployment (II)

- RPM based distribution already available, widely used for off-site installation
- RPMs installation on LCG CE via Grid done
- Publish (via CE) sw version installed
  - Info published is not fully user oriented CMS_95_3 and not ORCA_8_4_0
- RPM preparation is not yet optimized, leading to the need of lot of space for every sw version, including tons of unnecessary things
- Using two different methods for analysis and production (rpm vs DAR)
  - There are reasons for this, but integration is highly desirable
  - More on Nick talk tomorrow
- Again, integration with EGEE: we have a system working, don’t want other to re–invent the wheel...

Job preparation

Create a job(s) that can be submitted to remote resources

- Key tool, to be used by end-user
- In principle, the only tool (s)he need to learn about
- Should be as simple as possible to use
- Should address as much use cases as possible Cannot guarantee for all use cases: very experienced user (hacker-like) will anyhow use private tools...
- Get user input
  - Data: just Dataset/Collection/Owner. All technicalities must be hidden
  - Software: define versions, etc...
  - Private code
  - Other input: configuration cards, etc...
  - Output: to be shipped back (or saved on SE, see after)
- Handle private code
- Job preparation, wrapper, etc..
- Deal with job splitting, jobs cluster (see after)

Job preparation (II)

An user view:

- User need to know how to run locally ORCA
- Remote (non-CERN) user should be allowed to develop his/her code on local resources
- Issues:
  - local sw installation: use Grid resources (so sw is already installed) or other (need to install, export)
  - to run ORCA, user need some data to test code, histogramming, etc.
  - Use locally available data (on near by SE, if any) or copy locally fraction of dataset from not-so-near SE or need some test samples (as the one available at CERN)
  - Distribute it? Who? Or produce locally? Again who?
- Once happy with local running, want to process full dataset(s)

Job preparation (III)

- Use standard tool (WM will provide)
- `scram` like interface would be nice ((s)he's already familiar with it)
- Private code preparation:
  - pack all libraries/executable in private working area: ship to CE
  - pack code and compile locally
  - `wget` (or alike) code/libs from CE
  - Put code/libs in SE and get from there
- Implication for job cluster
- Job splitting
- Jobs submission

Job preparation (IV)

An example for user interface

`.wmtoolrc` (tool configuration)

```
InputCollection=/System/Owner/Dataset
TotalEvents=-1
EventPerJob=1000
Executable=MyH2WWAnalysis
Output=H2WW.root
Orcarc=.orcarc
...
```

prepare wrapper, pack code, job splitting, etc...

```
~>wmtool prepare
```

submit to grid resources

```
~>wmtool submit
```

monitor status

```
~>wmtool status
```

get output

```
~>wmtool getoutput
```

Job splitting

- Crucial item for effective resource usage
- Most(all?) analysis run on many events the same code
- Parallel jobs are the same but for few bit (event range)
- Many issues
  - If splitting done too early, RB see individuals jobs, not a job cluster
  - Effective splitting should know which resources are available (eg number of available nodes, speed, bandwidth, etc...)
  - Must know also where data are: use case of dataset available in many sites or splitted among different Tn...
  - Private code: Need to ship/compile many time exactly the same stuff
  - ...
- Job splitting is CMS specific (only CMS know how to split a job)
- Need a high level Resource Broker with CMS component/plugin inside

Job splitting (II)

- Probably most important deliverable of WM for mid/long term
- Need workplan on this crucial issue, in collaboration with LCG/EGEE
- Possible solutions (naïve ideas):
  - Send to RB a single job which actually does the splitting (using also information available from RB), creating n sub-jobs
  - Multi level splitting: first at RB, then at CE
  - First job arriving on CE open a “tunnel” for authentication, cluster sandbox (private code), etc...
  - Pilot job (“a’la DIRAC”)
  - DAG like job: first one get/compile code, the others use it
  - ...
- In any case, collaboration with LCG/EGEE WMS group
- Try to organize meeting with some preliminary ideas to be proposed/discussed

Monitoring and bookkeeping

Monitor the status of the jobs

- Resubmission on failure
- Allow debug of user code, problems in data access,...
- Resource monitoring: LCG. May suggest metrics: can/should be used for effective RB
- Application monitoring is application specific.
- Handling of information flow, bookkeeping can reuse many EGEE components
- Monalisa
- BOSS
- ...

Job output status

- ORCA knows best is something went wrong, and what (in principle)
- Not very smart develop log parser to reverse–engineering problems

Number of events processed, software version user for analysis, where the output is stored, ... (see after)

Output retrieval and publication

- Soon to become a popular use case
- Group, PRS trees
- Simple use case: ntuple/tree
  - Just send it back to user
  - Merging in case of job splitting
- Or store in SE: big output or group wide usage
  - Publication of stored output: how?
  - Tony's simple and nice idea
    - Why not on PubDB?
    - Can be used also for bookkeeping (events in a set of trees, sw used to produce them, etc...)
    - Should investigate further, but promising
  - Implication with DM, in case output is to be moved around, etc...

Documentation and training

- Very important: the clients are physicists!
- Don’t need (nor want) to be GRID expert
- Surely don’t to learn yet an other complex tool: (s)he already had to learn too much things (C++, scram, Pool, ...)!
- Simple documentation and tools easy to learn and use
- Interface, user command similar to familiar tools (eg scram)
- Tutorial for user training

Status and Future

- Many tools developed in past few mounts
- Data accessed successfully via Grid at PIC, FZK, LNL, Bari, etc...
- Data publication was still very rough (hand made web pages)
- Useful as proof of concept to understand concrete problems and possible solutions
- Most of functionalities already present
- Need to develop/coordinate for a production quality tool
- User feedback is crucial! Select limited set of average–user to test the tools in the early phase

- Actual workplan focused on batch analysis
- Follow strictly (within APROM) other analysis scenarios, such as interactive, etc...
- New use cases can arise: understand how to develop strategies to match them, have analysis framework flexible enough
- Non event data (calibration, etc...): how to access it?

Lucas task decomposition

- Architecture and basic services
  - Interaction with application framework
  - Security infrastructure
  - Farm installation and management
  - Monitoring tools
- Batch job management tools
  - CMS jobs for LCG
  - MOP
  - Using BOSS for LCG
- User tools
  - RefDB extension and interfaces for Users
  - Tool for job creation for users
  - GUI
  - Interactive analysis
- Validation of Grid for User
  - CMS/LCG testbed
  - Validation of non–LCG project releases
  - Validation of LCG releases

Backup

LCG-2 Components

Use as much as possible LCG-2 components to fulfill task

- User Interface: access to the Grid for end-user, authentication, login, etc...
- Wish: light UI, easy to install on (every) node used by user (desktop, laptop...)
- Resource Broker: decide where to send the job
- matchmaking based on CE information (such as OS, VO, sw installed, etc...), and Data location
- Data location file-based not satisfactory. User want to access Dataset (collection), not single files. User don't know which individuals files are needed (and don't want to!)
- Data location based on logical Dataset, implemented by CMS specific Dataset Catalog (PubDB, see later)
- RLS: not used directly for analysis. Need to understand link between File catalog and Dataset catalog, with Data Management project
- Computing Element: where the jobs actually run, and where the CMS software is installed
- Storage Element: where the data is located

Integration with EGEE

Recent architecture document

- Follow strictly the EGEE development
- Most of non CMS specific tools, infrastructure, etc will be deployed by EGEE
- Need to integrate them with CMS specific tools (such as job preparation, Dataset catalog, etc...)
- Provide use case for CMS in this early phase, to be tested against architecture proposed
- Integration with US grid (NorduGRID)
- Native integration between the two (three) grids or ...
  - Common part, CMS specific
  - part LCG/EGEE dependent
  - part US (OSG/Grid3) dependent
  - ...