CRAB Status Report and Plans

and more: CRAB–CROSS integration plan

Stefano Lacaprara

Department of Physics
INFN and University of Padova

APROM meeting, 14 january 2005

Outline

CRAB Status

- CMS Remote Analysis Builder
- version 0_0_3 released before Xmas
- Functionalities:
  - Data discovery using RefDB/PubDB (V2.8)
  - Automatic user code packing and shipping
  - Job splitting according to user directive
  - Simple monitoring
  - Automatic output retrieval
- given to a set $\mathcal{O}(10)$ beta–tester
- some feedback (few) received

CRAB Short Term Plan

- CVS repository available: need to organize packages and commit
- Adapt and test to PubDB V3.0(?) (improved `FileType`) see after
- Tutorial scheduled for February
- Improve user interface (increase user friendliness)
- Documentation

PubDB and CRAB

- Start using CRAB with newly produced DST
- Need to publish DST Dataset/Owner in PubDB
  - **PubDB V2.4** Does NOT contains enough information to allow remote submission
  - **PubDB V2.8** (enough for CRAB submission) already available (and tested) at: CNAF FZK PIC LNL Bari
  - **New PubDB Vx.x** (?) extend FileType info
    - FileType = { AttacchedMeta, VirginMeta, MCInfo, Hit, Digi, DST, AOD, ... }
    - Needed to understand if a set of catalogs can satisfy user request (eg. user want to acces DST and MCInfo)
    - Each catalog declare what kind of Data it publishes

PubDB and CRAB

- Create complete set of catalogs is the most tricky part today
- Meta
  - in a separate xml catalog
  - within EVD catalog
  - Catalog (web) + COBRA variables (CERN)
- EVD
  - can be many catalogs (DST, Digis, . . .)
  - complex to understand complete set
  - Situation much simpler if just one catalog for all EVD for SITE
  - Agreement with PhedEx team:
  - Use the same catalog used for Data Transfer
  - Dataset trasferred with PhedEx (V2.1) automatically provide complete mysql catalog for all EVD files
  - What for Dataset produced *in site*? Provide xml catalog, should be used to fill mysql catalog.

MetaData access

- FullyAttacched MetaData created by Production
- Available via HTTP from CERN (cmsdoc.cern.ch/production/...)
- together with COBRA Variables to be added in user `.orcarc`
- External Site:
  - Use MetaData by Production
  - Create local cache for scalability
    - HTTP Proxy
    - *hand–made* local copy
  - In both case need to define URL in local PubDB
  - COBRA variable MUST be written in PubDB. Already foreseen but not done

Catalogs set creation

- Already available tool (NoNewNamePlease!) able to create, using PubDB info (for a given Dataset/Owner):
  - `.orcarc` fragment with FileCatalogURL with all needed catalogs
  - Easy to add also COBRA variables
  - Eventual *stagein* command if needed
  - Already used in CRAB
  - Can be VERY useful also for local User (CERN)

CRAB–GROSS Integration Short Term

- a CRAB–GROSS meeting held in Bologna before Xmas
- Cross analyzed pros and cons of both approach to get the best out of the two
- Try to define a road–map for integration
- Short term
  - Parallel development (in a co–operative way)
  - Doubles chances to have something which works reliably
- Status (by Stuart W.)
  - implemented automatic output retrieval
  - one step preparation, submission and output retrieval
  - `scram` integration (find exe/libs)
  - finalizing PubDB interface
  - next step separate GROSS–BOSS installation

CRAB–GROSS Integration Long Term

- Extend BOSS functionality with many GROSS features for job submission/management introducing task (multiple jobs)
- Separate monitoring and logging
  - *Monitoring:* "nice if you can get it but life goes on if you can't" . Temporary information
  - *Logging:* based on reliable retrieval mechanism
- C++ and Python API provided
- Python tool CRAB will deal with
  - Contact Data/MetaData system for data discovery
  - Job preparation and splitting
  - etc ...
- will create **tasks** passed to B(GR)OSS by python API
- submission/monitoring/logging/tracking handled by B(GR)OSS
- important *by product*: job submission via BOSS, thus possible on US grid!

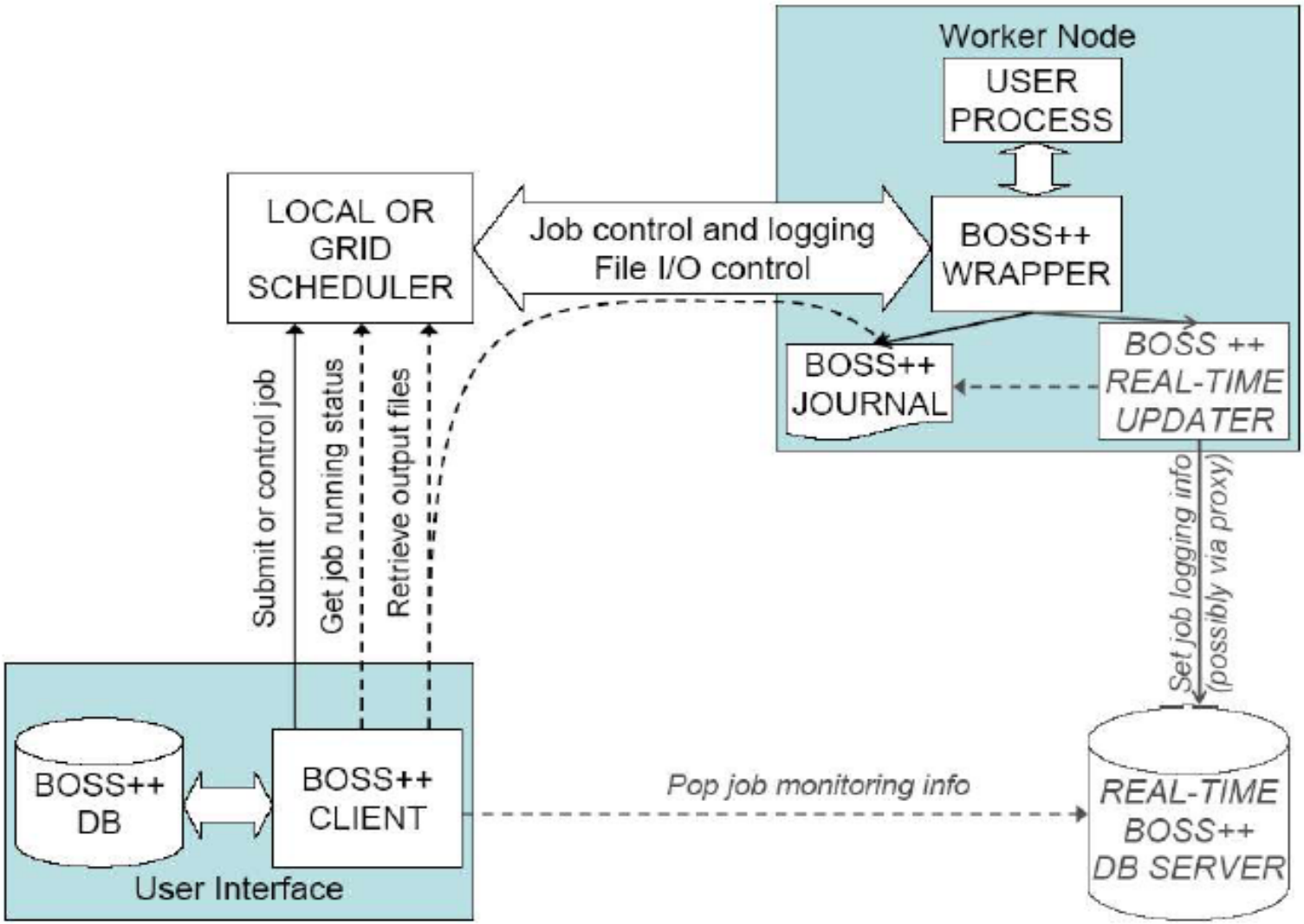
B(GR)OSS++ architecture

- Information in the DB indexed by computational task and not by job
- Multiple job per task (resubmission), multiple task per job (chained task). Task and jobs grouping
- RunTime monitoring working in parallel with logging system, collecting infos while the job still run
- Logging is stored in a DB local to the user
- Logging info comes from selected monitoring info and/or journal file retrieved at the end of the job with job output
- Monitoring stored in different DB
- Has limited lifetime (deleted once the logging info updated)

B(GR)OSS++ architecture (by Claudio G.)

- On UI: Boss client which talks with Boss DB
- Boss talks with scheduler (submission, control, query, output retrieval . . .)
- Submit boss–wrapper to scheduler
- Wrapper executes user process on WN
- Wrapper start logging process (write to Boss journal) and optionally Real-Time Updator
- RealTime Updator is a client of a real-time database
- Send info of the journal (possibly via web proxy) to real-time DB server (if possible)
- When job finished, output and journal retrieved and Boss DB updated
- Boss client optionally *pops* info of running jobs from Real Time DB, and purge it at the end

B(GR)OSS++ architecture (by Claudio G.)



Summary

- Usable CRAB released: under user test
  - Need PubDB $>$ 2.8 properly linked with RefDB
- Updated PubDB (with `FileType`) under test
- Crab–Gross integration plan agreed
  - need mode detailed specifications
- B(Gr)oss new architecture presented