APROM

CERN, Monday 26 November 2004

Report on WorkLoad Management activities

Federica Fanzago, *Marco Corvo*, **Stefano Lacaprara**,
Nicola de Filippis, **Alessandra Fanfani**

Stefano.Lacaprara@pd.infn.it

INFN Padova, *Bari*, *Bologna*

Outline

- Catalogs validation tools
- Update PudDB
- Submission tool
  - Data discovery
  - Integration with new RB
  - `.orcarc` creation
- Test plan

Catalog Validation tools (Nicola)

- Goals:
  - Creation of local POOL catalogs for dataset (Hit, Digi, PU + DST) with attached or virgin META
  - Retrieve relevant information from RefDB
  - Check file access (posix or RFIO)
  - Optional attach runs to virgin META and fix collection
  - Produce separate catalogs for Meta and EVD
  - Validation of catalogs via ORCA executables
  - Creation of POOL MySQL catalog
  - Publication to PubDB
- Two shell scripts with functions
- At the end provide following infos, taken **from validation procedure** *i.e.* **real data access!**:
  - List of attached runs
  - First and last run
  - Total number of attached events

Catalog Validation tools (Nicola) II

- Available at:

  `http://webcms.ba.infn.it/cms-software/orca/index.html`
- `Publish_dataset_v10.sh`
- Usage:

  `./Publish_dataset.sh <dataset> <owner>`
- Used successfully at:
  - All catalog in **Bari** published and validated
  - Many in **Bologna** and **CNAF**
  - Used at **LNL**
  - Partially in **Pisa** and **Firenze**
  - Used in **FZK**
- Future:
  - Publish catalogs for subset of runs
  - Publish catalogs for private production
  - Possible integration with PhedEx, as last step after dataset transfer

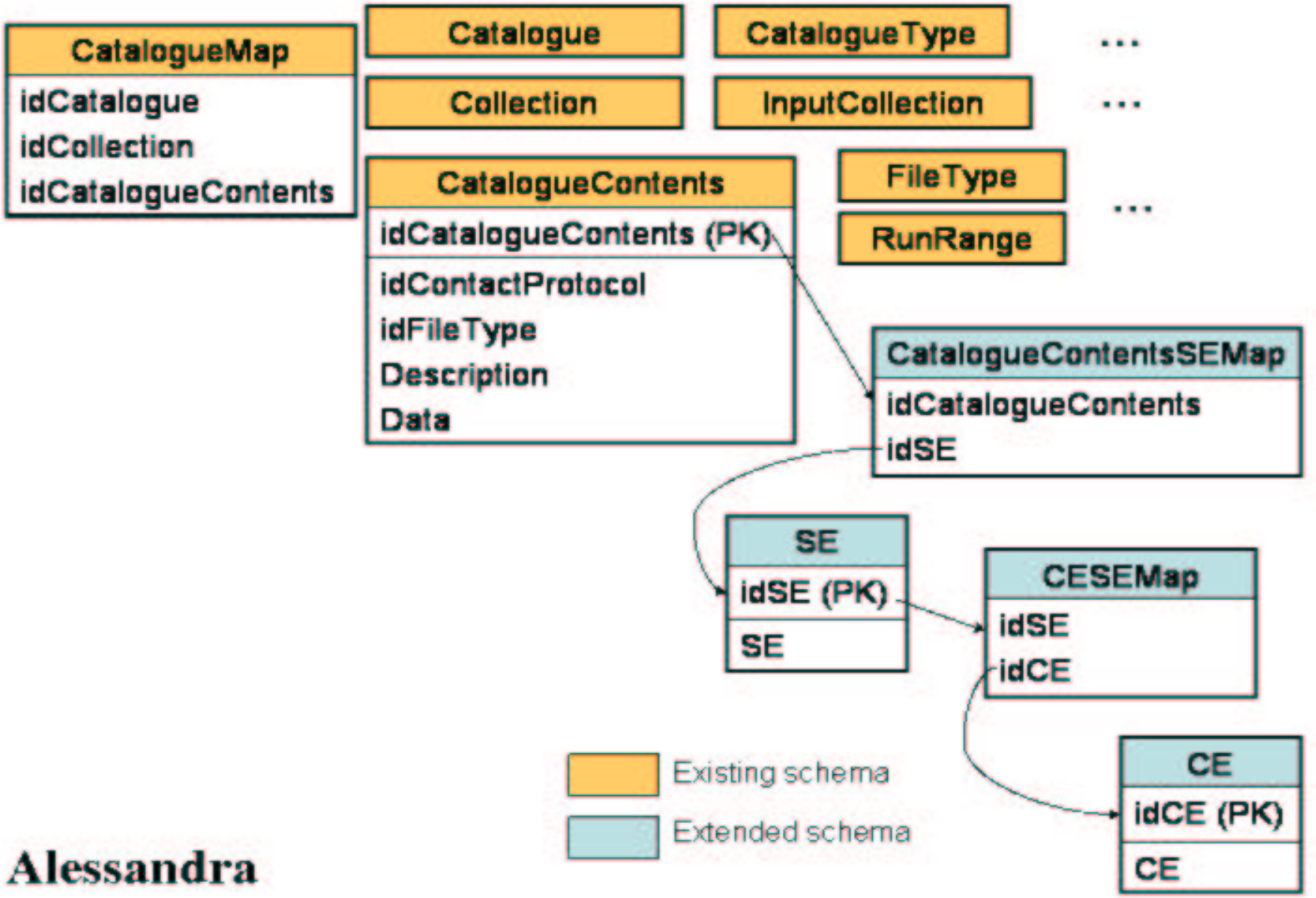
PubDB (Alessandra and Nicola)

- Create a *dev* PubDB to implement and test new schema and functionality
- Deployed Bo, Ba, LNL, FZK
- Soon linked with RefDB
- Update PubDB schema:
  - Total number of events in RunRangeMap
  - SE for each published catalog, as decided during last CPT week
  - Put *also* mapping between SE and CE
    - Technical reason: bug in LCG matchmaker to find CE from SE!
    - Assume that CE is just the *local* CE (Bo$\leftrightarrow$Bo, ...)
    - Easier life for `jdl` creation: put CEs as requirements

PubDB (II)

- Also new PHP to return all the info relevant for submission
- Modify PubDB command line tool to allow site manager to directly add information in PubDB
  - FileType
  - Validation status (VALIDATE, NOT_VALIDATED)
  - SE and/or CE
  - Run range, number of events
- It is rather boring to fill PubDB by hand
- Need a higher level tool to extract information and update PubDB
- Validation tool is a perfect candidate!

PubDB new schema (Alessandra)



Submission tool (Federica, Marco, SL)

- Use *agrape* framework as starting point
- Proposed new name ***CRAB***: Cms gRid Analysis Builder
- Already presented by Federica in past APROM/WM meetings
- Now integrated with new PubDB for dataset discovery and `.orcarc` preparation
- Dataset Discovery:
  - Query to RefDB to get CollID from Dataset/Owner provided by user
  - Get list of PubDBs publishing dataset/owner
  - Get contact strings for all PubDBs
  - Create `jdl` with requirements for publishing CEs
  - ```
    Requirements = Member((other.GlueCEInfoHostName ==
    "gridba2.ba.infn.it" || other.GlueCEInfoHostName ==
    "cmsboce1.bo.infn.it");
    ```
    ```

Dataset Discovery

UI

crab.cfg
dataset = ...
owner = ...
events = ...

Dataset/Owner

RefDB

CollID, PubDBs

PubDB

FileType = Complete
ValidationStatus = VALIDATED
ContactString= ...
ContactProtocol = http
CatalogType = xml
SE = ...
CE = ...
Nevents = 10000
RunRanges = 1−5

FileType = META
ContactString= ...
....

CRAB create JDL using CE as requirements

Submit to GRID

Dataset Discovery (II)

- Integration with new RB from Heinz (w/o “t”)
  - Trivial!
  - Just put `lds:/Owner/Dataset` as requirement in the jdl
  - New RB does the matchmaking
  - Still need to query the RefDB and PubDBs to get contact strings etc for local access
  - Would be better to get this info once on WN (query just one PubDB)

Private code preparation (Marco)

- User specify executable within a `scram` area
  - Use `$LOCALRT` to get actual ORCA version used
  - Put into the `jdl` as requirement for CMS software

```
Requirements = Member("VO-cms-ORCA_8_6_0",
other.GlueHostApplicationSoftwareRunTimeEnvironment)
```

  - Get executable and private libraries via `ldd executable`
  - Pack `tar-ball (tgz)` exe and libs
  - Send tgz via sandbox
  - Registration to SE foreseen (with automatic versioning)
  - Possible issue about deletion from SE after usage: who?
  - When all jobs finished (and output merged): eventually up to $x\%$

orcarc preparation

- User specify in crab.cfg the `.orcarc` used locally
- User specify also events per jobs and number of jobs
  - crab remove from `.orcarc` local `FileCatalogURL`
  - modify according to job splitting input
  - actual splitting done by skipping n events
  - Fine for complete dataset (actual scenario)
  - Not correct if incomplete dataset
  - Must use RunRange information to do splitting
  - Job splitting *per run*
  - Need to understand how to use RunRange
  - defining the first and last run for collection with discontinuous run ranges can became really complex

orcarc preparation (II)

- Complete `.orcarc` is generated by job wrapper script on WN
- Use info previously retrieved from PubDB at UI
- Infos for all sites send via `sand_box` to WN
- In WM substitute ContactString with correct one according to site chosen by RB
- Eventually first copy locally the catalog if needed (contact protocol is RFIO or SRM ...)

orcarc preparation (III)

- By far the most complex operation
  - Depends on how the catalogs have been published
    - One catalog for everything
    - One for META, one for everything else
    - Digis separated from Hits
    - etc . . .
  - For each combination, must deal also with catalogs published with different protocol
  - If Digi separated from Hits, must also get ancestors info
  - Go back to RefDB, with ancestor CollID, go again to PubDB, get entry for ancestors catalogs....
  - Situation can become very complex if we have to deal with all possible combinations!

orcarc preparation (IV)

- As discussed and agreed last CPT week, can simplify a lot if:
  - META catalog separated, accessed via web
  - One master copy at CERN, META replicated (by hand or via web proxy) at sites
  - Catalog for all other EVD in just one place!
  - MySql catalog for all *events* data
- `.orcarc` much simpler!
- Just two entries for FileCatalogURL: (mysql and web)
- Few other cards for Location variables

orcarc preparation (IV)

- FileType issue
  - Must define clearly FileType entry in PubDB to understand what can be accessed using a given catalog/set of catalogs
  - User need also to specify which data wants to access: DST, Digi, Hits, PU: eventually all
  - Job creator matches requirements and available catalogs to build a complete set
  - Catalog should publish:
    - attachedMeta,
    - virginMeta,
    - DST,
    - DIGIS,
    - HITS,
    - PU,
    - plus combination (DIGIS+HITS+PU)

orcarc preparation (IV)

- Plan to have tool to provide for a given site (PubDB), Dataset/Owner:
  - `CatalogStageIn()`
  - To be issued before job start, to copy (if needed) the needed catalogs locally
  - `.orcarc`
  - Complete and correct for given site/dataset/owner, provided the init script is issued
- Can be very useful also for local user
- Can be a common component for any submission tool
- Difficult to have a single `.orcarc` usable everywhere
- Future
  - Local event file catalog can be a local RLS
  - Integration with grid data location and replication services
  - Data access guaranteed by grid protocols and tool

CRAB flow

- Three basic operations:
- Can be done in one shot or in three separate steps (eg if you want to check the script before submission)
- `crab` keeps track of declared/created/submitted/retrieved jobs
  - Job creation
    - Dataset discovery
    - Query to Ref/PubDB
  - Job submission
    - Submission done via `edg-job-submit`
  - Job monitoring and output retrieval

Monitor/output retrieval

- Job monitoring
  - Monitoring done via grid command (direct usage of grid python modules)
  - Simple and quick status of jobs
  - Also possible to use JAM (Giacinto and Marcello)
  - Scalability test with $\mathcal{O}(1000)$ jobs successfull
- Output retrieval
  - Output retrieval done automatically once job finished
  - User can declare an output file (*eg* root file)
  - Output name modified according to splitting (_1)
  - Foreseen *merger* script to merge splitted output: at least for simple cases
  - Also possible to **store** output into a **SE**, with registration in RLS
  - Foreseen also for merged jobs

Test plan

- `crab` already tested by developers in Pd, Ba, Bo
- Very rapid development cycle
- First usable release yesterday
- Still very *beta*, but successfully used and tested
- Need new PubDB schema **and** publication of datasets with all infos into PubDB
- Available at Bo, Ba, LNL (still some technical problem . . .), FZK
- Need link from `refDB` for real test (today done via hard-coded link)
- Distribute to a set of beta tester: already a list of volunteers (tell me if you want to be one)
- Create CVS area to ease parallel development and versioning