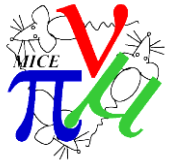


CDB and Batch Processing

Janusz Martyniak, Imperial College London

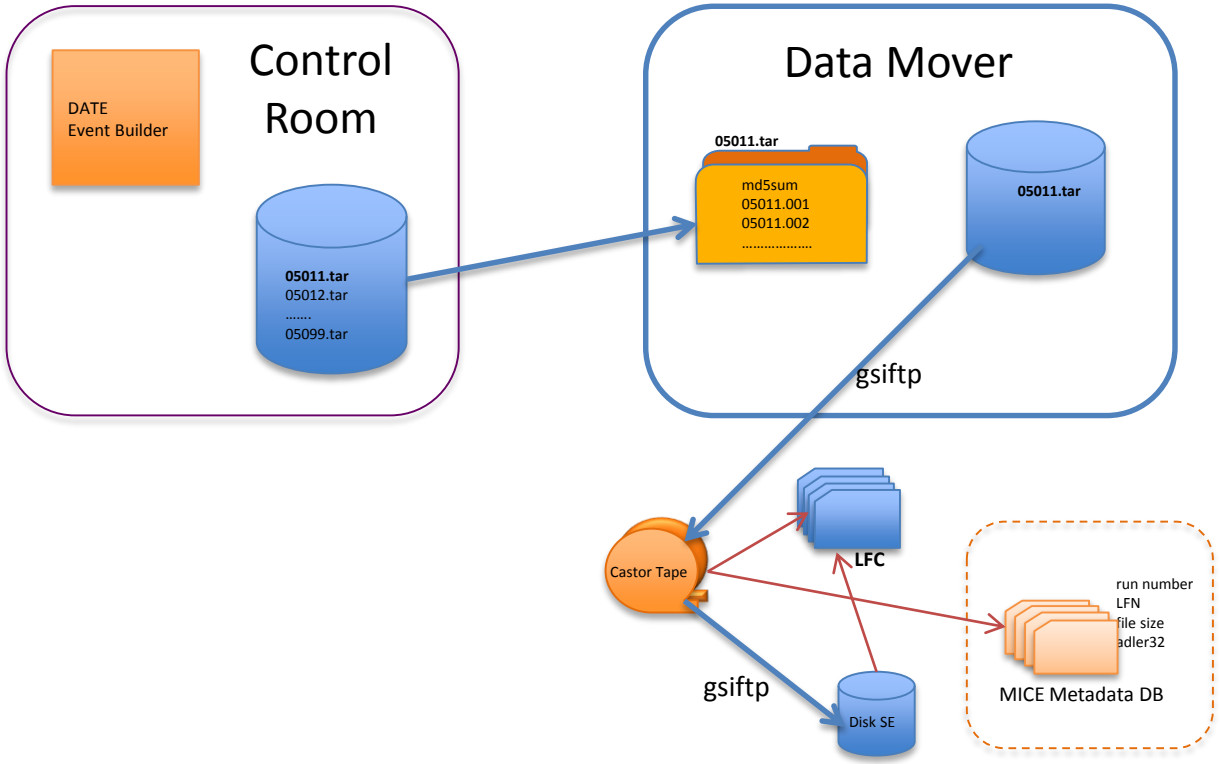
MICE CM42 Analysis, Software and Reconstruction

Raw Data Mover (2!)

- The Raw Data Mover has been upgraded
- Runs on a new SL6 box with EMI UI 3.x
- Ugly system command executing has been removed
- Uses EMI new GFAL2 Python API – the code is now more “pythonic”.
- Successfully tested during mock runs last fall
- Runs w/o major interruptions during current data taking

Raw Data Mover Workflow

(as a reminder)



Raw Data Mover

Summary

- Developed in summer 2011, in use since Oct 2011,
- Rewritten in 2014 to use GFAL2
- Successfully used to store data on Castor to date,
- Uses up-to-date Python API,
- Use robot certificate, currently my personal,
- Still not using the dongle

MAUS Reconstruction on the Grid

- GRID MAUS installation via CVMFS at RAL,
- Propagated to Brunel, Imperial and the PPD
- Build and compiled on SL6, last build used is MAUS_v0.9.4, last built and installed - 0.9.5
- All current data being reconstructed with this version, no major glitches
- No major changes to the framework (see my talk for CM37)

Configuration Database

project status

- We maintain a primary CDB server in the Control Room, including a supermouse WebService front end.
- A hot standby in the PPD which mirrors TH primary DB, read only
- A preprod DB with write access

Configuration Database (contd.)

An extension to the server API:

- Batch Iteration number which allows storing multiple MAUS data cards for the same MAUS version,
- MC serial number
- Server-site code extended
- Python API for the client written

A few bug fixes have been done on the server (Calibration, DAQ Cabling and Tracker calibration)

Encoding problem fixed on preprod, so BLOBs are decode properly now.

Configuration DB

Request from Pierrick to provide API to store/retrieve data for the cooling channel.

- Relevant table existed in the DB,
- The Web Service interface is deployed, Python API exists.
- The C-API has been written using gSOAP binding, unit test using Cunit. Ready since last year and rusting.

Configuration DB (contd.)

Latest development - Beamline a.k.a Run Control C-API

- Python API cannot be used due to problems with multithreading
- Reverse-engineered the Python API and C equivalents written.
- As for the Coolingchannel C-API gSOAP C binding is used,

Beamline CDB API

- A set of C structs has been created to hold beamline data both for getters and setters
- A set of ‘constructors’ (default and custom) has been written to facilitate API usage. The destructor exists as well.
- A set of ‘push’ functions is provided to simplify creation of linked lists (like the magnet list).

Beamline CDB API

- The API is 90% ready, one low priority ‘get’ function not yet complete.
- Uses XML schema to validate data supplied by a user
- CUNIT tests exist
- Documentation exists (doxygen)
- Project on Launchpad:

bzr+ssh://bazaar.launchpad.net/~janusz-martyniak/mcdb/mice.cdb.client.api-C/

CDB C API

In numbers ..

- ---
- Language files blank comment code
- ---
- C 83 739 1583 4468
- XSD 7 77 58 515
- C/C++ Header 12 90 129 402
- make 1 29 17 73
- ---
- SUM: 103 935 1787 5458
- ————————————————————————————————

Well commented ;) ?

CDB - Summary

- CDB is fully operational, bugs discovered have been fixed
- Server code Python API has been extended (Batch Iteration Number and MC Serial Number)
- C-API developed for the Collingchannel (not used yet), and Beamline (only roughly tested in the Control room)