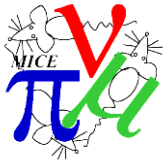


Configuration Database

Janusz Martyniak, Imperial College London

MICE CM45 Software and Computing

Configuration Database

project status

- Run Control (beamline) and Cooling Channel C API status
- Absorber table and APIs
- Geometry corrections API – status
- Data and reco quality flags API – status
- CDB viewer

Configuration DB (contd.)

Beamline C API

- The API has been written in C (gSOAP) to facilitate integration with run control code
- Successfully implemented for:

1. Retrieving beamline settings (tags)form the database
2. Storing beamline magnet data to the hardware
3. Storing the settings to the CDB for a given run
4. Storing new tags

- Project on Launchpad:

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

Beamline and Coolingchannel API – experience so far

- Used for the current run period
- The current version of the setters allows for a client-side logging of both request and response messages to/from the CDB (for debug purposes)
- Run control writes run start information, followed by beamline magnet data, coolingchannel magnet data and absorbers settings (in this order)
- At run end a termination run record is written to the DB
- We should still decide how to handle exceptions thrown by any of the steps above to decide if we should avoid orphaned Coolingchannel data for example (probably not …).
- CDB only disallows writing run end record if no run start has been written. Magnet data (both bl and cc could be written regardless)

Beamline/Coolingchanel API
error handling

Every operation which writes data to the CDB returns a structure, which contains:

- Return code (int, 0 means OK)
- Return XML message for logging or display
- Request XML message for client-side logging

It is **extremely important** to verify that the ret code is 0, failing to do so might leave us with incomplete run information in the CDB.

The logs (client and server side) will allow to retrofit missing information. We have done it more than once (!) already…

Cooling Channel CDB API
an important design change

- Original Coolingchannel database table was timestamp based. It was linked to a run number it was valid for by a rather complicated time comparison, leaving to wrong results if the CC was written after the beamline.
- It was not easy to fix for a case of (possible) missing run end info or orphaned CC records, to name just those 2 cases.
- We have decided to drop this inconvenient design and link coolingchannel data to run number directly (obvious, no ?).
- This required passing a run number from the Run Control CC module with the CC call.
- The server side has been changed to use a new table and fall back to the old one for older runs (run number < 7929)

Coolingchannel Absorber API

- Absorber related server side operations added to the existing Coolingchannel WebService
- C API extended to allow storing/retrieving absorber tags and writing absober data to the CDB
- Python API written to read/write absorber tags and data to the CDB
- Implemented in the Control Room (less tag writing)

Absorber data record

- We are writing/reading:

1. absorber name
2. material
3. shape
4. temperature
5. pressure
6. comment

The table is indexed by run number.

Geometry Corrections
(issue #1817)

- For a given geometry ID we are storing following data for a geometry module:

module name (string)

dx(float)
dx_err (float)
dy (float)
dy_err (float)
dz (float)
dz_err (float)
dx_rot (float)
dx_rot_err (float)
dy_rot (float)
dy_rot_err (float)
dz_rot (float)
dz_rot_err (float),

Comment (string)

Geometry Corrections API (Python)

Corrections may be stored in the DB by using one of the Python functions:

- ***set_corrections(modules, geometry_id, comment=")***

Where *modules* is a list of dictionaries:

modules=[{'name':'TOF0','dx':12.1,'dx_err':0.01, 'dy':22.20,'dy_err':0.02, 'dz':32.30, 'dz_err':0.029,.....}, {.....}]

or

- ***set_corrections_xml(corr_xml)***

Where *corr_xml* is:

"<GeometryID value='12'>
  <ModuleName name='TOF0' dx='12.1' dxerr='0.01' dy='22.2' dyerr='0.02' … />
  <ModuleName name='TRACKER0' dx='12.4' dxerr='0.001' dyerr='0.002' … />
</GeometryID>"

There are 2 methods to retrieve the corrections, by run or by geometry id:

- ***get_corrections_for_run_xml(self, run):***
- ***get_corrections_for_geometry_id_xml(self, gid)***

Both return an XML document. Thedocument format is identical to one listed above.

Data Quality and Reco Flags
(in progress)

Store and retrieve bit coded data quality and detector flags. Implemented for reco only.

- *get_reconstruction_flags(run_number, maus_version, batch_iteration_number)*

Return Python dict keyed by detector name. For bits for detector.

- *get_reconstruction_flags_for_detector(detector, run_number, maus_version, batch_iteration_number)*

Return a flag (4 bits) for a given det. It uses the call above.

- *set_reconstruction_flags(run_number, maus_version, batch_iteration_number, flags)*

Set flags for all detectors.

CDB Viewer!

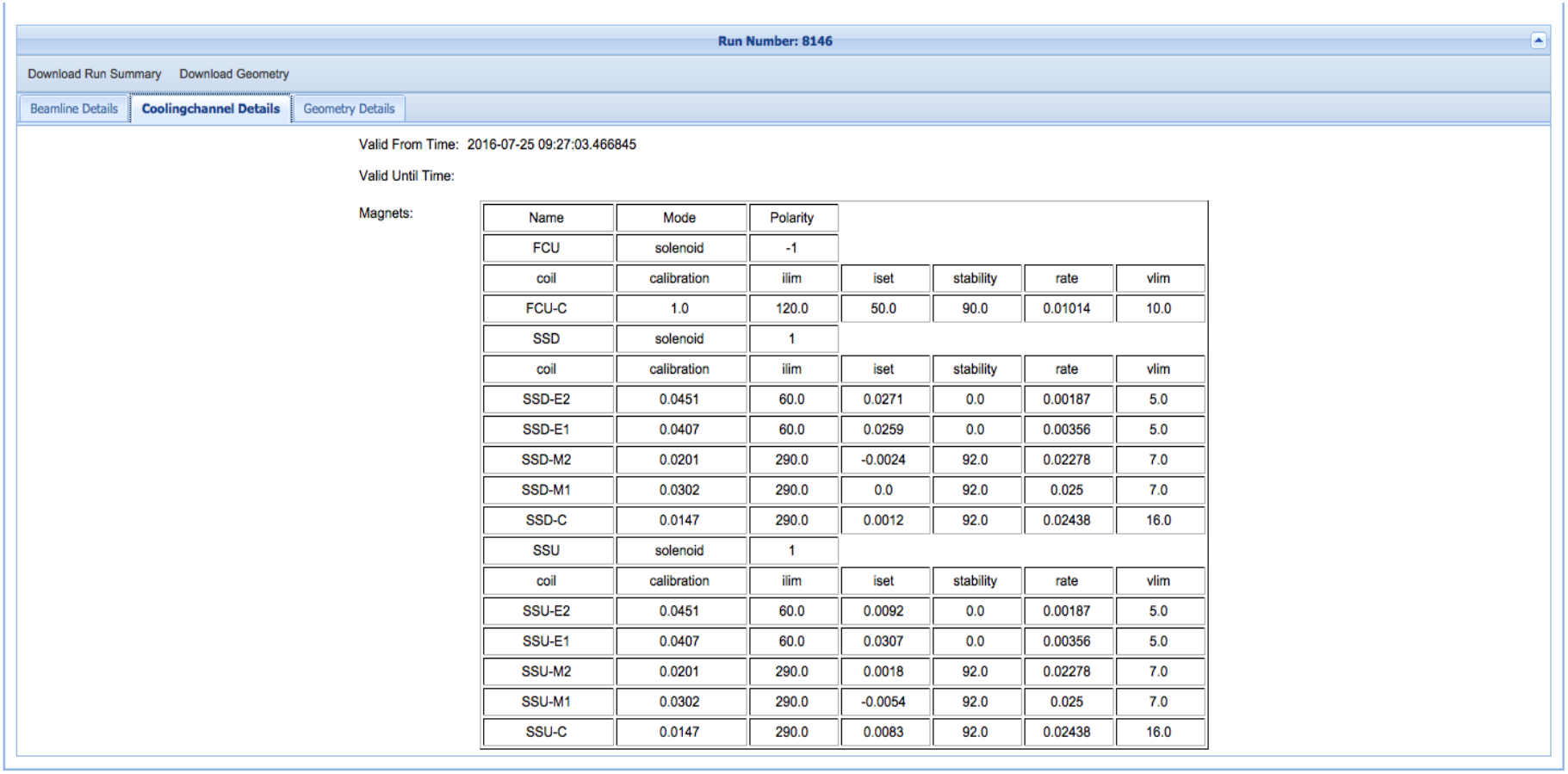
- A WEB service installed on the same machine (Tomcat) as the public CDB Web Service
- Uses GWT (Google Window Toolkit) to present data in a Web browser
- Pure java, compiles client stuff to javascript…
- The server side of the viewer uses CDB java client calls (*getBeamlineForRun()* for instance) to contact the CDB service:

Browser->CDBviewerWS->CDBclient->CDBWS->DB

- Originally written by Antony Wilson and forgotten ever since ;-)

CDB Viewer

- Added Coolingchannel data and some Beamline improvements (http://cdb.mice.rl.ac.uk/cdbviewer/):



Run Number: 8146

Download Run Summary Download Geometry

Beamline Details | **Coolingchannel Details** | Geometry Details

Valid From Time: 2016-07-25 09:27:03.466845

Valid Until Time:

Magnets:

Name	Mode	Polarity				
FCU	solenoid	-1				
coil	calibration	ilim	iset	stability	rate	vlim
FCU-C	1.0	120.0	50.0	90.0	0.01014	10.0
SSD	solenoid	1				
coil	calibration	ilim	iset	stability	rate	vlim
SSD-E2	0.0451	60.0	0.0271	0.0	0.00187	5.0
SSD-E1	0.0407	60.0	0.0259	0.0	0.00356	5.0
SSD-M2	0.0201	290.0	-0.0024	92.0	0.02278	7.0
SSD-M1	0.0302	290.0	0.0	92.0	0.025	7.0
SSD-C	0.0147	290.0	0.0012	92.0	0.02438	16.0
SSU	solenoid	1				
coil	calibration	ilim	iset	stability	rate	vlim
SSU-E2	0.0451	60.0	0.0092	0.0	0.00187	5.0
SSU-E1	0.0407	60.0	0.0307	0.0	0.00356	5.0
SSU-M2	0.0201	290.0	0.0018	92.0	0.02278	7.0
SSU-M1	0.0302	290.0	-0.0054	92.0	0.025	7.0
SSU-C	0.0147	290.0	0.0083	92.0	0.02438	16.0

CDB - Summary

- CDB is fully operational, bugs discovered have been fixed, missing Coolingchannel data retrofitted.
- CAPI for Beamline implemented in the Control Room
- Cooling Channel API implemented in the Control Room
- Coolingchannel absorber API implemented in the Control Room (apart from storing tags – done from Python for a time being)
- State machine C-API written (user interface only, no supermouse) – not used so far
- Geometry corrections. Ready for testing.
- CDB Viewer V.2 installed (CC data display added)