CDB for detector operations

Urs Langenegger

2024/10/22

- Introduction
- DQM matters
- Detector configuration

Introduction

- **CDB - "Conditions" DataBase**
  - ▷ non-event data required for reco/analysis
    - status: good/bad/whatever
    - (mis)alignment
  - → additions to "Conditions" requested

- **RDB - run database**
  - ▷ MIDAS writes
    - BOR - begin-of-run record
    - EOR - end-of-run record
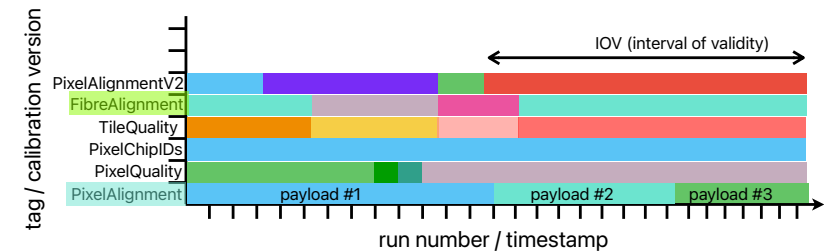   - many things to be added!
  - ▷ (online) DQM output

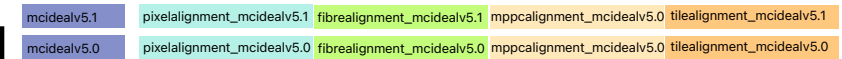- **detector configuration**
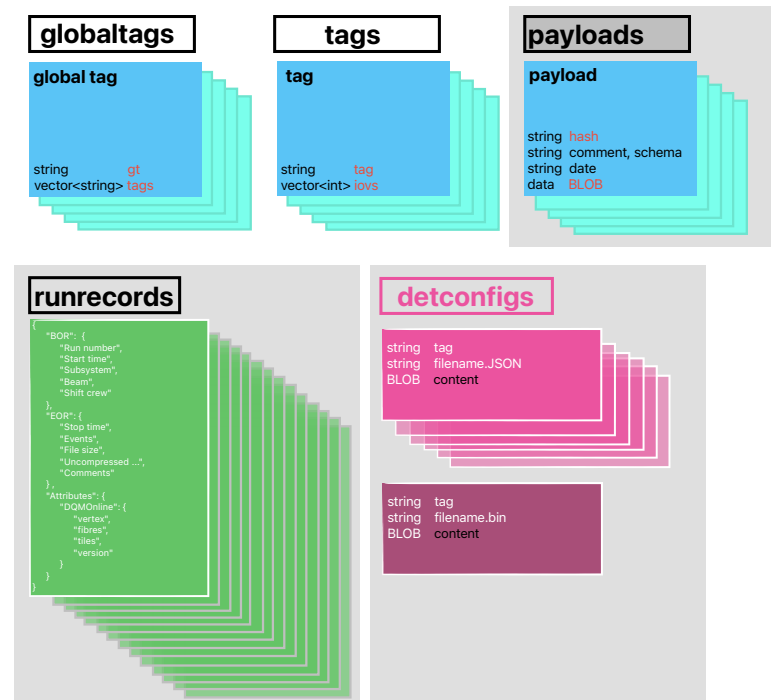 - ▷ first discussion yesterday

- **Within PSI network**
  - ▷ http://pc11740.psi.ch/rdb/
  - ▷ http://pc11740.psi.ch/cdb/findOne/globaltags/mcidealv5.1

Tags combine conditions/calibrations with intervals of validity

Global tag collects consistent set of tags

Database contents organization

RDB - run database

Updates from DQM

- **DQM output:**

  ```
  {
      "Attributes": {
          "DQMOnline": {
              "vertex": "1",
              "fibres": "1",
              "tiles": "1",
              "version": "2024/10/22"
          }
      }
  }
  ```

  - ▷ appending runrecord in CDB/RDB with "attribute"
    - • write a summary n quality levels for m subsystems
    - • possibly version of code
  - → write JSON file, invoke curl

    e.g. curl -X PUT -H "Content-Type: application/json" --data-binary @/Users/ursl/tmp/maskfiles/dqm.json http://localhost:5050/rdb/addAttribute/12345

  - ▷ updating mask files in CDB/detConfigs
    - • probably only store the "good" maskfiles
    - • after iterations, after "manual" (intelligent) validation
  - → write mask files, invoke curl
  - ▷ the "PixelQuality" record/payload in CDB/Conditions
    - • e.g. list of noisy channels
  - → generate payloads, invoke curl

- **Online DQM "harvesting" should not delay end-of-run DAQ transition**
  - ▷ better DQM summary from prompt-reco/offline

- **Code version should be recorded**
  - ▷ DQM, because of quality criteria changing over time
  - ▷ DAQ s/w version documented where?

Detector configuration

- **Discussion with pixel people**
  - ▷ versioned mask files (different startup settings/backup)
  - ▷ versioned configuration JSON (different startup settings/backup)

- **Implemented**
  - ▷ curl-based upload/download of JSON file
    - JSON compression $\approx 10^3$ into BLOB
    - curl/REST api without mongodb coupling
  - ▷ curl-based upload of binary mask files
    - curl/REST api without mongodb coupling
  - ▷ curl-based download of binary mask files for a tag
    - returns all files of a tag in one zip file

- **Notes for discussion**
  - ▷ no integration into MIDAS so far. Reading configurations from files!
  - ▷ in essence, versioned different startup settings/backups of files
  - ▷ one common JSON file? tagged JSON files (à la tags and global tags)?