Data Analysis Working Group

Metadata discussions: Belle II Global Tags

Paul Laycock

BROOKHAVEN NATIONAL LABORATORY

U.S. DEPARTMENT OF ENERGY

HSF Data Analysis Working Group Meeting - 19th January 2021

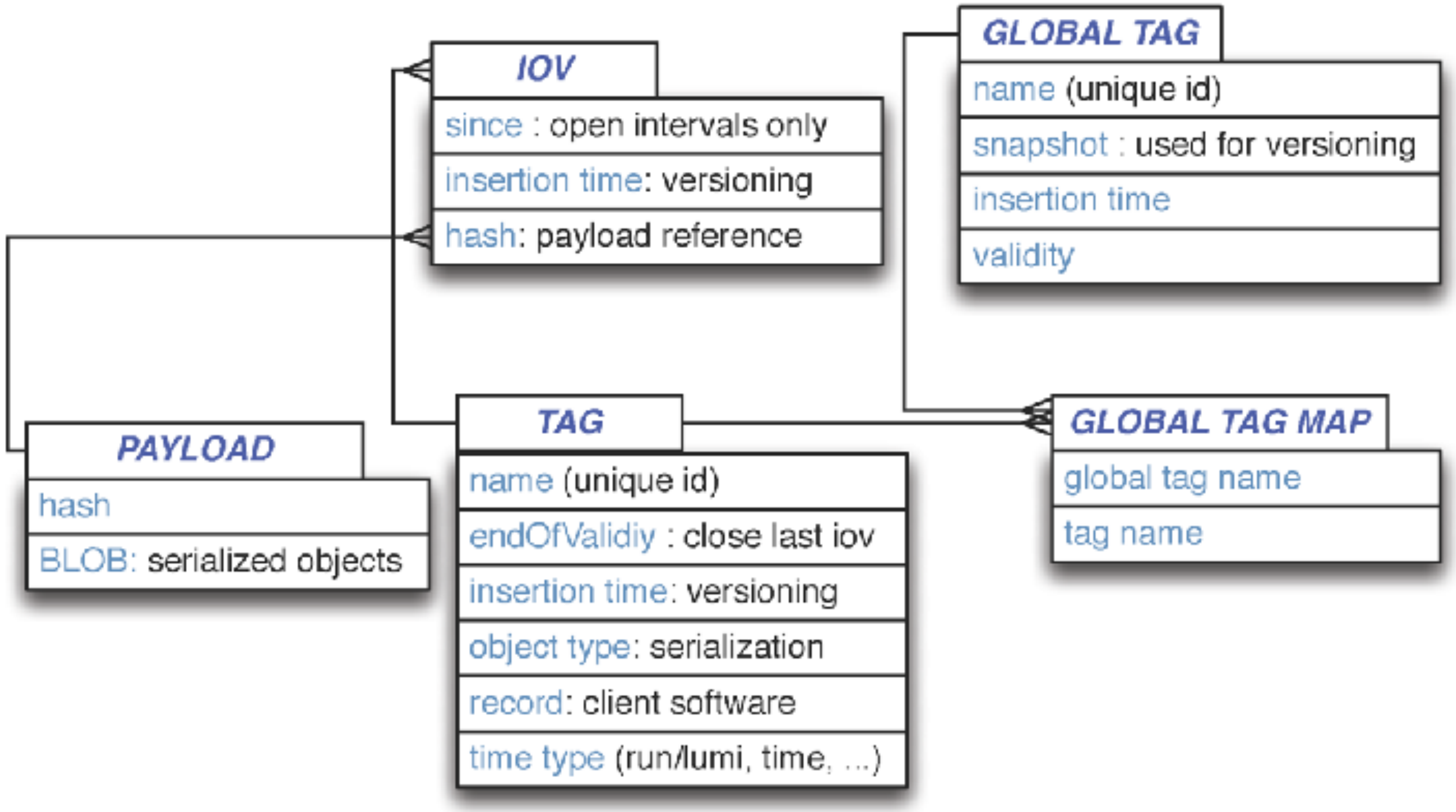
Introduction

- Disclaimers
  - Belle II physics started in 2019, so still ramping up in terms of software&computing complexity
  - ***I'm not doing analysis!***
  - Some ideas are very mature, well tested for several years
    - File production and conditions in particular
  - But…

- This is therefore a short talk!
  - Thanks to M. Ritter and S. Cunliffe for material
  - Apologies for any errors

Conditions Metadata

- Data Model: relational DB, a global tag tells you which payloads you need
- Single tables for payload, tags, IOVs (instead of one per system)
- IOVs and payloads resolved independently
  - Cache-friendly design
  - Payloads referenced by a unique hash, can be factorised as a file-service (it is for Belle II)


IOV
since : open intervals only
insertion time: versioning
hash: payload reference
GLOBAL TAG
name (unique id)
snapshot : used for versioning
insertion time
validity
PAYLOAD
hash
BLOB: serialized objects
TAG
name (unique id)
endOfValidiy : close last iov
insertion time: versioning
object type: serialization
record: client software
time type (run/lumi, time, ...)
GLOBAL TAG MAP
global tag name
tag name

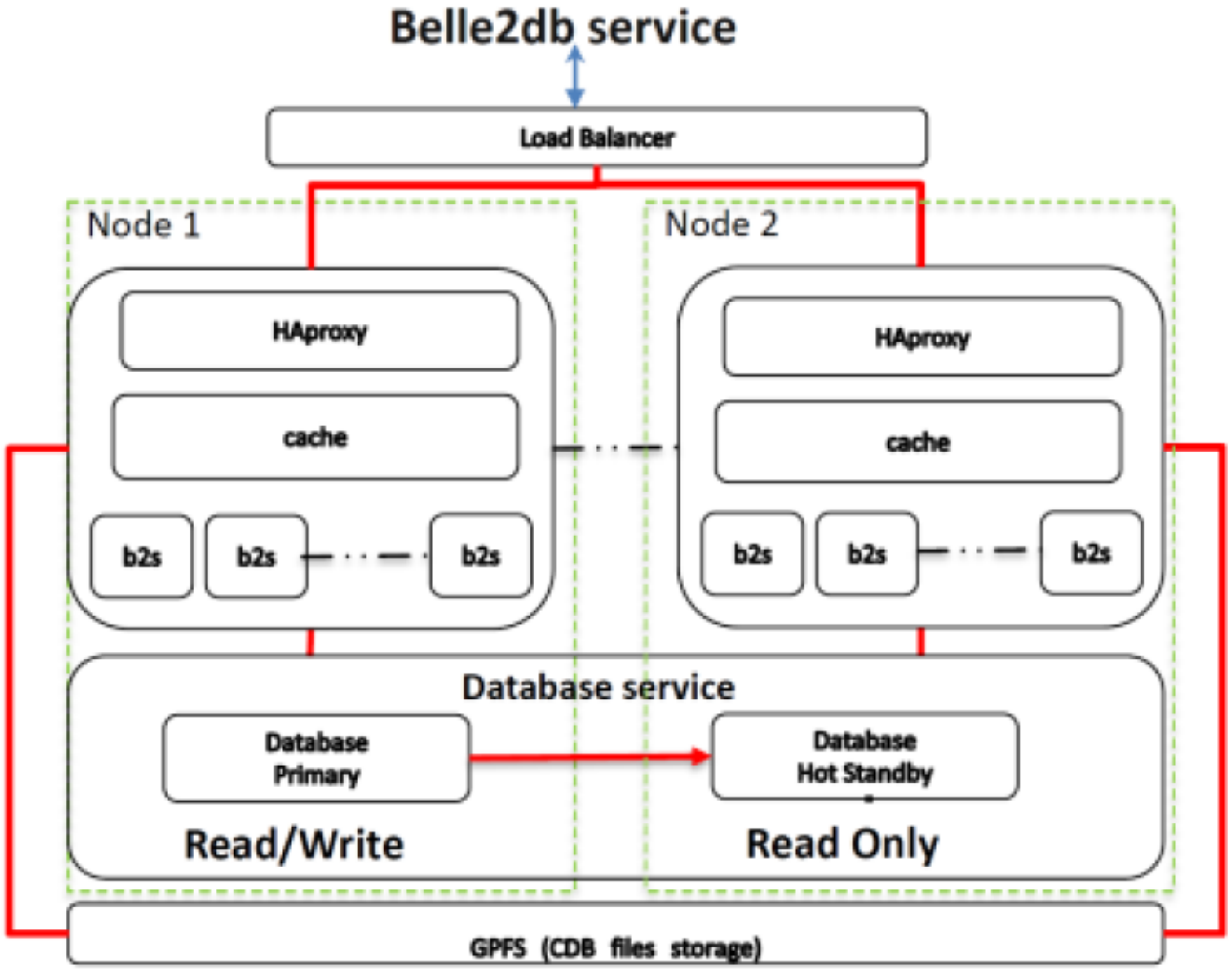
This is from the HSF conditions WG paper, Belle II schema is slightly different

Belle II Conditions Service(s)

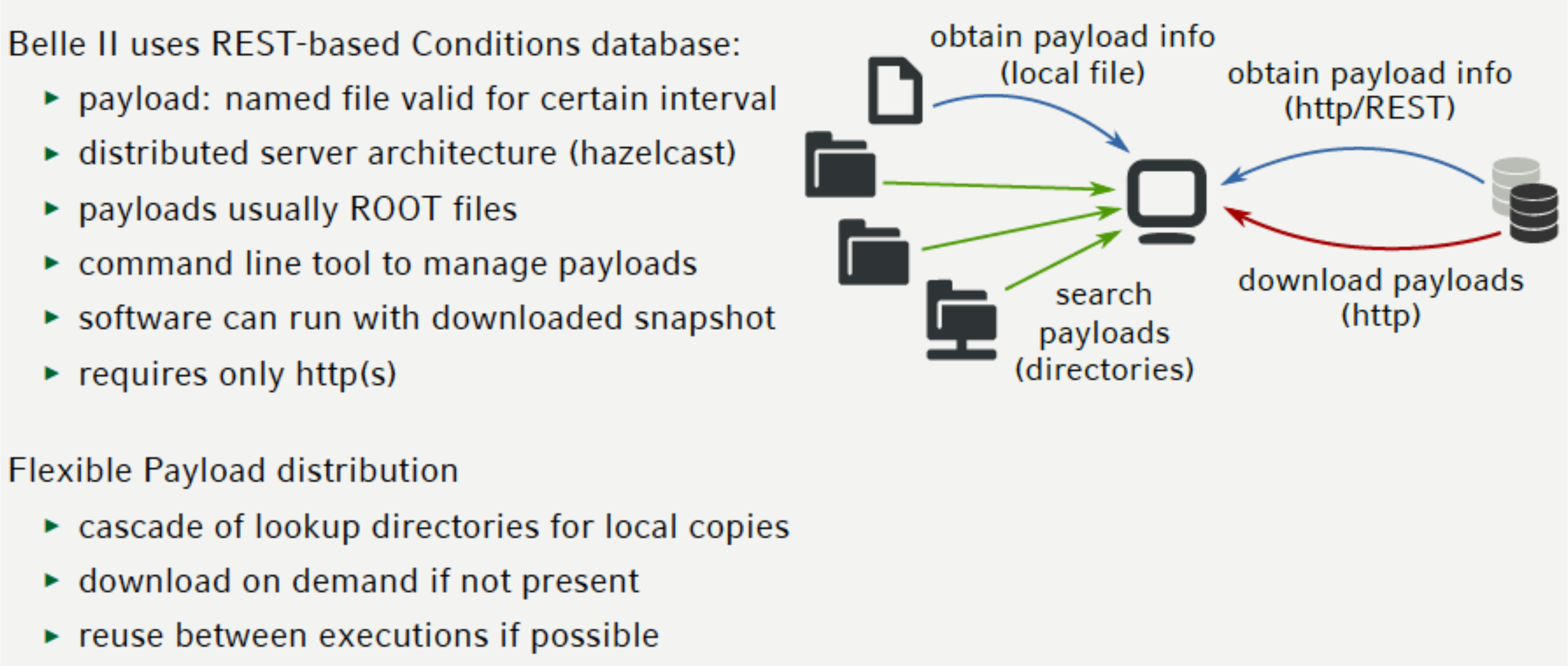
- Industry-standard components
  - **REST** interfaces

- ***Payloads entirely factorised***
  - Postgres for metadata service (that resolves global tags)
  - Separate file service for reading payloads (also using a REST interface)
- Both metadata DB and payloads are backed up to cvmfs
  - Payloads taken preferentially from there, and cached locally
  - Metadata DB in mysql files on cvmfs in case of service outage

- DB schema *largely experiment agnostic*
  - *Essentially blobs*



Analysis conditions



Belle II uses REST-based Conditions database:

- payload: named file valid for certain interval
- distributed server architecture (hazelcast)
- payloads usually ROOT files
- command line tool to manage payloads
- software can run with downloaded snapshot
- requires only http(s)

Flexible Payload distribution

- cascade of lookup directories for local copies
- download on demand if not present
- reuse between executions if possible

- From M.Ritter, analysis ecosystem workshop in 2017
- Analysis conditions are handled using the same conditions database
- List of global tags given to a job, now possible to merge global tags (active rather than passive resolution of potential consistency issues)

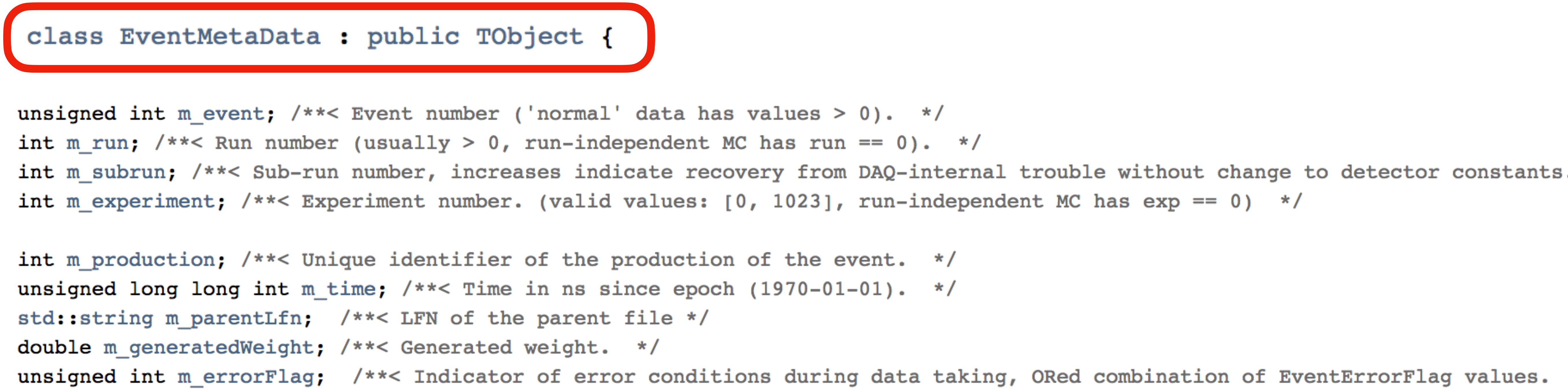
Analysis conditions II

- Currently have a list of global tags given to a job
    - Conflict resolution by order of preference
- Now possible to merge global tags, potential to actively resolve potential consistency issues

- Pros: Organise analysis conditions and in principle have a complete record of all of the conditions used in an analysis (merge the global tags)
- Copy on cvmfs is nice and easy to browse
- The service is http, so this works very much like cvmfs, not crazy to think of a POSIX interface to the CDB service itself

- Cons: Somebody mentioned the word database

In-file metadata - Event metadata

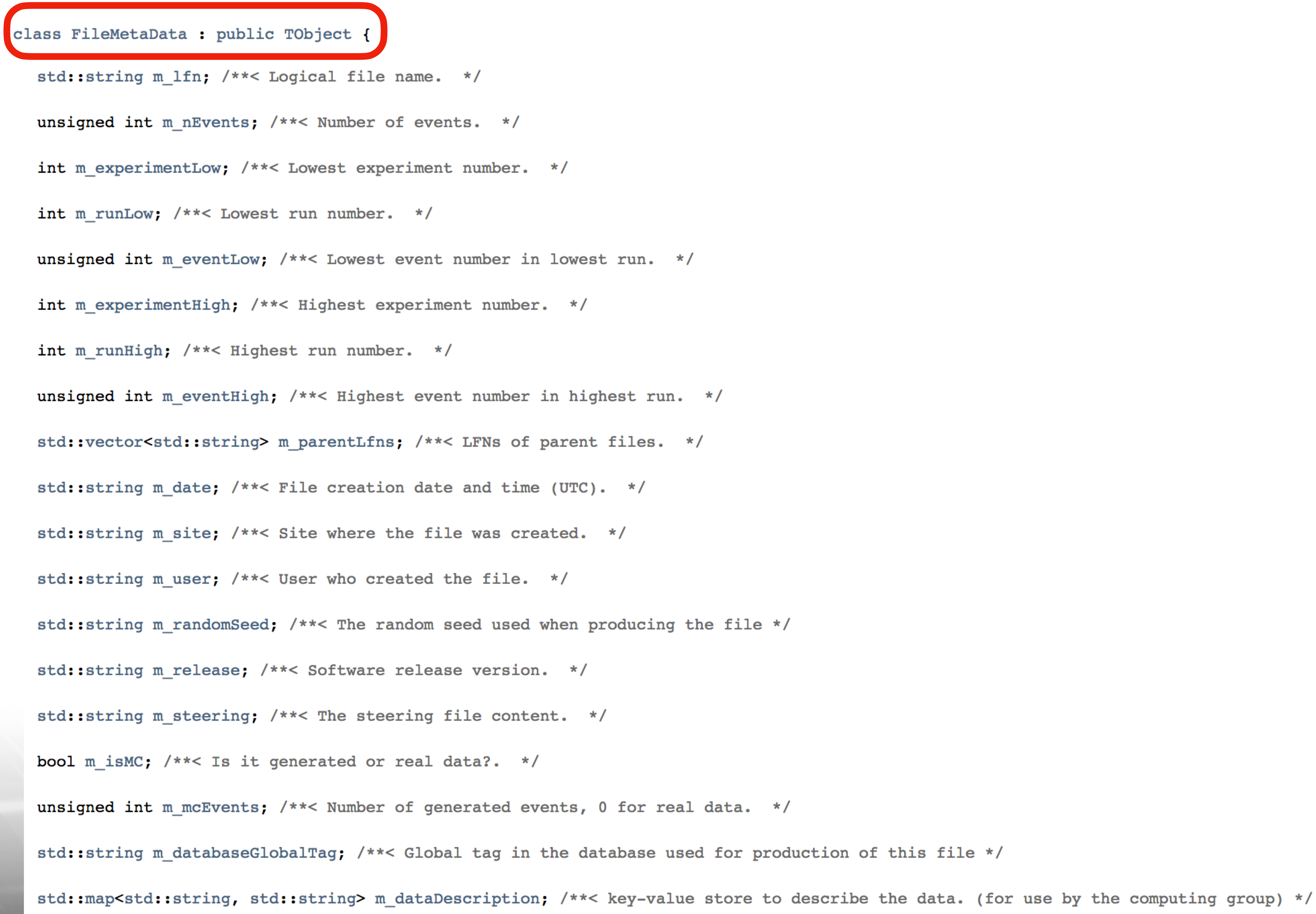


```
class EventMetaData : public TObject {

unsigned int m_event; /**< Event number ('normal' data has values > 0).  */
int m_run; /**< Run number (usually > 0, run-independent MC has run == 0).  */
int m_subrun; /**< Sub-run number, increases indicate recovery from DAQ-internal trouble without change to detector constants.
int m_experiment; /**< Experiment number. (valid values: [0, 1023], run-independent MC has exp == 0)  */

int m_production; /**< Unique identifier of the production of the event.  */
unsigned long long int m_time; /**< Time in ns since epoch (1970-01-01).  */
std::string m_parentLfn;  /**< LFN of the parent file */
double m_generatedWeight; /**< Generated weight.  */
unsigned int m_errorFlag;  /**< Indicator of error conditions during data taking, ORed combination of EventErrorFlag values.
```

- Simple TObject written into the file
- One class for “event” metadata
- Another class for file metadata

- File metadata master catalogue used in production is AMGA:
  - http://amga.web.cern.ch/amga/
- In addition, a dataset search tool which is fed from AMGA allows arbitrary keyword searches

In-file metadata - File metadata



```
class FileMetaData : public TObject {
  std::string m_lfn; /**< Logical file name.  */
  unsigned int m_nEvents; /**< Number of events.  */
  int m_experimentLow; /**< Lowest experiment number.  */
  int m_runLow; /**< Lowest run number.  */
  unsigned int m_eventLow; /**< Lowest event number in lowest run.  */
  int m_experimentHigh; /**< Highest experiment number.  */
  int m_runHigh; /**< Highest run number.  */
  unsigned int m_eventHigh; /**< Highest event number in highest run.  */
  std::vector<std::string> m_parentLfns; /**< LFNs of parent files.  */
  std::string m_date; /**< File creation date and time (UTC).  */
  std::string m_site; /**< Site where the file was created.  */
  std::string m_user; /**< User who created the file.  */
  std::string m_randomSeed; /**< The random seed used when producing the file */
  std::string m_release; /**< Software release version.  */
  std::string m_steering; /**< The steering file content.  */
  bool m_isMC; /**< Is it generated or real data?.  */
  unsigned int m_mcEvents; /**< Number of generated events, 0 for real data.  */
  std::string m_databaseGlobalTag; /**< Global tag in the database used for production of this file */
  std::map<std::string, std::string> m_dataDescription; /**< key-value store to describe the data. (for use by the computing group) */
```

Conclusions

- Analysis conditions utilising the conditions database infrastructure is arguably a good idea!
  - Consolidating as much metadata as possible under one roof (global tag) drastically reduces the entropy

- File provenance et al are well exercised

- Metadata for analysis provenance, reproducibility and ***full analysis chain*** are work in progress
  - This was alleged two years ago, there is now a Data Preservation Task Force coming back to this and related questions
  - Far from trivial, interesting to understand how other experiments tackle this