What I understand about the Raw Data Model (with inspiration from Zhe's 4148 title)

Brett Viren

Physics Department



2009/11/2

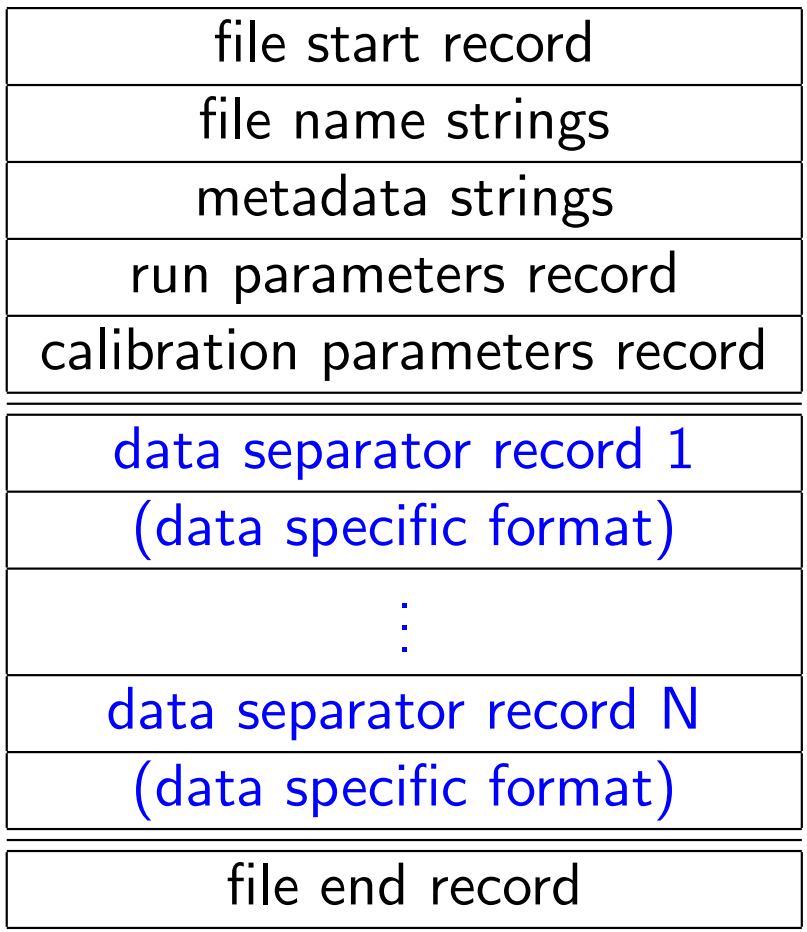
What is this about?

I will try to explain some design issues, as I see them, involved in our Raw Data Model.

I focus on issues involving real data from DAQ, with one small comment on simulation.

I do not discuss the Data Model for run-level meta-data as this has been in place since last summer (DataModel/RunData).

Raw Data Blocks



Raw data file format

- DocDB 3442-v6.
- Format has two parts:
  1. High level meta-data
  2. Low level DAQ data specific (one example: 3618-v3)

Calibrated Data

We **really** want to play with Calibrated data more than Raw data.

For each **hit PMT** we want a measurement of:

time: absolute time of discriminator crossing

charge: number of PEs collected

- These are "true" values that we estimate by measurements.
- These values should be (by construction) independent from details of Raw data (ie, systematic effects).
- It is the calibration algorithm that must adjust to (and hide) changes in raw data schema. Other users do not "see" these raw data details.

How to get there

We need an intermediate "raw" data model

- Model must expose all details of the raw format.
- Provides input to calibration processes.
- If possible, would like to chose an "optimal" model.

What is **optimal**?

Possible issues we will face: format change

Consider: Online group develops improved algorithms
$\Rightarrow$ DAQ data-specific format changes.

How does Offline respond when a change occurs

- **before** data in new format is processed?
  - Option to change raw data model in sync with new raw format files
- **after** data in new format is processed?
  - Must go back and reprocess the data at some level

Our Raw Data Model has implications as to how we respond to a format change.

- Changes most likely to come **during commissioning** and **early running**.
- Under **strong time constraint** to produce initial θ_{13} measurement/limit from first 6 months of data.

Two approaches

So far there are **two, different approaches** to interpret the raw data file format. They differ as to **how/where** the raw format schema is expressed:

- Unpacked, Data Member Schema (RawEvent)
  - Unpack raw block, copy result into data members.
  - Data Model schema enforced by data members.
- Packed, Object Method Schema (RawData)
  - No copy, unpack raw block from object methods.
  - Data Model schema enforced by object methods.

Both (can/should) have the same user-visible API.

Implications to format change

What happens when the **raw file format changes** and **data model has not**?

Unpacked, Data Member Schema

- New format copied into old model object members.
- Underlying data members are now corrupt.

⇒ Must update code before writing.

Packed, Object Method Schema

- Underlying data is untouched and uncorrupted.
- Old data model methods return garbage.

⇒ Can update code after writing.

Packed, Object Method Schema approach provides more flexibility when responding to raw data format changes.

Implication to complexity

What implications are there to code complexity?

Unpacked, Data Member Schema

- Changes to format require changes to converters.

Packed, Object Method Schema

- No changes to converters, I/O just sees simple blocks

Object Method Schema allows for simpler code.

Summary of Raw Data Model Design Issues

- Two approaches considered for raw data model:
  - Unpacked, Data Member Schema
  - Packed, Object Method Schema
- Both expose sufficient details of raw format to provide input to calibration.
- No significant benefits to Unpacked, Data Member Schema
- **Packed, Object Method Schema** is more **optimal** in terms of flexibility and simplicity in the face of changing formats.

Final Note on Simulated Data

How to handle simulation?
Choices:

1. Writing (fake) packed byte stream from SimReadout
   - Downstream of byte stream (and Readout) is same for real and simulated data.
2. Skip raw format, Readout directly "unpacks" SimReadout
   - `IReadout` interface with separate implementations for (a) consumer of byte stream and (b) SimReadout.
   - Downstream of (just) Readout is same for real and simulated data.

- Both handle new and old simulation files.
- Neither have implications on Raw Data Model.
- Simon P. is looking into these issues.

Personally prefer #1 but maybe a little more work.