Analysis of BS, ESD and AOD data

- What trigger data is available offline and what data should be available?
- How are we going to analyse HLT data?

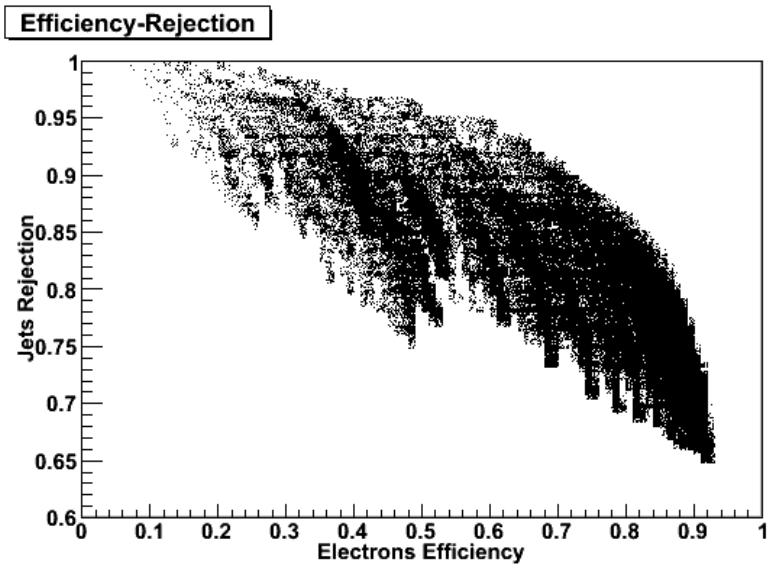
Motivation

1) Trigger signature/menu study and optimisation:
   - Should be done before including signatures in online menu (i.e. signature requests should normally be justified by studies)
   - Check trigger decision/objects against truth information
   - Can be validated by re-running trigger on force-accepts and prescaled loose-threshold events

2) Check trigger operation:
   - Offline: compare with truth to get efficiency/purity, improve reconstruction, estimate bias
   - Testbed studies: trigger behaviour must be the same “online” and offline ⇒ compare Bytestream and ESD for the same generated events
   - Online: study trigger operation in force-accepted events and prescaled signatures; debug aborted events (diagnostic stream); pathological events…

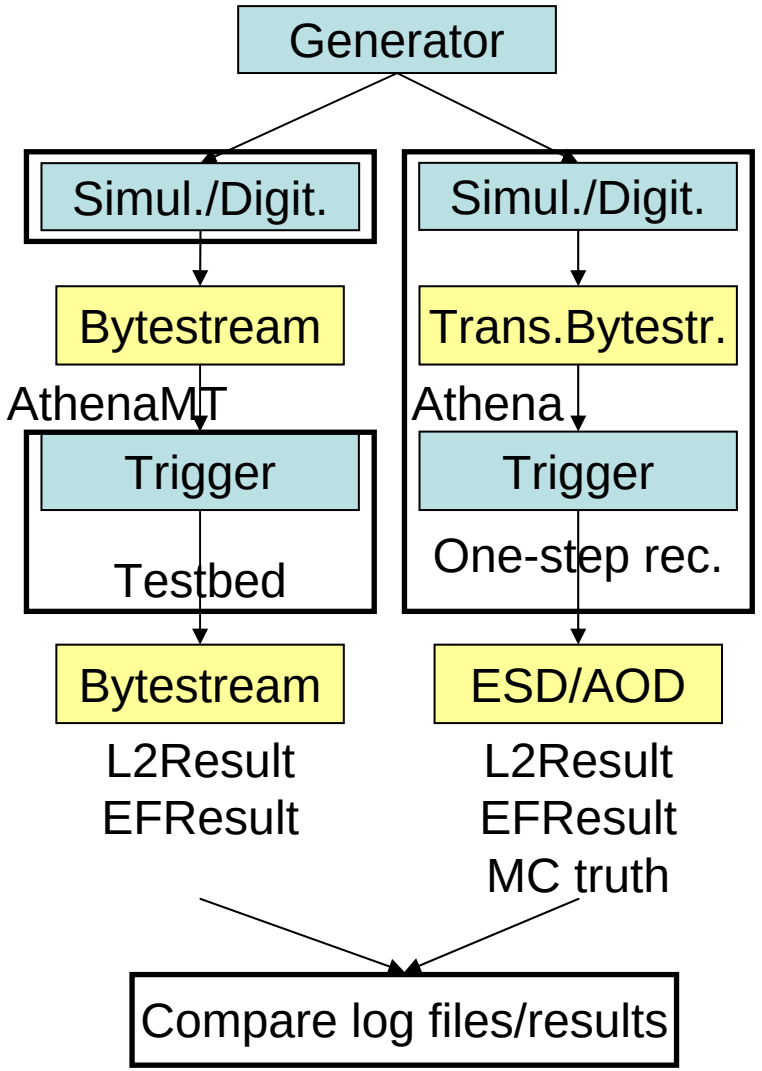
Signature/Menu studies

- Few optimisation studies have been performed and these took a lot of effort;
- These required relevant data to be available offline;
- A large menu, with O(100) signatures x3 levels must be optimised for commissioning;
- Further studies will be needed when background rates are better known;
- Bytestream files from prescaled low-threshold signatures and force-accepted events may be used to generate data for these studies.



Trigger operation debugging

- Statistical: monitoring histograms
  - No need to store any data offline
  - Probably not useful if we're looking for small effects or unusual occurrences
- Event by event:
  - Only way to make HLT event data available is through L2/EFResult ⇒ Serializer
  - Regression-type test, running on bytestream data and comparing to ESD: in both cases, retrieve data from L2/EFResult and run hypothesis algorithms on it
  - MC truth information also available in ESD/AOD: compare reconstruction and truth



What do we have at present?

From: **https://uimon.cern.ch/twiki/bin/view/Atlas/TriggerEDM**

HLT

Class	level	Status	persistency		Documentation
			POOL?	Serializer?	in 11.0.5
L2Result	LVL2	ok	yes	n/a	Doxygen
TriggerElement	LVL2+EF	ok	no	n/a*	Doxygen
RoIDescriptor	LVL2+EF	ok	no	n/a*	Doxygen
LVL1::RecEMTauRoI	LVL2	ok	no	n/a*	Doxygen
LVL1::RecEnergyRoI	LVL2	ok	no	n/a*	Doxyegn
LVL1::RecJetEtRoI	LVL2	ok	no	n/a*	Doxygen
LVL1::RecJetRoI	LVL2	ok	no	n/a*	Doxygen
LVL1::RecMuonRoI	LVL2	ok	no	n/a*	Doxygen
TriggerDecision	LVL1 + LVL2 + EF	in development, planned for 11.0.5/11.4.0	Yes	n/a	Doxygen

Steering objects

Accessible from both BS and POOL

Needs algorithm to fill it

* TriggerElement, RoIDescriptor and the various RecRoIs are included in the L2Result but this is done with special code, not the serializer.

MuonFeature	LVL2	ok	yes	yes	Doxygen	
CombinedMuonFeature	LVL2	from 11.4.0	?	yes	Doxygen	
TrigDiMuon	LVL2	?	?		Doxygen	
TrigCaloCluster	LVL2	used from 11.0.5	yes	no (container)	Doxygen	New Calo EDM from 11.0.5
TrigEMCluster	LVL2	used from 11.0.5	yes	no (array)	Doxygen	
TrigTauCluster	LVL2	used from 11.0.5	yes	no (arrays)	Doxygen	
TrigT2Jet	LVL2	used from 11.0.5	yes	yes	Doxygen	
TrigMissingET	LVL2 & EF	new	probably	probably	Doxygen	Not tested but looks OK

TrigElectron	LVL2	used from 11.0.5	yes	yes	Doxygen	Only electron and tau for now…
TrigTau	LVL2	used from 11.0.5	yes	yes	Doxygen	
CaloCluster	EF	ok	yes	no	Doxygen	From Offline EDM
Rec::TrackParticle	EF	ok	yes	no	Doxygen	
tauObject	EF	ok	yes	no	Doxygen	
egamma	EF	ok	yes	no	Doxygen	
Jet	EF	ok	yes	no	Doxygen	

What's missing?

- **Truth-association classes**
- **Filling TriggerDecision**
- **New functionality in Serializer (STL container serialization)**
- **Having all objects in L2/EFResult (EF "persistent" EDM? Review started.)**
- **Some steering functionality: "accept all" mode**
- **What else?**

At present:

- Recent tests in 11.0.5 (mostly Monika)
- Running slices at level 2:
  - Muons
  - e/gamma
  - Taus
  - Jet
- Running slices at EF:
  - Just e/gamma for now
- In the ESD/AOD (list from M.W.):
  - L1EtmissObject
  - L1JetObjectContainer
  - MuonFeature
  - TrigT2JetContainer
  - TrigTauClusterContainer
- See e/gamma analysis Wiki for example analysis job:

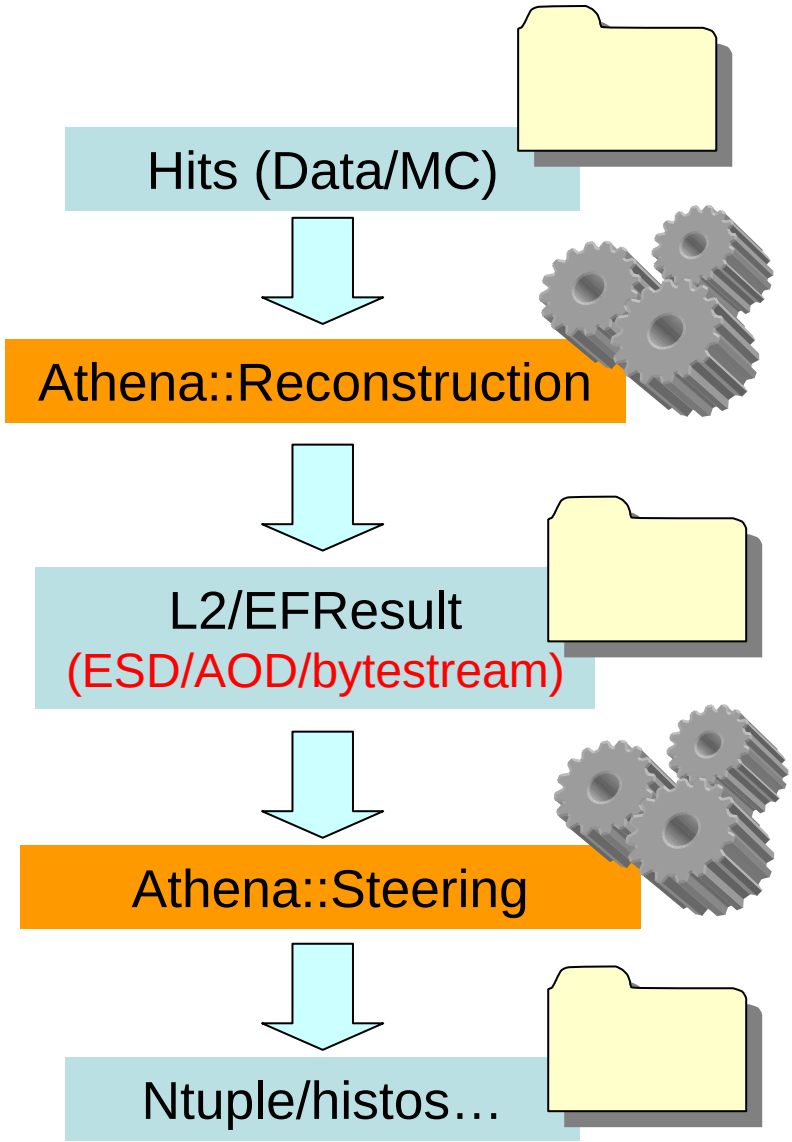
Most of these are in an incomplete state, e.g. data objects exist but no hypothesis are running

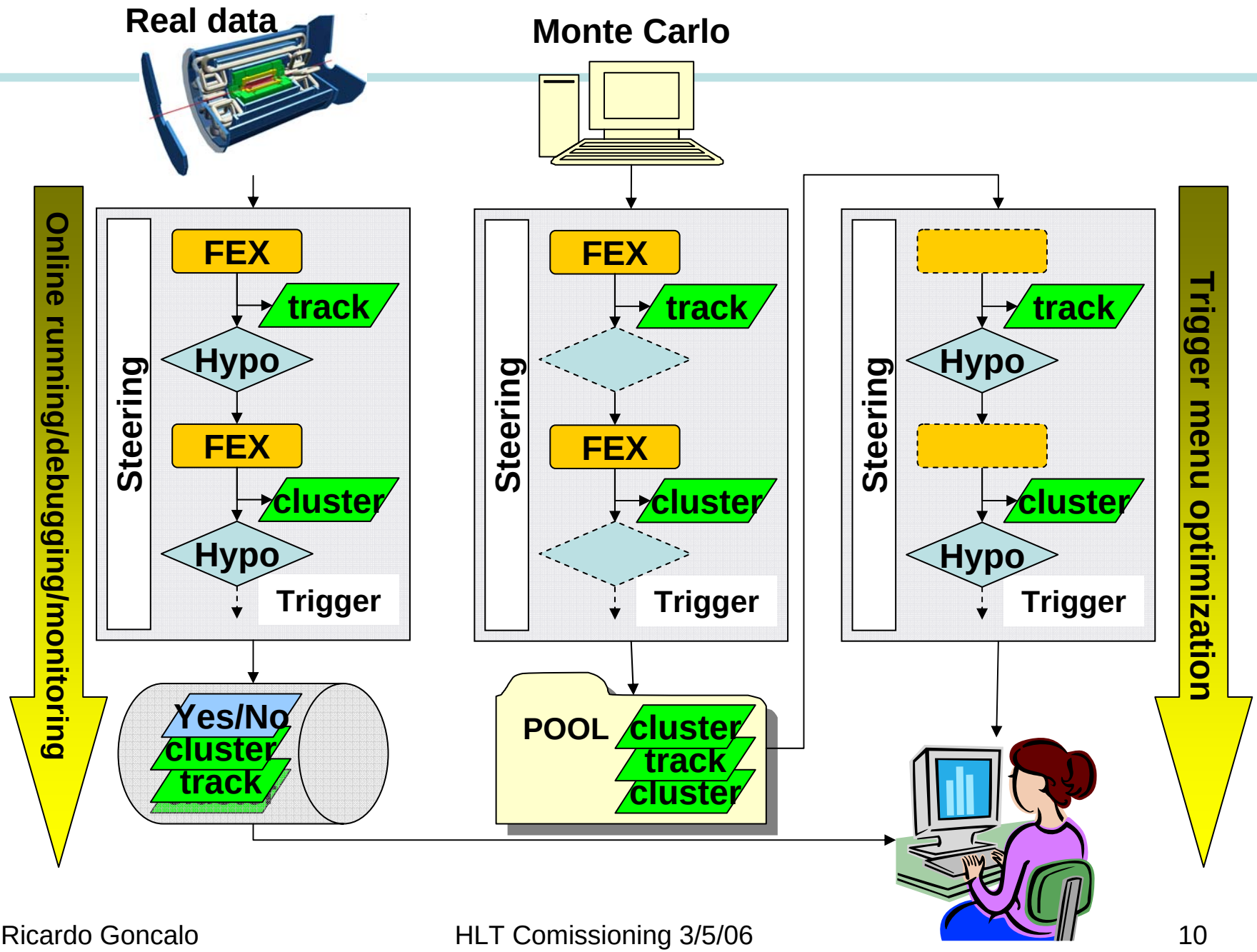
Other tests (of the e/gamma slice) show that TrigInDetTracks, TrigEMClusters, TrigElectrons are correctly stored in ESD/AOD, more testing and debugging to be done

https://uimon.cern.ch/twiki/bin/view/Atlas/EgammaTriggerAnalysis

How to analyse ESD/AOD/BS

- To produce ESD/AOD/BS:
  1. The Steering serializes all relevant data objects into L2/EFResult (not at present)
  2. TriggerDecision object filled after Steering has run and put in ESD/AOD
  3. L2Result and EFResult attached to event (BS) or stored in ESD/AOD
- To analyse ESD/AOD/BS:
  1. The Steering de-serializes all data objects
  2. Then either look at reconstructed features and TriggerDecision or
  3. Run hypothesis algorithms only on reconstructed features
- **Note**:
  - In this way, the hypothesis algos can be run many times over the same data objects as if running online, and the cuts **optimized**
  - The cuts can only be tightened wrt the original cuts




Real data
Monte Carlo
Online running/debugging/monitoring
Trigger menu optimization
Steering
FEX
track
Hypo
FEX
cluster
Hypo
Trigger
Steering
FEX
track
FEX
cluster
Trigger
Steering
track
Hypo
cluster
Hypo
Trigger
Yes/No
cluster
track
POOL
cluster
track
cluster

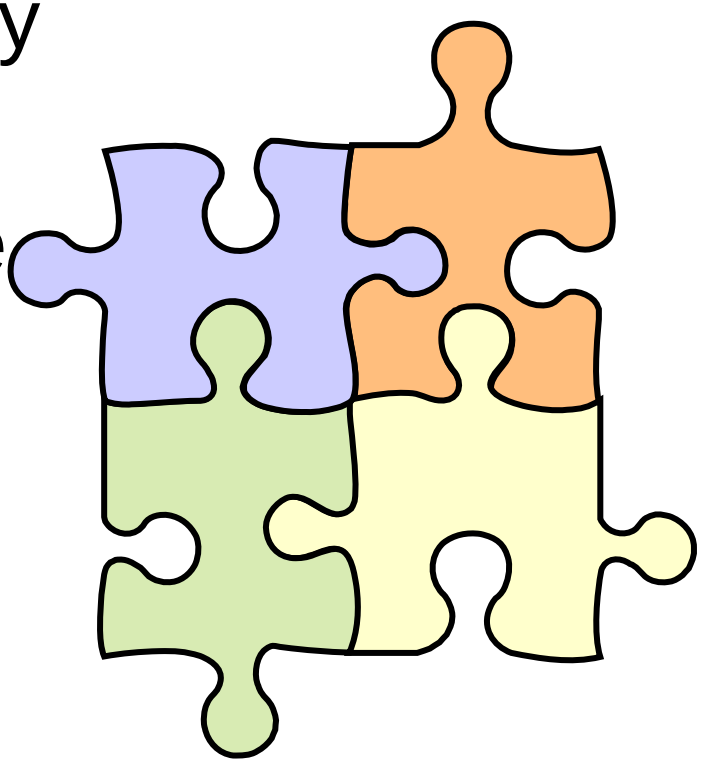
Plans for release 12 (slide from Calo SW workshop in Feb 06)

- **Wish list** (e/γ specific..):
  - New calo EDM (**`TrigEMCluster`**,..): serializable, tested - DONE
  - **`TrigInDetTracks`**: serializable, tested - DONE
  - **`TrigInDetTrack`** association: (**`TrigInDetTrackTruth`** and **`TrigInDetTrackTruthMap`**) storable in POOL and tested – not yet
  - TrigParticle (**`TrigElectron/TrigTau`**): hypothesis algorithms to fill them; example menu configurations - DONE, others to follow
  - **`TriggerDecision`**: configured with a default menu; filled with Steering information and stored in AOD/ESD for users - getting there
  - Test jobs for each of the above - …hmmm not quite there yet
  - Default doTrigger=True in CSC production

- This will allow many studies that were hard to do/impossible up to now
- It would also generate lots of feedback from the physics community
- We can finally start to build a realistic menu combining Electrons, Taus, Muons, Jets, xE_T

Conclusions

- A lot of progress recently on trigger EDM and persistency
- Many different pieces in the puzzle are coming together
- Clear need for HLT data to be available for: debugging and menu study and optimization

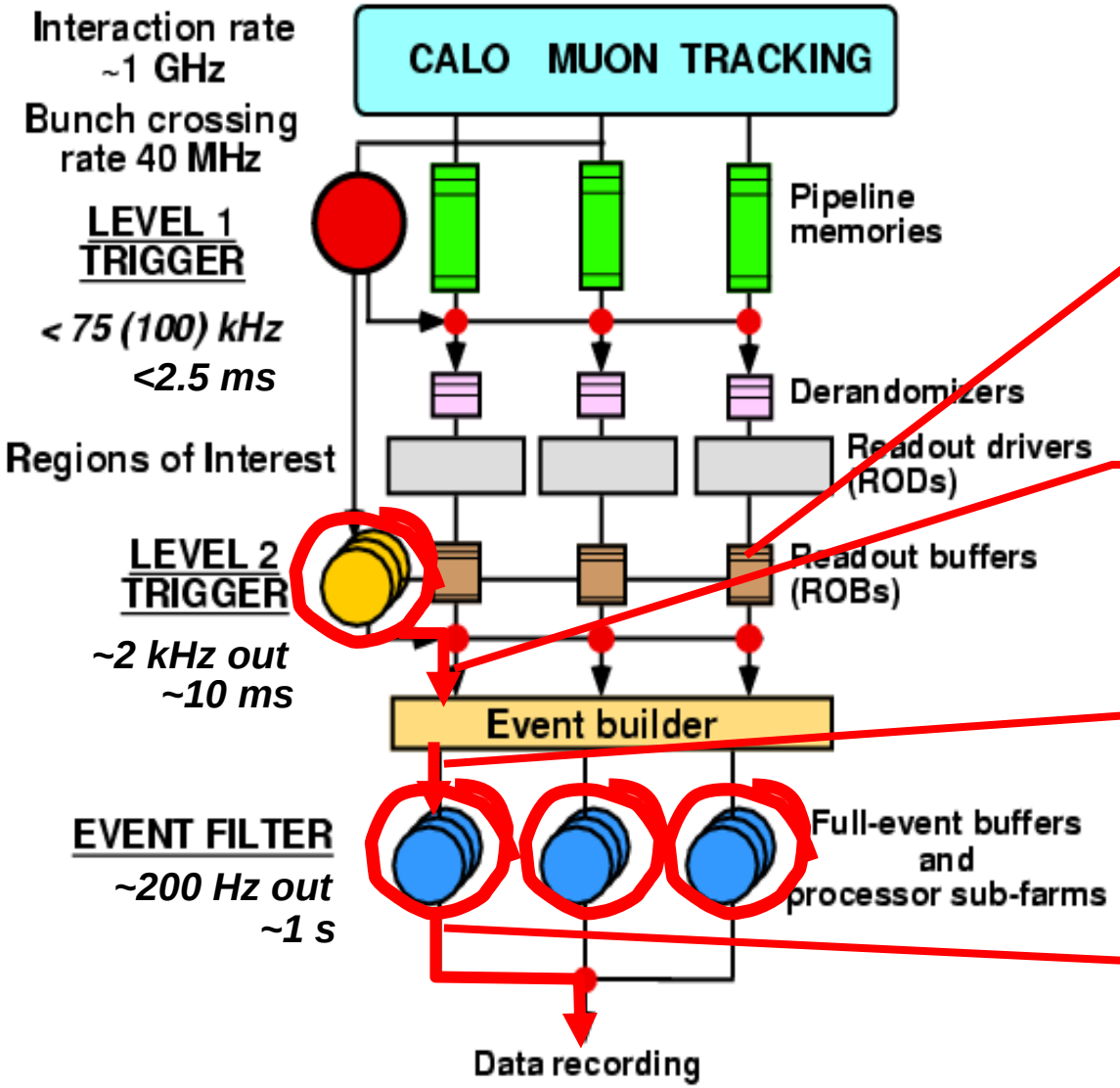


Backup slides

What do we need in principle?

- Trigger decision: yes/no for each signature/level
  - What's the signature efficiency at level 2? How much overlap is there with other signatures? What other signatures/levels were satisfied?
- Configuration information (how easy is it to retrieve?)
  - What was the prescale factor for this signature in 2008?
- Navigation information: which track/muon/jet comes from which RoI
  - Is it always a em/jet/tau/mu RoI giving this problem?
- Reconstructed objects used in decision: vertices/clusters/electrons
  - How well does the extrapolated track match the cluster?
- Truth association information: Track-GenParticle,…
  - Why didn't this electron leave a reconstructed track?

Sources of event data from the trigger



TDAQ data flow

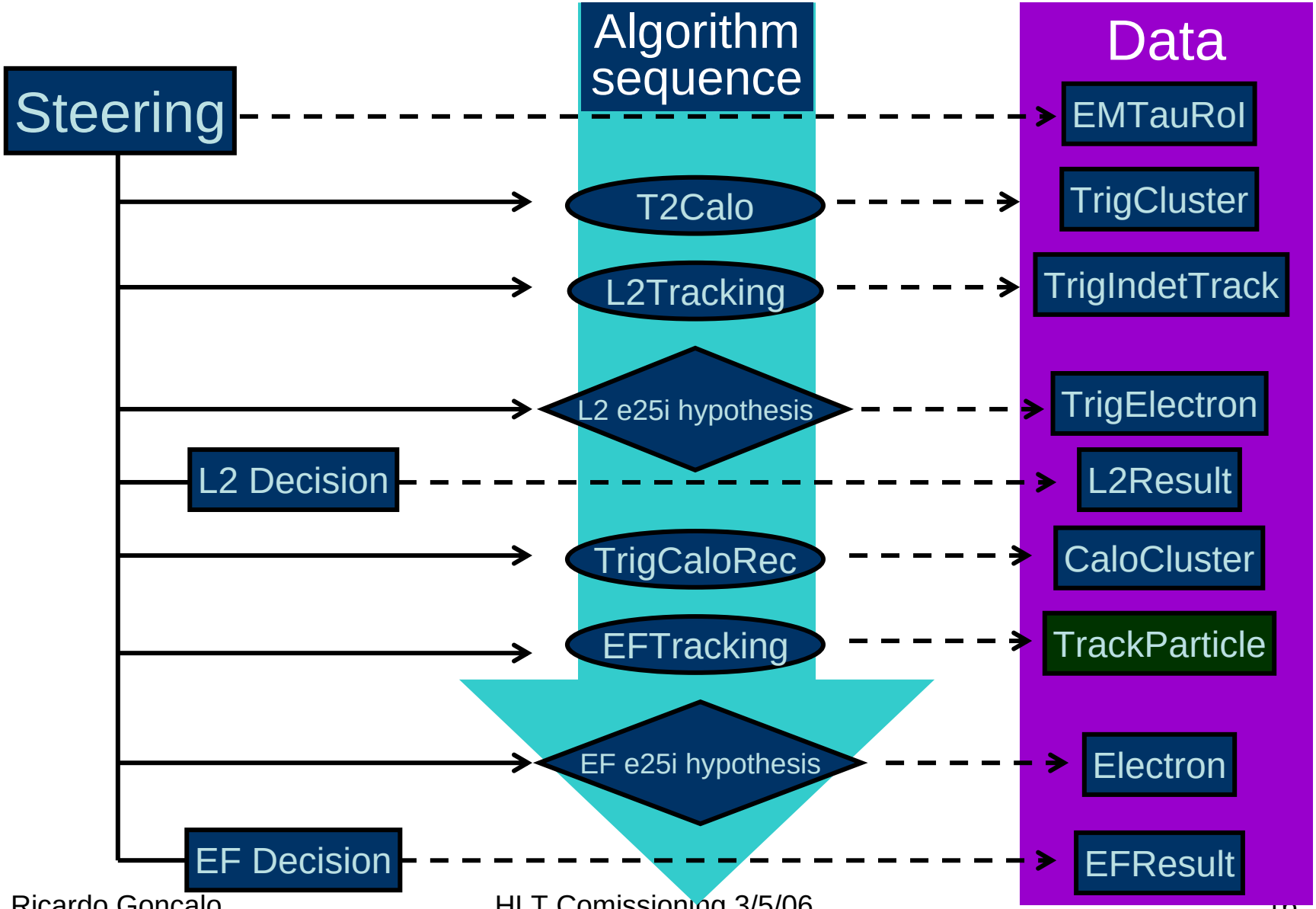
Detector data - raw event fragments

Level 2 data appended to raw event

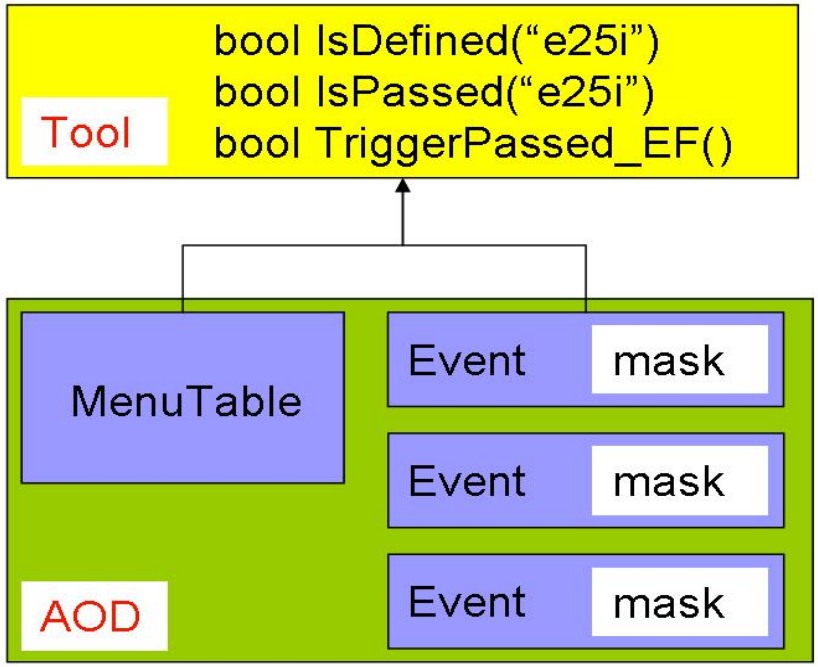
Event passed to EF, uses LVL2 data to seed

EF data appended to raw event

Data produced by the HLT – e/γ example



Trigger Decision: yes/no result



- Signatures passed/failed/prescaled encoded in a **bit pattern** stored once per event
- Bit pattern **interpreted** through MenuTable (in conditions DB or in RunStore…)

- A Tool would provide the user interface to L1/L2/EF and individual signature results by interpreting bit patterns in AOD. It would give:
  - Decision bit for each signature
  - Access to trigger configuration through methods like isDefined()

TriggerDecision: status

- **Short term implementation**: while there are only a few signatures
  - Store object in AOD consisting basically of map:

    ```
    map<string label, bool accept>
    ```

  - Derive trigger decisions from Hypothesis algorithms
  - Only a few signatures - wasted AOD space by repeating labels each event is negligible

- **To do:**
- TriggerDecision has been implemented (Monika)
- Need algorithm to fill TriggerDecision for a given menu (don't need /can't have machinery to deal with generic menus for now)

Status of the Serializer

- Simon, Gianluca, Jiri
- Uses Reflection library to serialize classes into `L2Result` and `EFResult` in bytestream or POOL
- Works with:
  - Simple native types (**int**, **double**, **float**)
  - **Pointers** (and **NULL** pointers)
  - Follows (non-NULL) pointers
  - Classes need to have **SEAL** dictionaries (same requirement for POOL)
  - All unsupported class data members should be declared **`transient`** (in `selection.xml`)
- To do:
  - Store references to POOL objects (e.g. to write **`EFResult,`** where objects are **not** serializable) will be there from 11.0.5
  - STL containers (std::vector<>, DataVector<>)
  - Schema evolution
  - For 11.3.0 (in ~2 weeks) switch from Reflection to Reflex

Level 2: TrigParticle

- To store the candidate object that was accepted by a signature
- Should be light, with no ElementLinks or heavy inheritance, to ease persistency
- Example:
  - **`TrigElectron`**
  - Summary data to use for **debugging** and **analysis**
  - **`TrigElectron`** data members:
    - Roi_Id
    - eta, phi
    - Z vertex
    - p_T, E_T
    - **pointer** to track
    - **pointer** to cluster

    (“best estimate” values from HLTHypoAlg)

```
class TrigElectron {
 public:
    TrigElectron();
...
    TrigTrack*   track(int i);
    TrigCluster* cluster();
    int RoI();
 private:
    int              m_roi;
    double           m_eta;
    double           m_phi;
    double           m_z_vtx;
    double           m_p_T;
    EMShowerMin*     m_cluster;
    vector<TrigTrack*> m_trk;
};
```

Other loose subjects

- Level 1:
  - Schema evolution: hasn't been a concern so far because trigger not run by default in productions: this should change soon
  - Thresholds passed/trigger configuration
- Level 2:
  - TrigInDetTrack truth association: under development
  - Until Serializer can deal with STL containers, using a few tricks that should be removed
- Event Filter:
  - Several classes being adapted from offline (Iwona)
    - TrackParticle-truth association
    - VxVertex
- The size of all this:
  - LVL2 objects very small compared to offline
  - EF objects **are** offline (and we're only reconstructing a fraction of the event)

- Dataset transformation(*)
- 004100.T1_McAtNLO_top muon
- egjet
- 004022.Electron_Pt_25 egjt
- 004024.Photon_Pt_60 egjt
- 004042.mu_pt10 muon
- 007600.SingleTaupt20 egjt
- 003035.J2_Pt_35_70 egjt
- 003036.J3_Pt_70_140 egjt
- 003038.J5_Pt_280_560 egjt
- 003039.J6_Pt_560_1120 egjt

- 2nd 100k Requests
- ==========
- Datasettransformation(*)
- 004201.ZeeJimmy egjt
- 004202.ZmumuJimmy muon
- 004203.WenuJimmy egjt
- 004205.pythia_W_tau_tauola egjt
- 004807.A3_Ztautau_tightfilter egjt
- 004934.mu_pt100 muon
- 005300.PythiaH130zz4l egjt
- muon
- 005310.PythiaH120gamgam egjt
- 017700.Bs_Jpsi_mu6mu3_phi_KplusKminus muon

- 3rd 100k Requests
- ==========
- Dataset transformation(*)
- csc11.005800.JF17_pythia_loosejet_filter.digit.v11004201 egjt

- 4th 100k Requests
- ==========
- Dataset transformation(*)
- csc11.005802.JF17_pythia_jet_filter.digit.v11004201 egjt