Trigger ESD/AOD

Simon George (RHUL)
Ricardo Goncalo (RHUL)
Monika Wielers (RAL)
Reporting on the work of many people.

ATLAS software week 26-30 September 2005
CERN

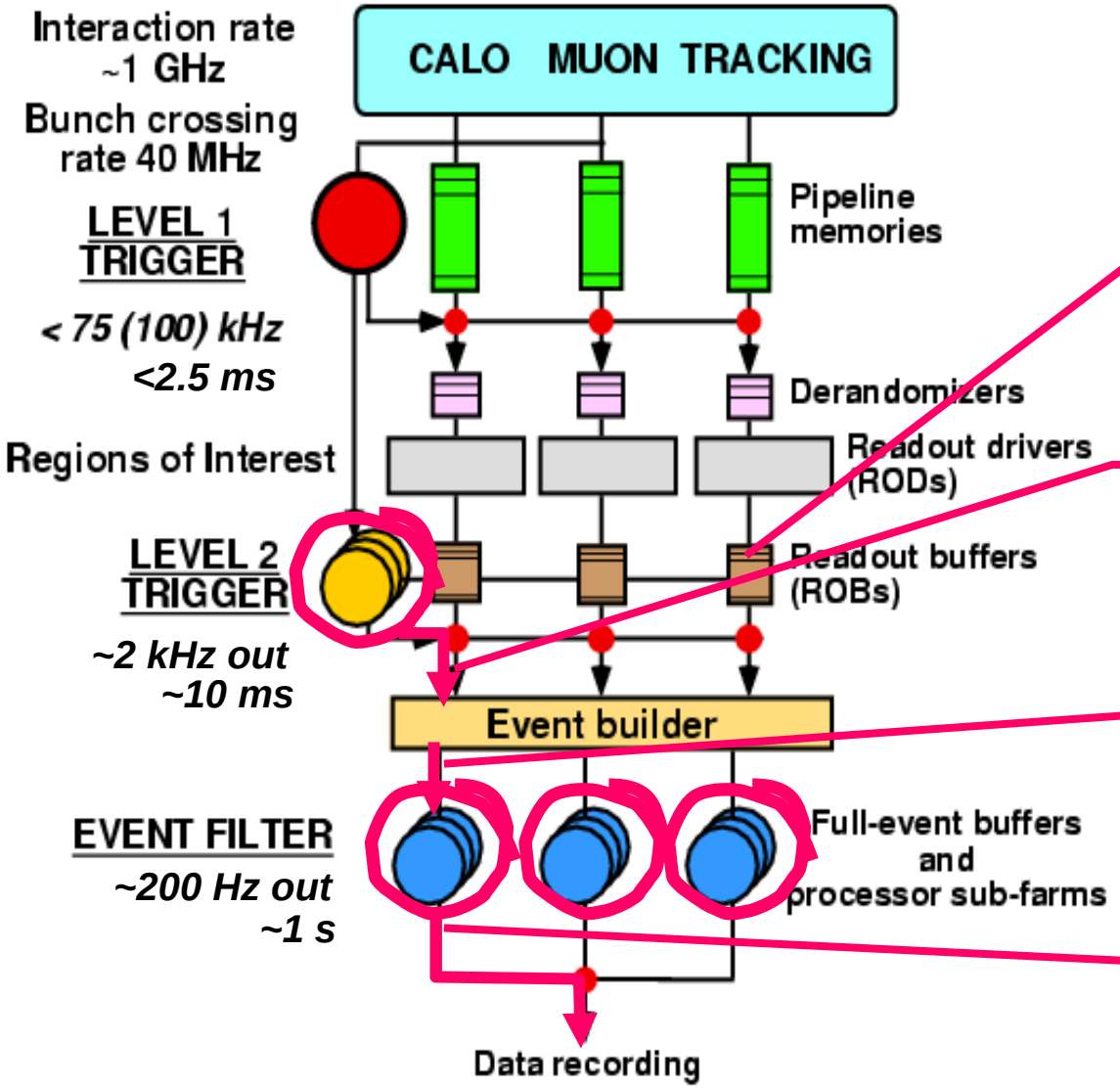
Contents

- Event data from the trigger
  - Data produced by the HLT
  - Use cases
  - Online constraints
- Inventory of ESD & AOD classes
  - LVL1, LVL2 & EF
  - Current (Rome data)
  - Planned classes
- Conclusions
- Practical example: see next talk

Introduction

- The trigger is a source of event data
- Real data taking
  - A small amount of trigger information is written out along with the detector data
  - Can be propagated to ESD/AOD for easy reference
- Reconstruction of simulated data
  - LVL1 simulation produces data
  - HLT: same software is run as online, but more data is available

Sources of event data from the trigger



TDAQ data flow

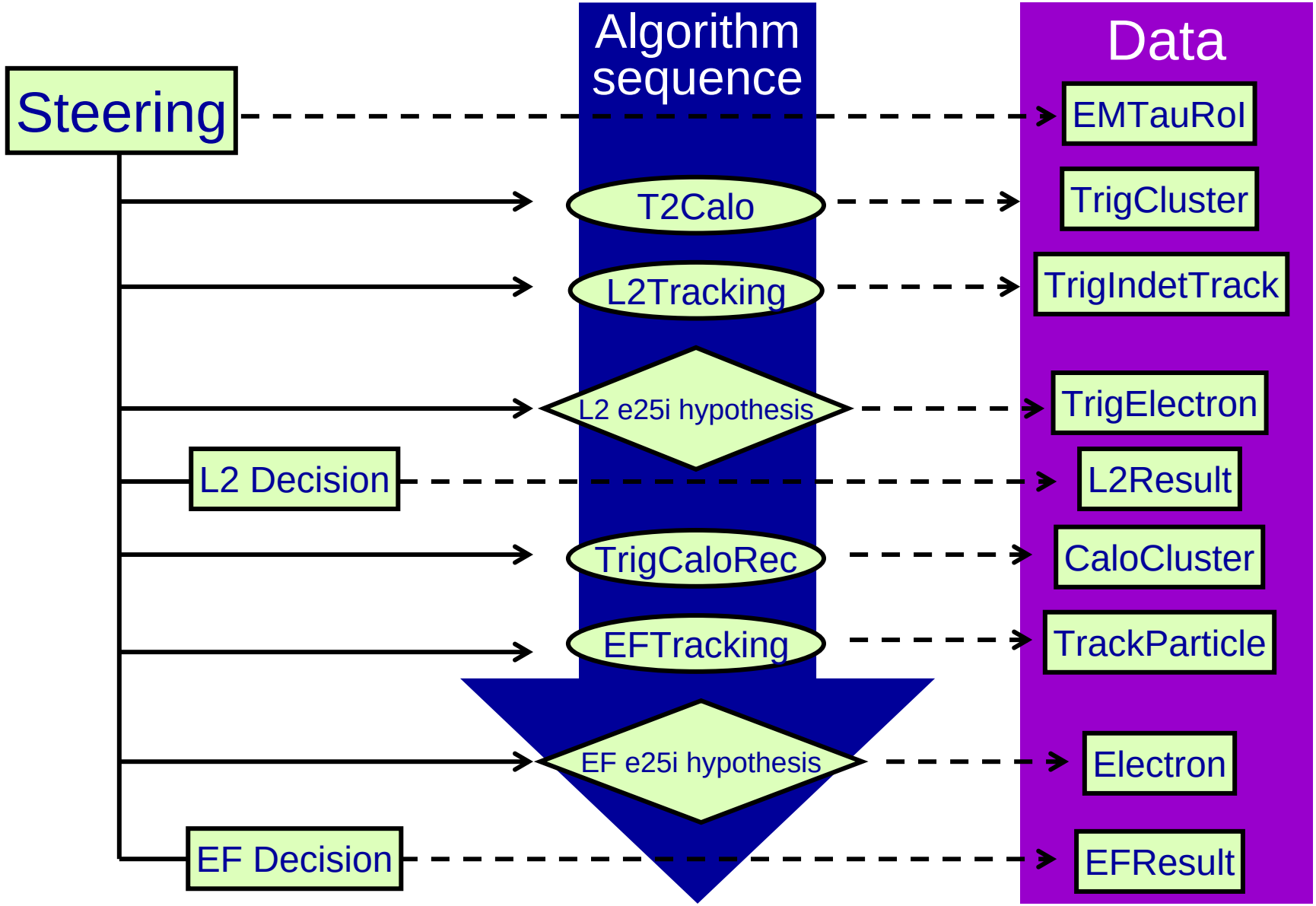
Detector data - raw event fragments

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



Use cases for HLT persistency

- The **same code is run online & offline**, so would like to write the same objects in both cases, but with different “persistency technology”, byte stream and POOL respectively.

- Online (writing to byte stream)
  - Basic L2 Result and RoI seeding information written out, then used by EF
  - Extended L2/EF Result written out, for monitoring, debugging, calibration
  - L2 & EF Results written out for use offline

- Offline (writing to ESD/AOD)
  - Physics analysis: get trigger decision for an event from AOD (sim or real data)
  - **Trigger performance tuning or more detailed physics analysis: re-run HLT hypotheses & decision on AOD.**
  - Detailed trigger performance studies: re-run algorithms on AOD.

- Online constraints
  - Dataflow imposes size and bandwidth constraints
    - Minimum necessary data for use case
  - Classes must be simple enough to serialize

Trigger performance tuning or more detailed physics analysis

- Arguably the m
- How do we see

For existing Rome data you have to redo your own private production from ESD

- Production runs HLT software as part of reconstruction:
  - Same algorithms and selection as online
  - Steering, reco algorithms, hypothesis algorithms, decision
- Production writes to AOD enough information re-run HLT hypotheses & decision
  - Need to write any data used by hypotheses
  - E.g. CaloCluster, TrackParticle
- User job runs on this AOD
- Joboptions include to run the same HLT steering + hypothesis algorithms as in the production
  - Need same steering configuration (sequences and menus)
  - Reco algorithms turned off
    - The data they would have produced is provided from AOD instead
  - Hypothesis algorithms are re-run, so cuts can be changed
- This way one can tune the HLT and study performance

Online constraints

- Size
  - Average L2Result size within 2kB/event
  - Max L2Result 64kB/event
  - Classes must be designed to convey minimum necessary data for use case.
    - E.g. use float rather than double if sufficient, bit-pack bools.
- Format
  - Objects are written out in raw event format (byte stream), not ROOT
  - LVL2 and EF may **append** a small amount of data to the **raw event**
  - It could also be used (within constraints) to extract information for debugging, monitoring and calibration
  - Simple, generic serializer turns objects into vector<int>
- Class design
  - Serializer constrains class design
    - E.g. only support float, double, int, pointer
    - Do not intend to support full offline EDM e.g. ElementLinks
    - Complex inheritance structures would cause problems
  - Well suited to L2 EDM but not much hope for offline EDM used by EF

LVL1 classes in AOD/ESD (already in Rome)

- RoI classes (in ESD and AOD)
  - EMTau_ROI, JetET_ROI, JETET_ROI, EnergySum_ROI, MUON_ROI
  - 'Hardware like'
  - Contain bit pattern of which threshold passed, eta/phi
- Reconstruction objects (only in ESD: also in AOD from rel.11.0.0)
  - L1EMTauObject, L1EMJetObject, L1ETmissObject
  - 'Software like'
  - Contain energies, isolation, eta/phi quantities which are used for optimisations
- CTPDecision (in ESD and AOD)
  - 'Hardware like'
  - Contains word with trigger decisions
  - Contains just few words
- Future plans:
  - MUON_ROI fine, no further plans
  - Need single consolidated Calo class, which contains cluster/isolation sums + thresholds passed
  - Final CTP hardware not yet decided upon (should be very soon), thus might need revision

LVL1 classes only in ESD (New from rel.11)

- TriggerTower (available in release 11)
  - 'Hardware like'
  - contains for towers above threshold (in total ~7200 tower, typically only 100-200 after zero-suppression)
    - EM and had energies (final calibrated 8-bit ET values) per tower
    - Raw energy, digits, filter output per tower
    - eta/phi
  - Currently contains more data than actually read-out
    - Hardware will only give digits and final energies + eta/phi
  - Future plans:
    - further re-writing/re-design:
    - separate raw tower for internal use + stripped down calibrated tower for persistency.

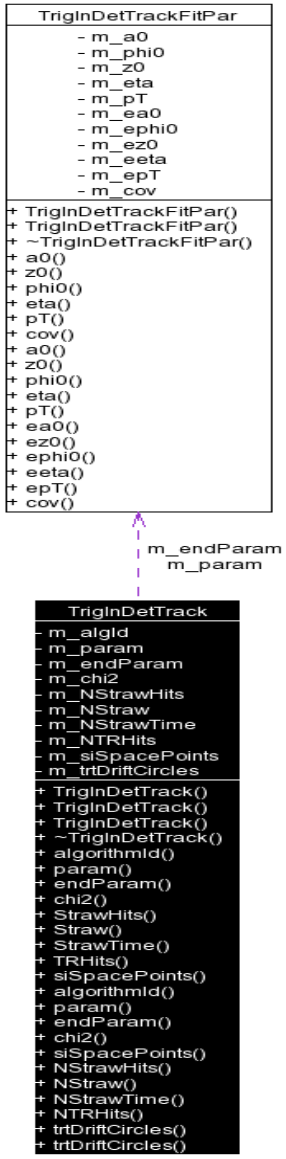
- JetElement (same as TriggerTower – release 11)
  - 'Hardware like'
  - Similar to TT but coarser granularity for jet trigger (~1k tower in total, again zero-suppressed)
  - Contains EM/had energy, eta/phi

What was available for LVL2/EF in Rome data

- EMShowerMinimal (ESD only):
  - Output from T2Calo
  - Contains relevant shower shapes and pointer to CaloCluster (also stored in ESD)
- TrackParticle (ESD and AOD…by accident)
  - Output from several LVL2 tracking algorithms
  - Obtained by conversion from TrigInDetTracks
  - Contains a few doubles, an ElementLink to Trk::Track, and pointers to RecVertex, MeasuredPerigee, TrackSummary, FitQuality…
- No L2Result!
- CaloCluster (only in ESD)
  - Produced by TrigCaloRec
- No other Event Filter reconstruction, so objects produced by offline reconstruction used instead (in ESD, AOD)
- No EFResult!

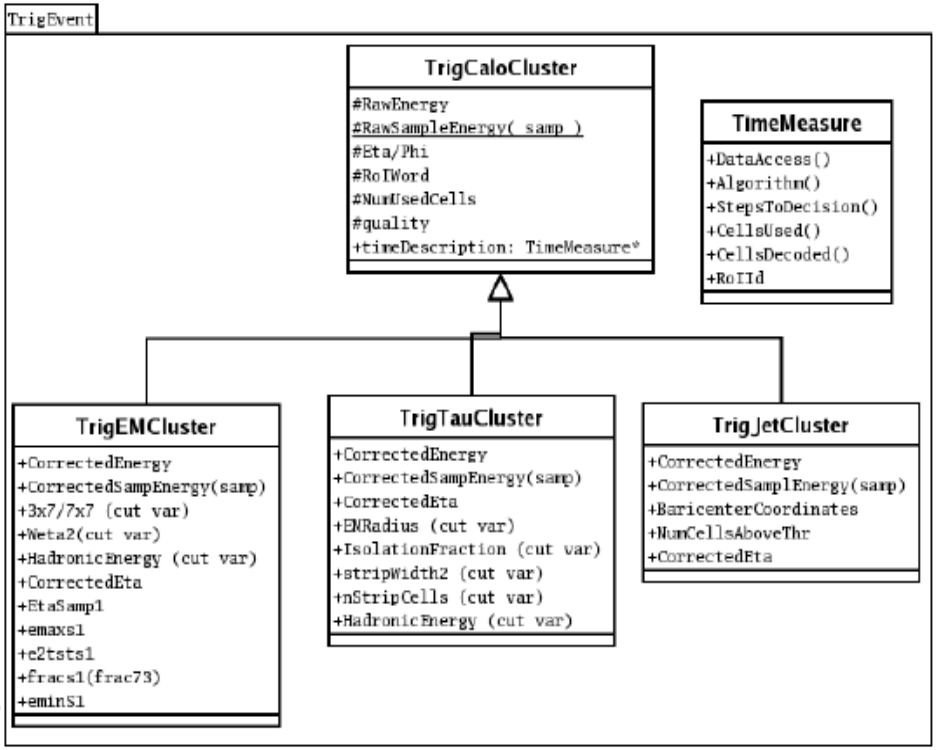
- Beware: EventInfo contains a class TriggerInfo but it is not filled
  - We will think about how best to use this and perhaps change it

Planned LVL2 classes in AOD/ESD – reconstructed objects



- TrigInDetTrack
  - Inner detector track quantities
  - 21 doubles and 5 int per track
  - Plan to optionally include space points for special trigger studies
- MuonFeature
  - Should be in rel. 11
  - Muon track quantities
  - 1 int, 7 float

- Calorimeter classes



- Discussed last LAr week
- Classes to be implemented very soon after rel. 11
  - Use in algorithms will come later than that

Planned LVL2 classes in AOD/ESD – particle objects

- 'Trig*Particle*' classes
  - Minimal summary data to use for seeding and analysis
  - Output from hypothesis
  - TrigElectron, TrigMuon, TrigTau, TrigJet…
- Example: TrigElectron
  - data members:
    - Roi_Id
    - eta, phi
    - Z vertex
    - p_T, E_T
    - pointer to track
    - Pointer to cluster
  - Variables filled by hypothesis algo with "best values"
  - Pointers can be 0 when track & cluster not needed
  - Aim to have prototype in release 11

- Other classes will be added as required
  - Reflect hypothesis algorithms in the trigger
  - E.g. J/Psi, Z, di-muon

Planned EF classes

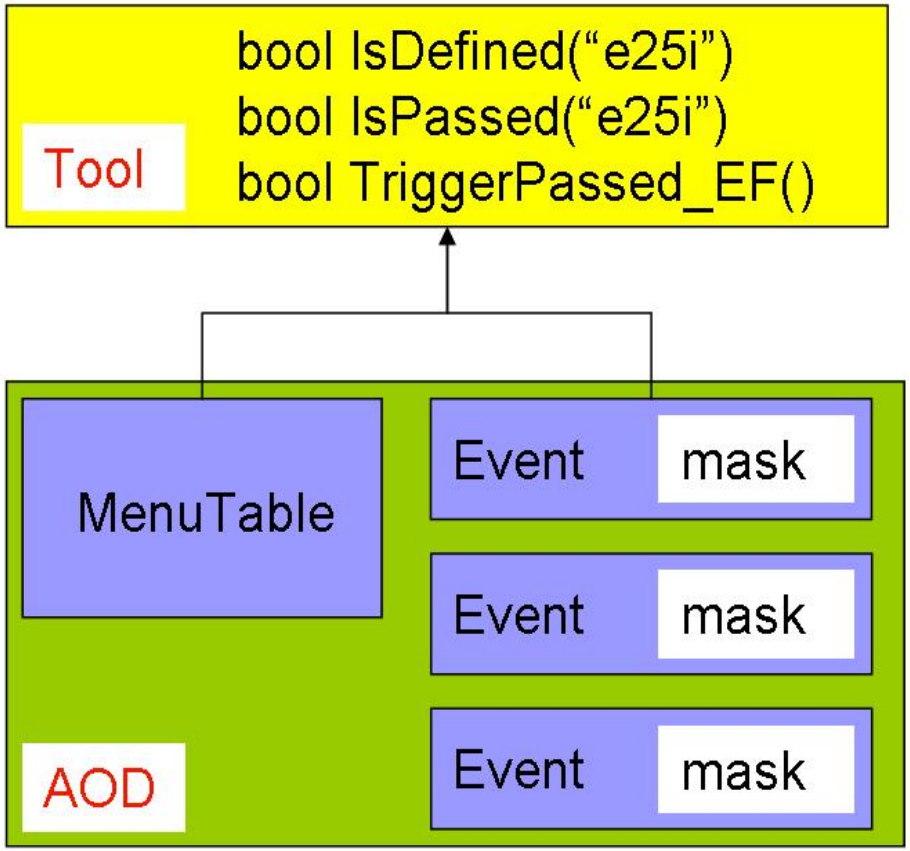
- EF reconstruction based on offline software
  - Re-use algorithms & tools
  - Typically seeded and simpler options
- So most event data are familiar offline classes
  - Have to save in ESD/AOD **in addition** to full offline reconstruction
- Examples
  - E/gamma: CaloCluster and TrackParticle in AOD
  - TrigMoore: Trk::Track (ESD), TrackParticle (ESD AOD)
    - Plans to provide ESD CombinedMuon and AOD Muon

Data from steering

- LVL2 and EF Result
  - **Decision**: bit for each signature
    - Which signature corresponds to each bit is defined by configuration (MenuTable)
  - Prescale masks or counters
  - Internal information from the steering
    - Association of reconstructed objects like tracks and clusters to their corresponding RoIs
    - ‘State’ of the RoIs and signatures being processed
    - These are needed to re-run a hypothesis.
- Availability
  - Nothing in Rome data
  - Planned for inclusion in ESD/AOD in release 11

Trigger Decision

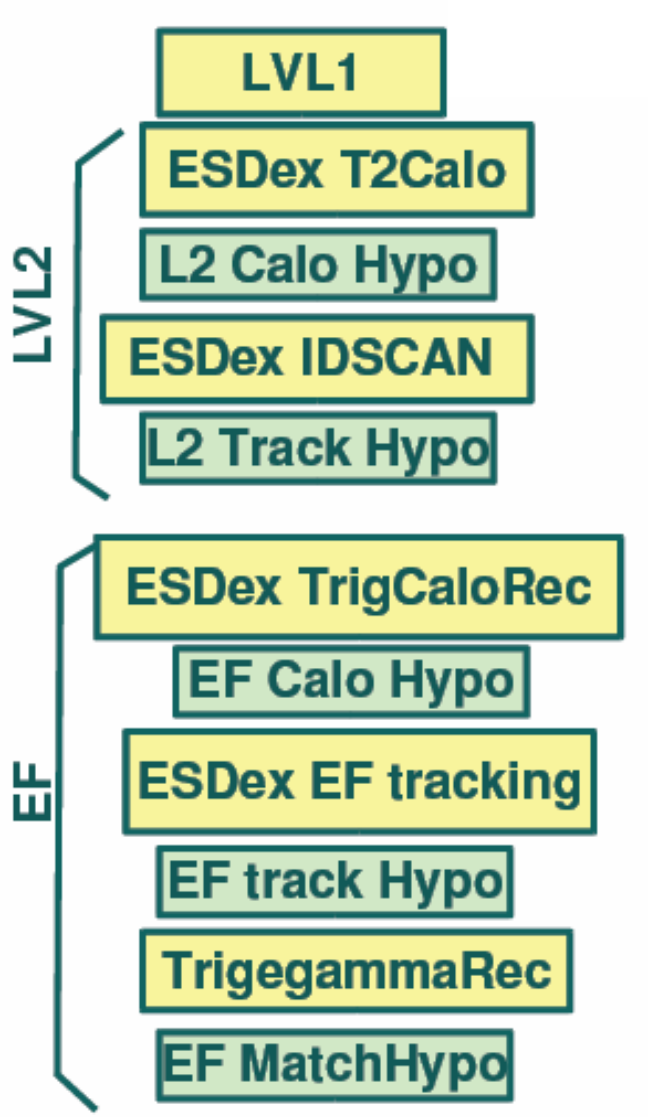
- Plan to provide a Tool to give L1, L2, EF results and signature results from AOD
  - by interpreting “decision” bit patterns
- MenuTable needed to interpret bit pattern
  - Currently no access to trigger configuration conditions data in AOD
  - Proposals in PAT talk
- We can provide a demo version now
  - Special implementation until conditions data issue addressed
  - Ok while there are only one or two signatures
- To use it:
  - Aim to have prototype in release 11
  - Derive trigger decisions from ESD
  - Write to AOD, from which they can be interpreted via this tool



How to use trigger decision today on Rome data

- The software described on the previous slides is not yet available. What is possible **today**?
- We currently offer 3 options to access the trigger decision when analysing Rome data:
  - Analyse ESDs
  - Create customised AODs
  - Use CBNTs and AODs
- For more information on each of the methods, look at the wiki
  - https://uimon.cern.ch/twiki/bin/view/Atlas/PesaEgamma

Analyse physics data on ESDs



- Run the HLT steering
  - Feature extraction algorithms (those that reconstruct objects, like tracks or clusters) are substituted by other algorithms that just read those objects from AOD
  - Run the real hypothesis algorithms so you can change cuts
  - Since there is no actual reconstruction, running is very fast and the job options are rather simple.
- Note: only the recolum01 Rome data contains the trigger information
- Instructions based on 10.0.1 can be found in
  - https://uimon.cern.ch/twiki/bin/view/Atlas/TrigChainOnESDs
    - How to run the e/γ slice
    - How to derive trigger efficiency
    - Solutions to known problems
- Recipe will be updated once release 11 is out

Analyse data using AODs

- 2nd method: create customised AODs
  - Make your own AOD from ESD
  - Add trigger classes in addition to ‘default’ classes into your AOD
  - Then one can run the trigger chain in the same way on the AODs as just described for the ESDs

- 3rd method: Bricolage
  - **Not recommended** - but we thought we should admit what we had to resort to, to get some of our results
  - Run the Root e/γ analysis program on CBNTs
  - Write out the event numbers that pass the trigger to a text file.
  - Read back text file during AOD analysis to get trigger decision
  - See Wiki page for details and limitations

Conclusions

- Trigger EDM exists
  - Fairly limited content in Rome data
  - Much more has been written since then
  - Try to get as much as possible into 11
    - Whilst acting as responsible tag approvers
  - Will be at least release 12 before it could be comprehensively in place
- Working model exists for trigger-aware analysis
  - How to re-run and tune hypothesis algorithms on AOD
  - Demonstrated for e/gamma
  - Will be made easier, better and will cover more triggers
- Overall trigger “decision” (pass/fail) will take time to develop
  - Not available now
  - Plan to make available soon for selected e/gamma signatures
  - Needs work at the next level down to develop the “hypotheses” for other trigger types
- Encourage physics studies to engage with the trigger at hypothesis level