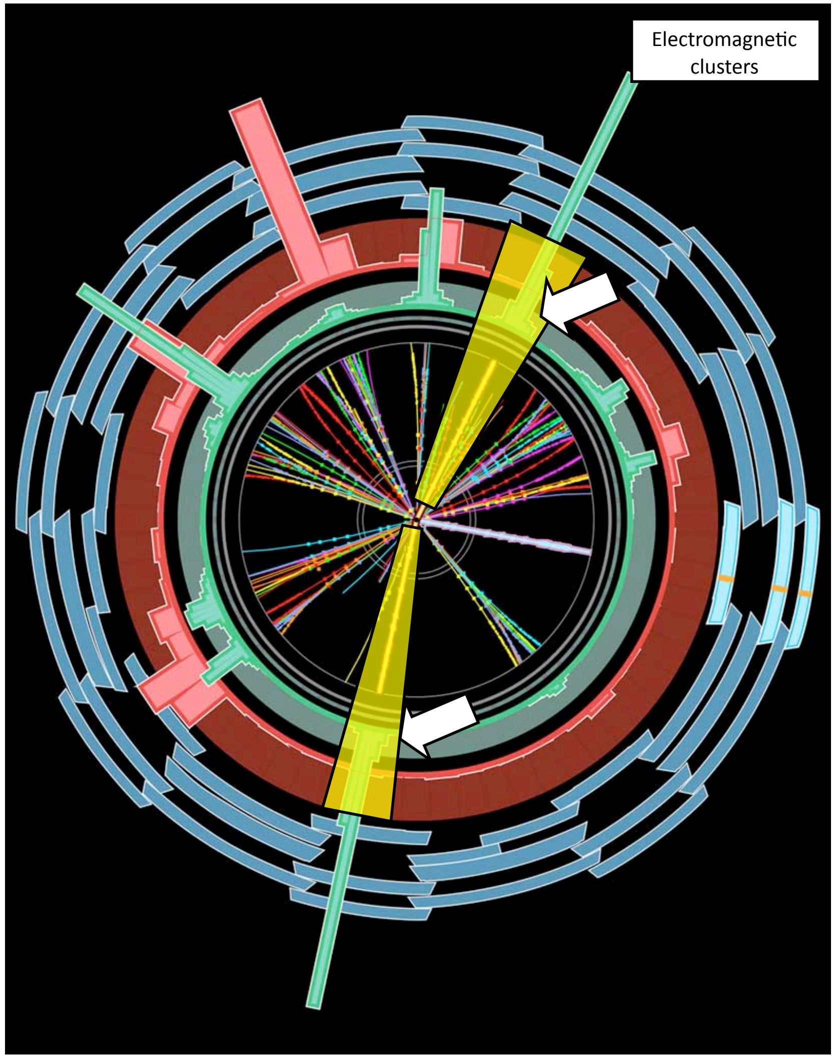
What's in the ATLAS data : Trigger Decision

ATLAS Offline Software Tutorial
CERN, 20-22 August 2008
Ricardo Gonçalo - RHUL

Royal Holloway University of London

Trigger Selection

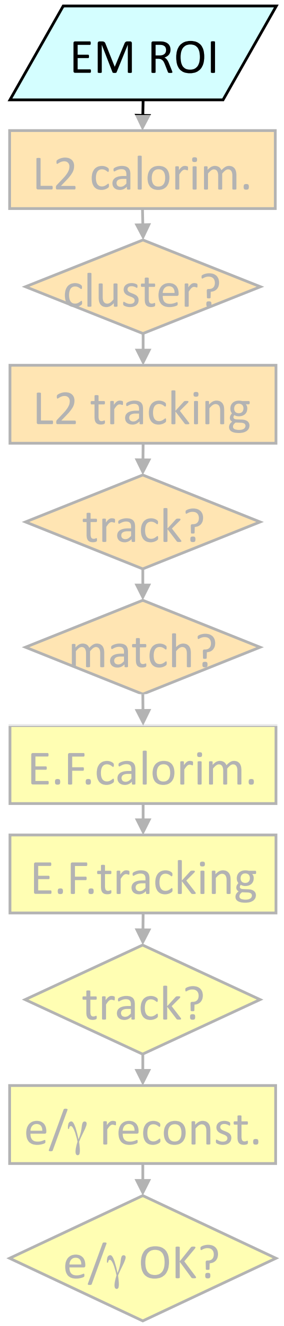
Event rejection possible at each step



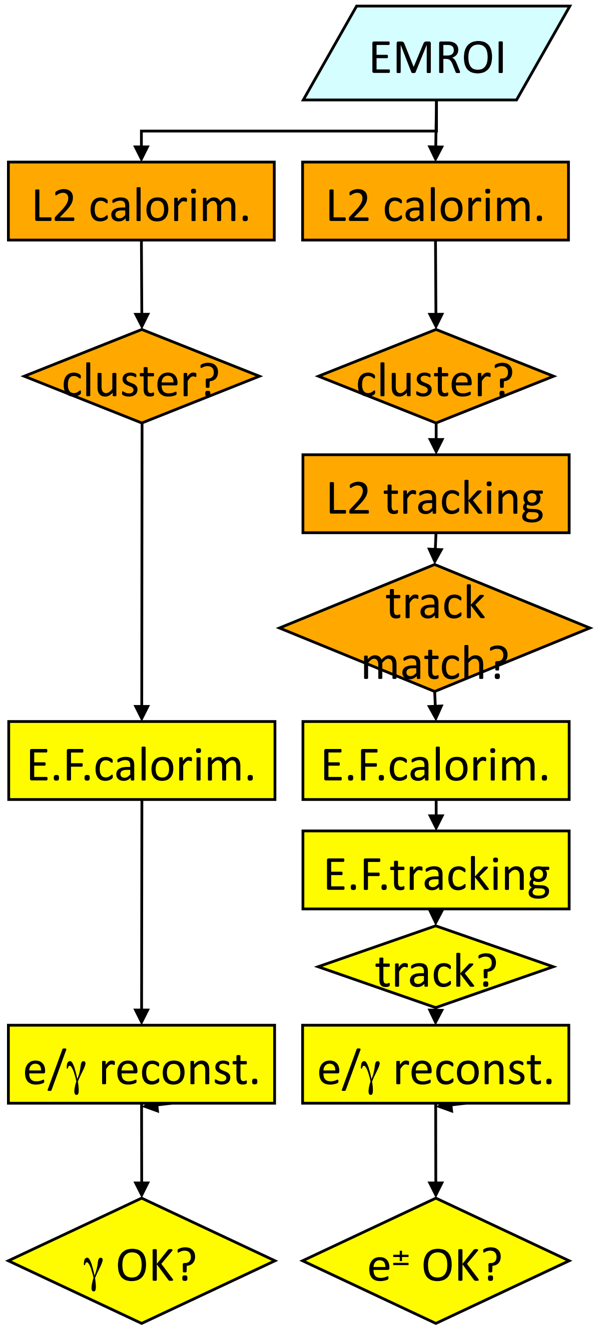
Level1 Region of Interest is found and position in EM calorimeter is passed to Level 2

Level 2 seeded by Level 1
Fast reconstruction algorithms
Reconstruction within RoI

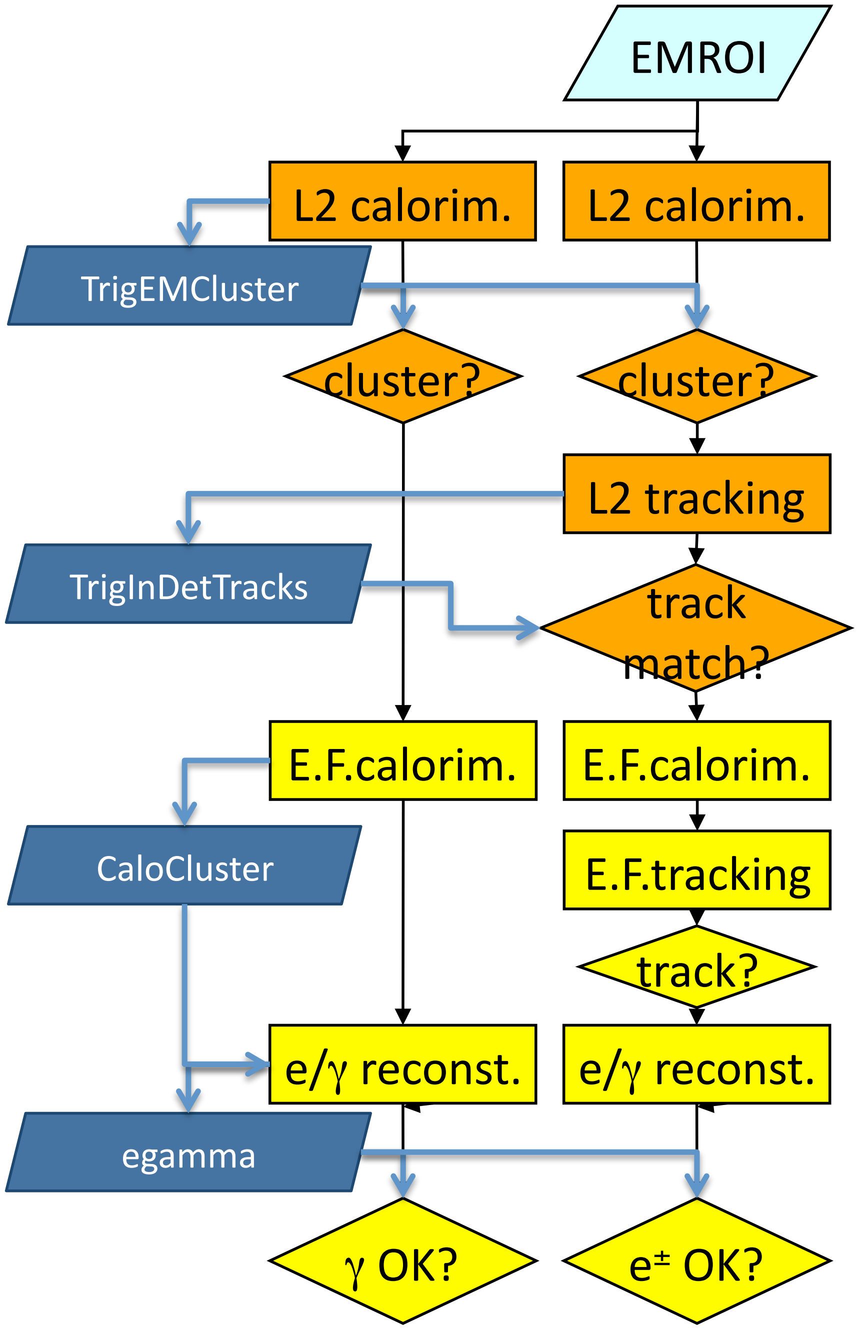
Ev.Filter seeded by Level 2
Offline reconstruction algorithms
Refined alignment and calibration



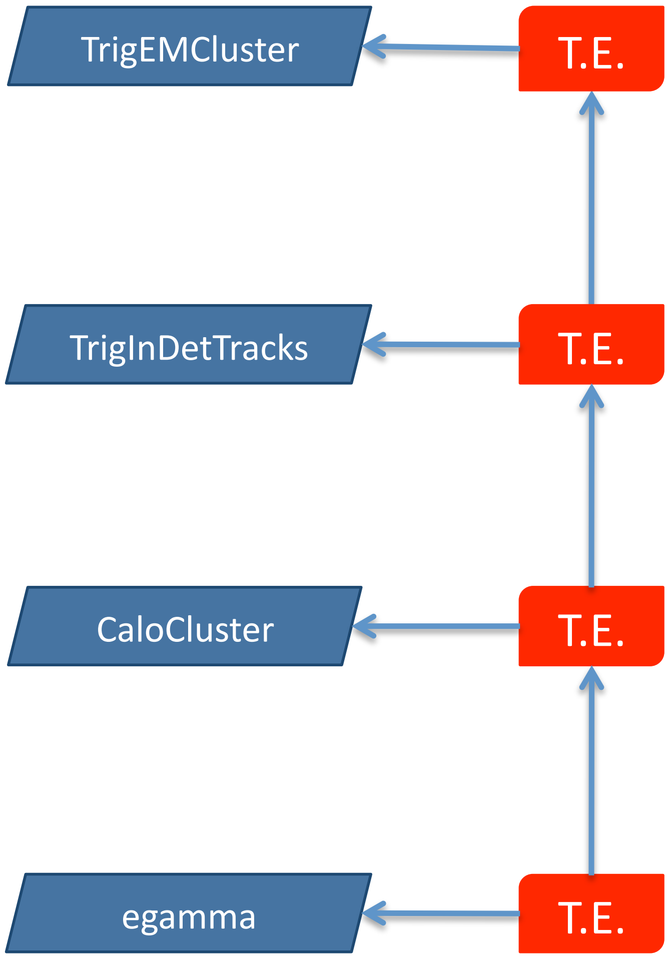
- Algorithm execution managed by Steering
  - Based on static trigger configuration
- Step-wise processing and early rejection
  - Chains stopped as soon as a step fails
  - Event passes if at least one chain is successful
- Feature Extraction algorithms (FEX)
  - Features (i.e. objects) cached to avoid time-consuming reconstruction
- Navigation links between TriggerElements
  - Navigation to objects possible in chain tree
- Trigger objects stored in ESD/AOD/DPD
  - Trigger “features” (id tracks, clusters, muon candidates, electron candidates, etc)
  - Navigation links (TriggerElements)
  - Trigger Configuration (file header)



- Algorithm execution managed by Steering
  - Based on static trigger configuration
- Step-wise processing and early rejection
  - Chains stopped as soon as a step fails
  - Event passes if at least one chain is successful
- Feature Extraction algorithms (FEX)
  - Features (i.e. objects) cached to avoid time-consuming reconstruction
- Navigation links between TriggerElements
  - Navigation to objects possible in chain tree
- Trigger objects stored in ESD/AOD/DPD
  - Trigger "features" (id tracks, clusters, muon candidates, electron candidates, etc)
  - Navigation links (TriggerElements)
  - Trigger Configuration (file header)



- Algorithm execution managed by Steering
  - Based on static trigger configuration
- Step-wise processing and early rejection
  - Chains stopped as soon as a step fails
  - Event passes if at least one chain is successful
- Feature Extraction algorithms (FEX)
  - Features (i.e. objects) cached to avoid time-consuming reconstruction
- Navigation links between TriggerElements
  - Navigation to objects possible in chain tree
- Trigger objects stored in ESD/AOD/DPD
  - Trigger "features" (id tracks, clusters, muon candidates, electron candidates, etc)
  - Navigation links (TriggerElements)
  - Trigger Configuration (file header)



Configuration Data Flow

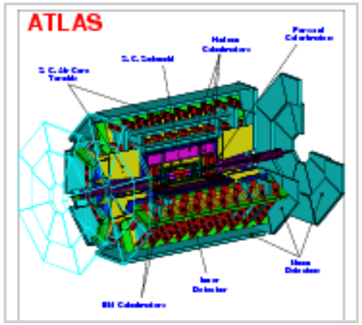
Preparation

TriggerDB
All configuration data

Configures

Stores decoded Trigger Menu

Data taking



Online Conditions Database

Encoded trigger decision
(trigger result from all 3 levels)

Decoded Trigger Menu

Reconstruction/ Trigger aware analysis

Trigger Result

- passed?, passed through?, prescaled?, last successful step in trigger execution?

Trigger EDM

- Trigger objects for trigger selection studies

Trigger Configuration

- Trigger names (version), prescales, pass throughs

access through TrigDecisionTool

ESD
AOD
DPD
TAG

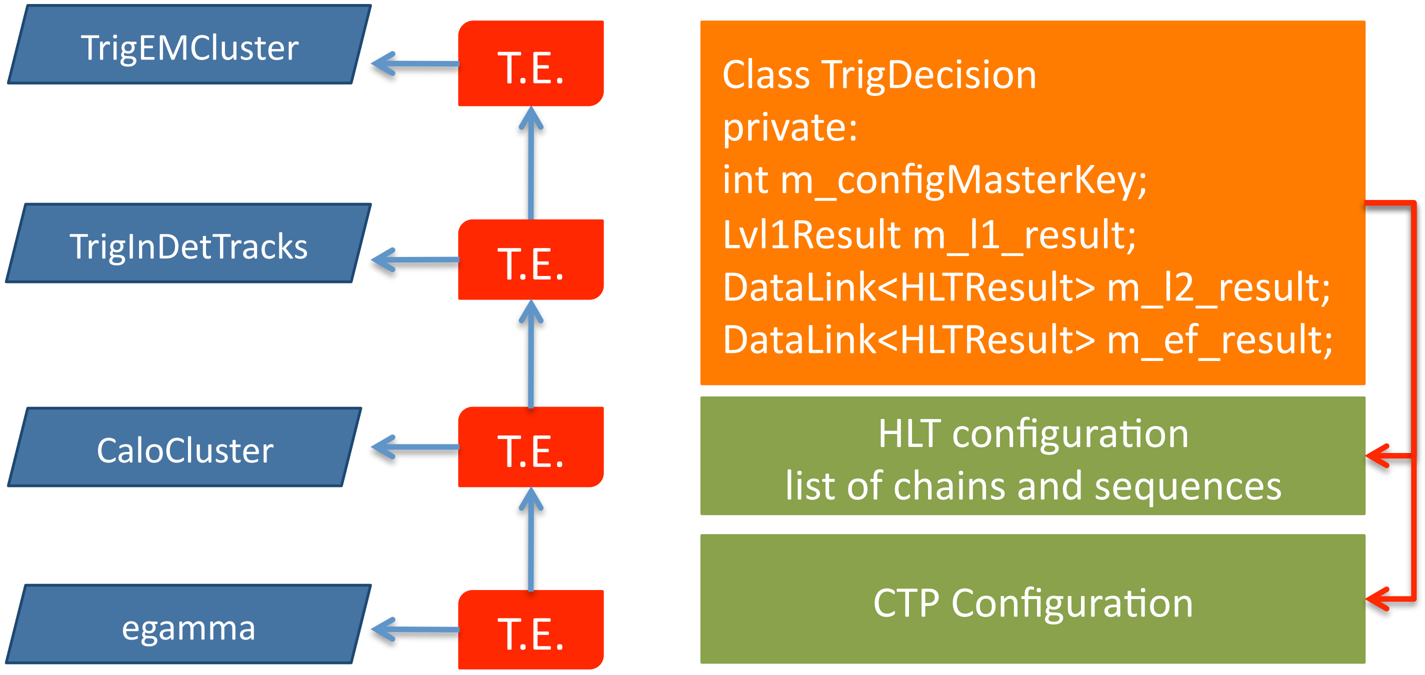
With decreasing amount of detail

TrigDecisionTool

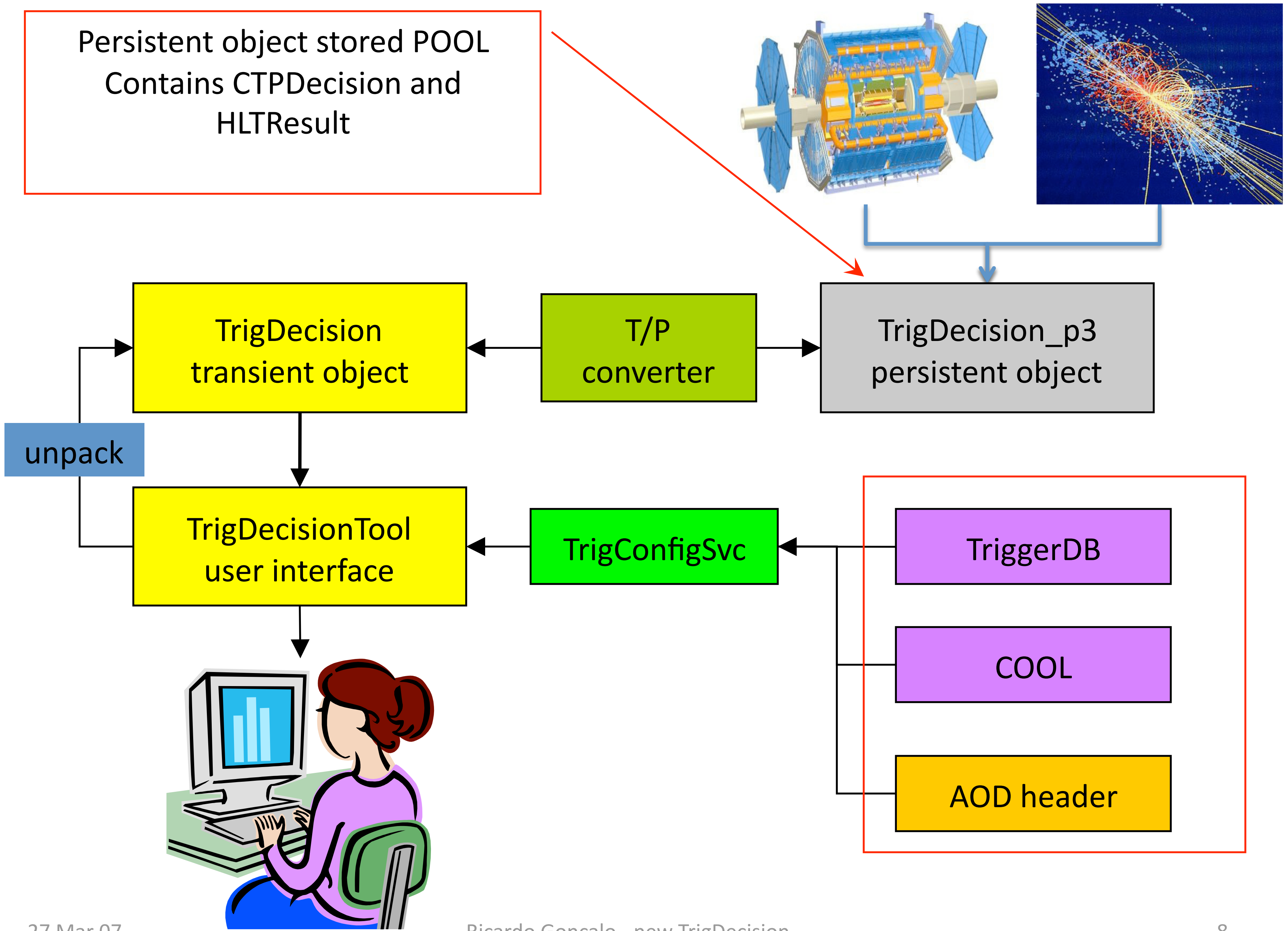
The TrigDecisionTool is the analysis interface to the trigger information in Athena

It reads the trigger decision object (TrigDecision) and the trigger configuration

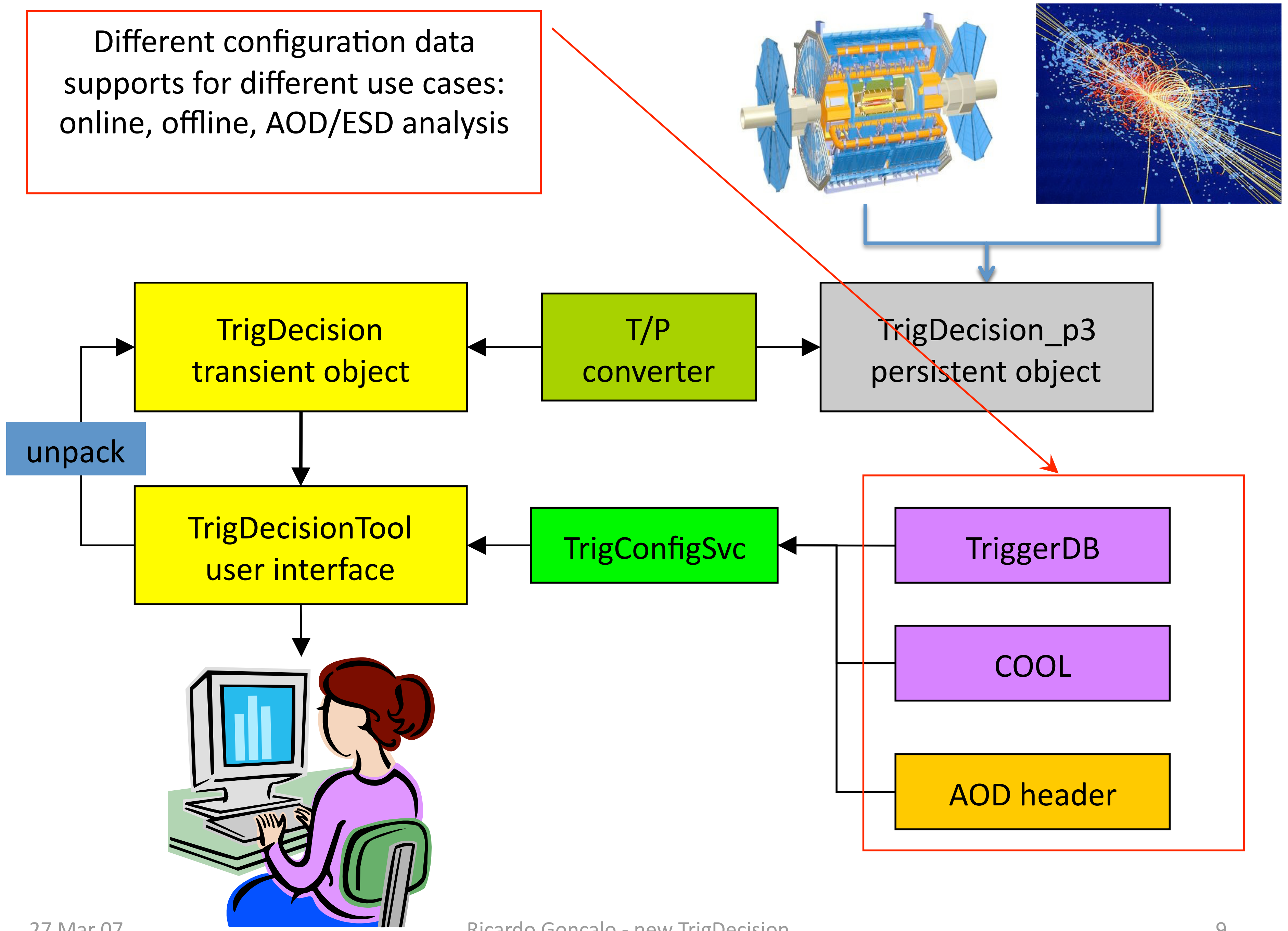
- Dynamic information (event-by-event)
  - What triggers passed/failed
  - Did they pass because of e.g. passthrough?
  - Navigate/retrieve trigger objects from each chain
- Configuration information:
  - Configured chains
  - Prescale and passthrough factors
  - (Trigger) stream tags

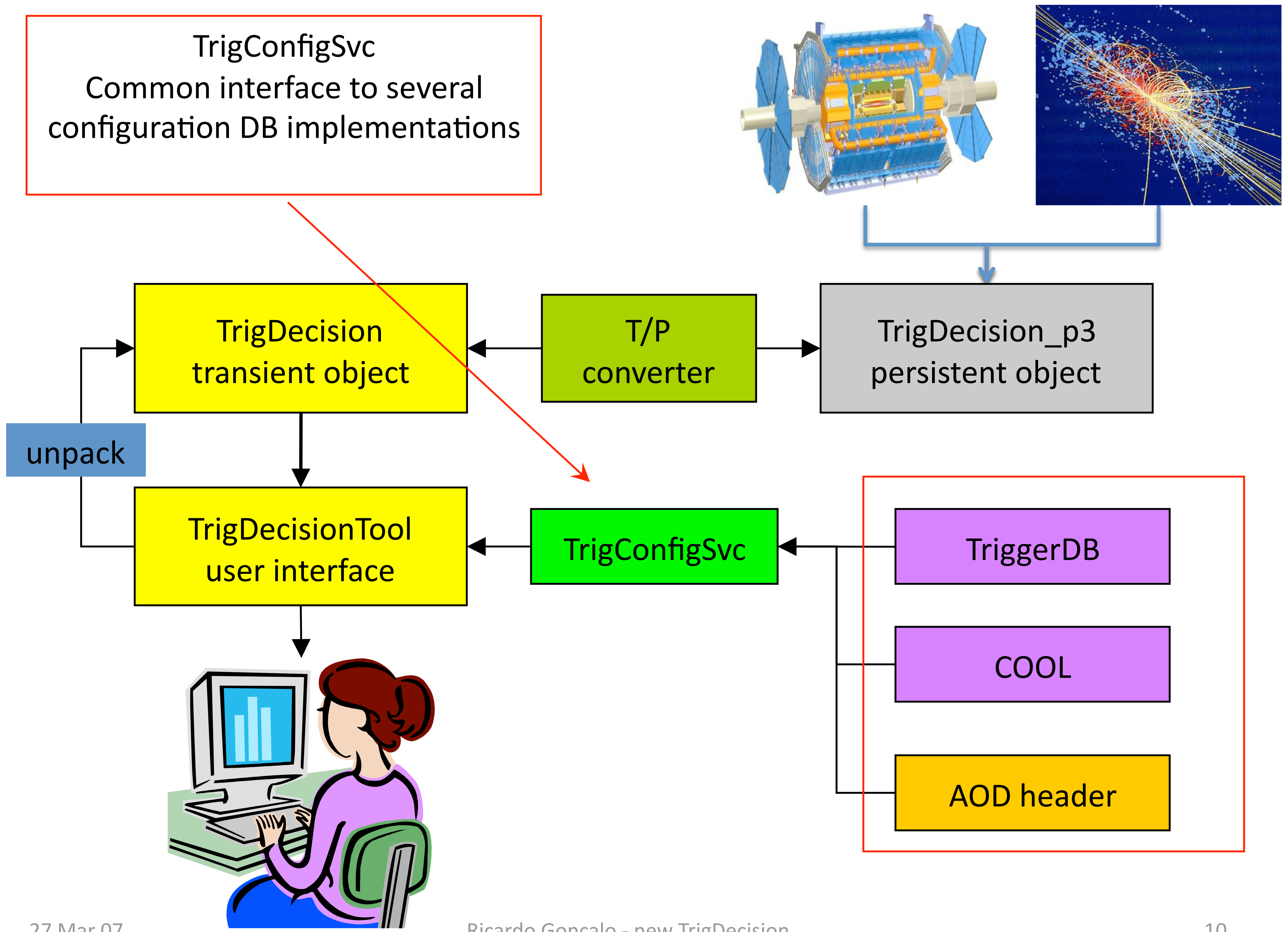


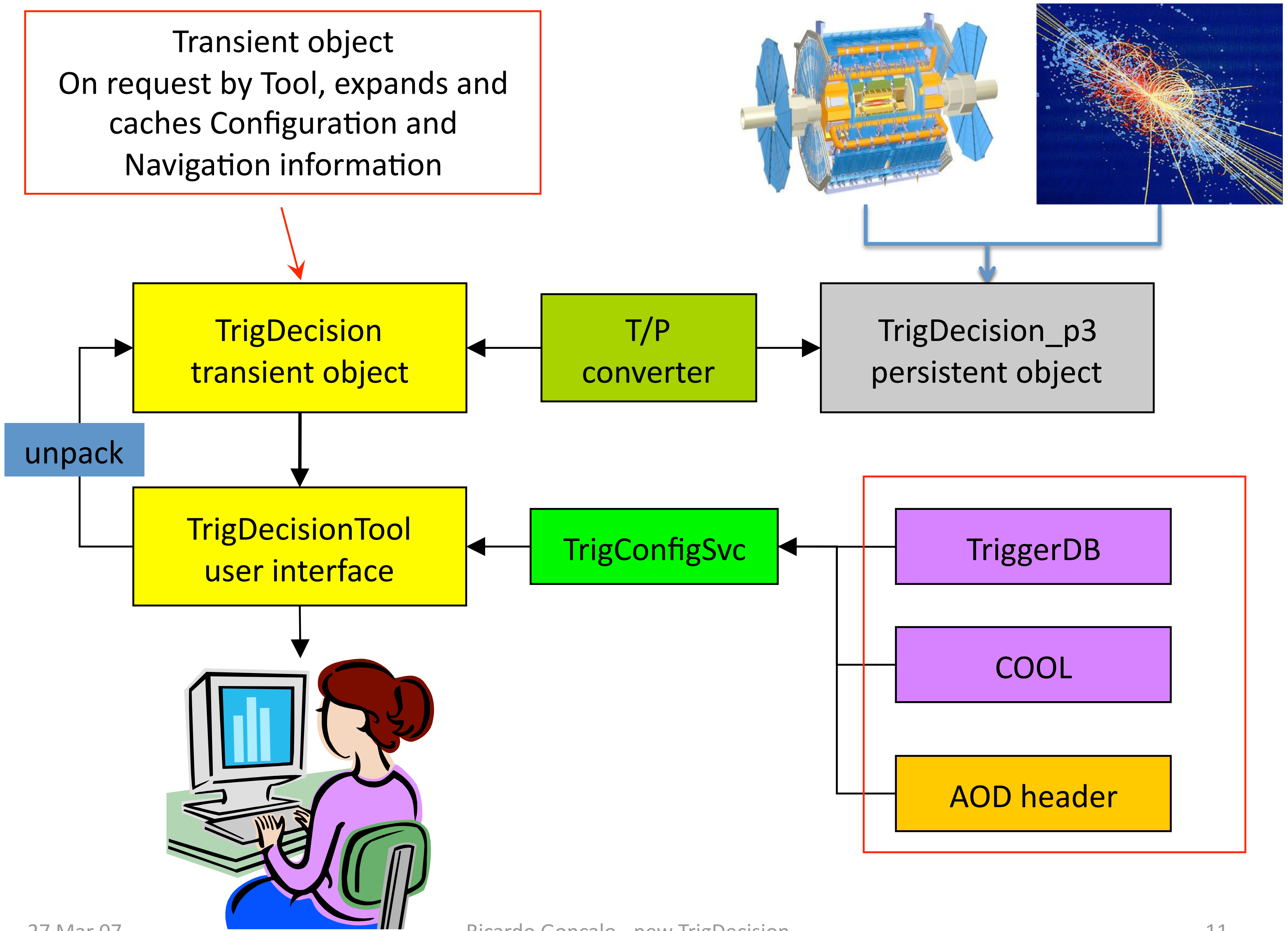
```
Class TrigDecision
private:
int m_configMasterKey;
Lvl1Result m_l1_result;
DataLink<HLTResult> m_l2_result;
DataLink<HLTResult> m_ef_result;
```

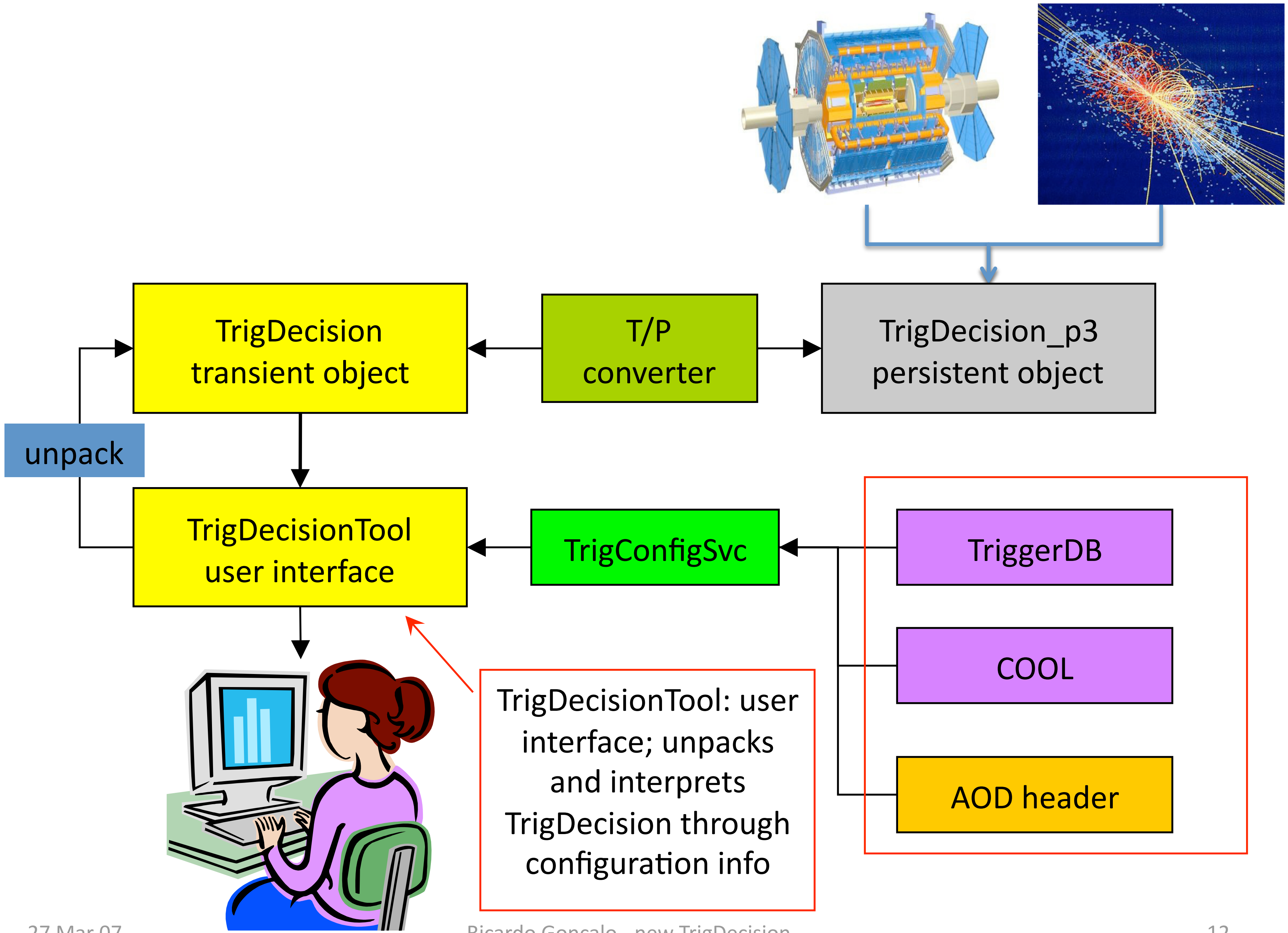

Persistent object stored POOL
Contains CTPDecision and
HLTResult
TrigDecision
transient object
T/P
converter
TrigDecision_p3
persistent object
unpack
TrigDecisionTool
user interface
TrigConfigSvc
TriggerDB
COOL
AOD header

Different configuration data supports for different use cases: online, offline, AOD/ESD analysis


TrigDecision transient object
T/P converter
TrigDecision_p3 persistent object
unpack
TrigDecisionTool user interface
TrigConfigSvc
TriggerDB
COOL
AOD header


TrigConfigSvc
Common interface to several configuration DB implementations
TrigDecision transient object
T/P converter
TrigDecision_p3 persistent object
unpack
TrigDecisionTool user interface
TrigConfigSvc
TriggerDB
COOL
AOD header


Transient object
On request by Tool, expands and caches Configuration and Navigation information
TrigDecision transient object
T/P converter
TrigDecision_p3 persistent object
unpack
TrigDecisionTool user interface
TrigConfigSvc
TriggerDB
COOL
AOD header


TrigDecision transient object
T/P converter
TrigDecision_p3 persistent object
unpack
TrigDecisionTool user interface
TrigConfigSvc
TriggerDB
COOL
AOD header

TrigDecisionTool: user interface; unpacks and interprets TrigDecision through configuration info

How-to use TrigDecisionTool in Athena

1. Add to your algorithm a ToolHandle to point to a TrigDecisionTool

```
private:
    ToolHandle<TrigDecisionTool> m_trigDec;
```

2. Initialize ToolHandle in constructor (adviseable to use as public tool)

```
MyAlgo::MyAlgo(const std::string &name, …
    m_trigDec("TrigDec::TrigDecisionTool/
    TrigDecisionTool”)
{
    declareProperty("TrigDecisionTool",
m_trigDec, ”The tool to access TrigDecision”);
```

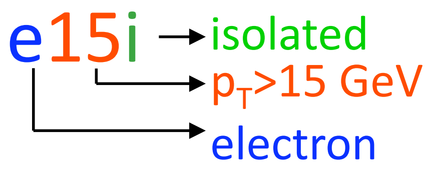
3. Retrieve tool in initialization

```
StatusCode sc = m_trigDec.retrieve();
if ( sc.isFailure() ) {
    (*m_log)<< MSG::ERROR<< “Help!”<< endreq;
    return sc;
}
```

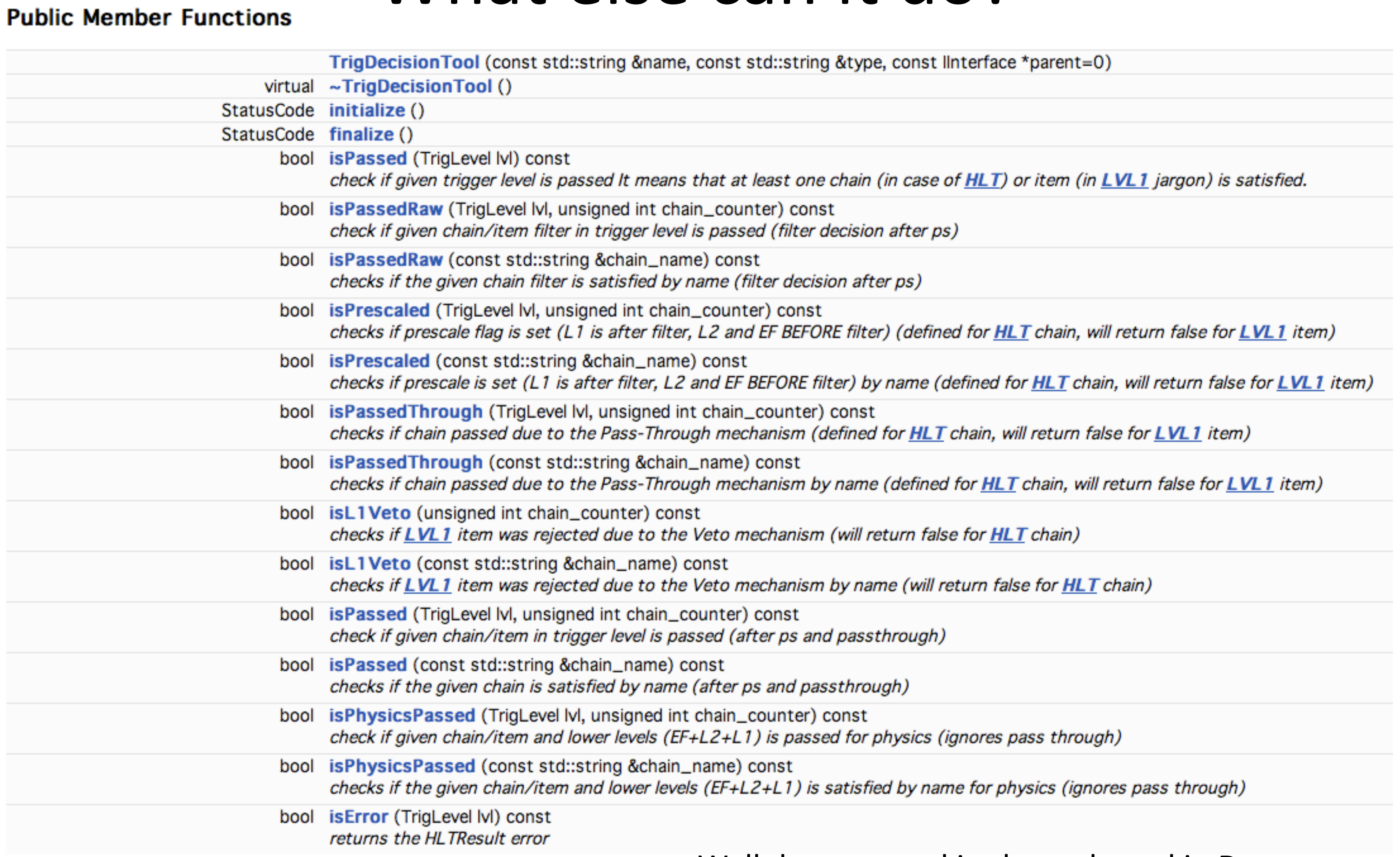
4. Use tool in execute

```
std::string sig_name(“L2_e15i”);
if (m_trigDec->isConfigured( sig_name )) {
    if ( m_trigDec->isPassed( sig_name )) {
        (*m_log) << MSG::INFO
                 << ”I’m happy!"
                 << endreq;
    }
}
```

More complete example in TrigDecisionTool14 twiki



What else can it do?



Public Member Functions

	TrigDecisionTool (const std::string &name, const std::string &type, const IInterface *parent=0)
virtual	**~TrigDecisionTool** ()
StatusCode	**initialize** ()
StatusCode	**finalize** ()
bool	**isPassed** (TrigLevel lvl) const *check if given trigger level is passed It means that at least one chain (in case of HLT) or item (in LVL1 jargon) is satisfied.*
bool	**isPassedRaw** (TrigLevel lvl, unsigned int chain_counter) const *check if given chain/item filter in trigger level is passed (filter decision after ps)*
bool	**isPassedRaw** (const std::string &chain_name) const *checks if the given chain filter is satisfied by name (filter decision after ps)*
bool	**isPrescaled** (TrigLevel lvl, unsigned int chain_counter) const *checks if prescale flag is set (L1 is after filter, L2 and EF BEFORE filter) (defined for HLT chain, will return false for LVL1 item)*
bool	**isPrescaled** (const std::string &chain_name) const *checks if prescale is set (L1 is after filter, L2 and EF BEFORE filter) by name (defined for HLT chain, will return false for LVL1 item)*
bool	**isPassedThrough** (TrigLevel lvl, unsigned int chain_counter) const *checks if chain passed due to the Pass-Through mechanism (defined for HLT chain, will return false for LVL1 item)*
bool	**isPassedThrough** (const std::string &chain_name) const *checks if chain passed due to the Pass-Through mechanism by name (defined for HLT chain, will return false for LVL1 item)*
bool	**isL1Veto** (unsigned int chain_counter) const *checks if LVL1 item was rejected due to the Veto mechanism (will return false for HLT chain)*
bool	**isL1Veto** (const std::string &chain_name) const *checks if LVL1 item was rejected due to the Veto mechanism by name (will return false for HLT chain)*
bool	**isPassed** (TrigLevel lvl, unsigned int chain_counter) const *check if given chain/item in trigger level is passed (after ps and passthrough)*
bool	**isPassed** (const std::string &chain_name) const *checks if the given chain is satisfied by name (after ps and passthrough)*
bool	**isPhysicsPassed** (TrigLevel lvl, unsigned int chain_counter) const *check if given chain/item and lower levels (EF+L2+L1) is passed for physics (ignores pass through)*
bool	**isPhysicsPassed** (const std::string &chain_name) const *checks if the given chain/item and lower levels (EF+L2+L1) is satisfied by name for physics (ignores pass through)*
bool	**isError** (TrigLevel lvl) const *returns the HLTResult error*

Well documented in the code and in Doxygen

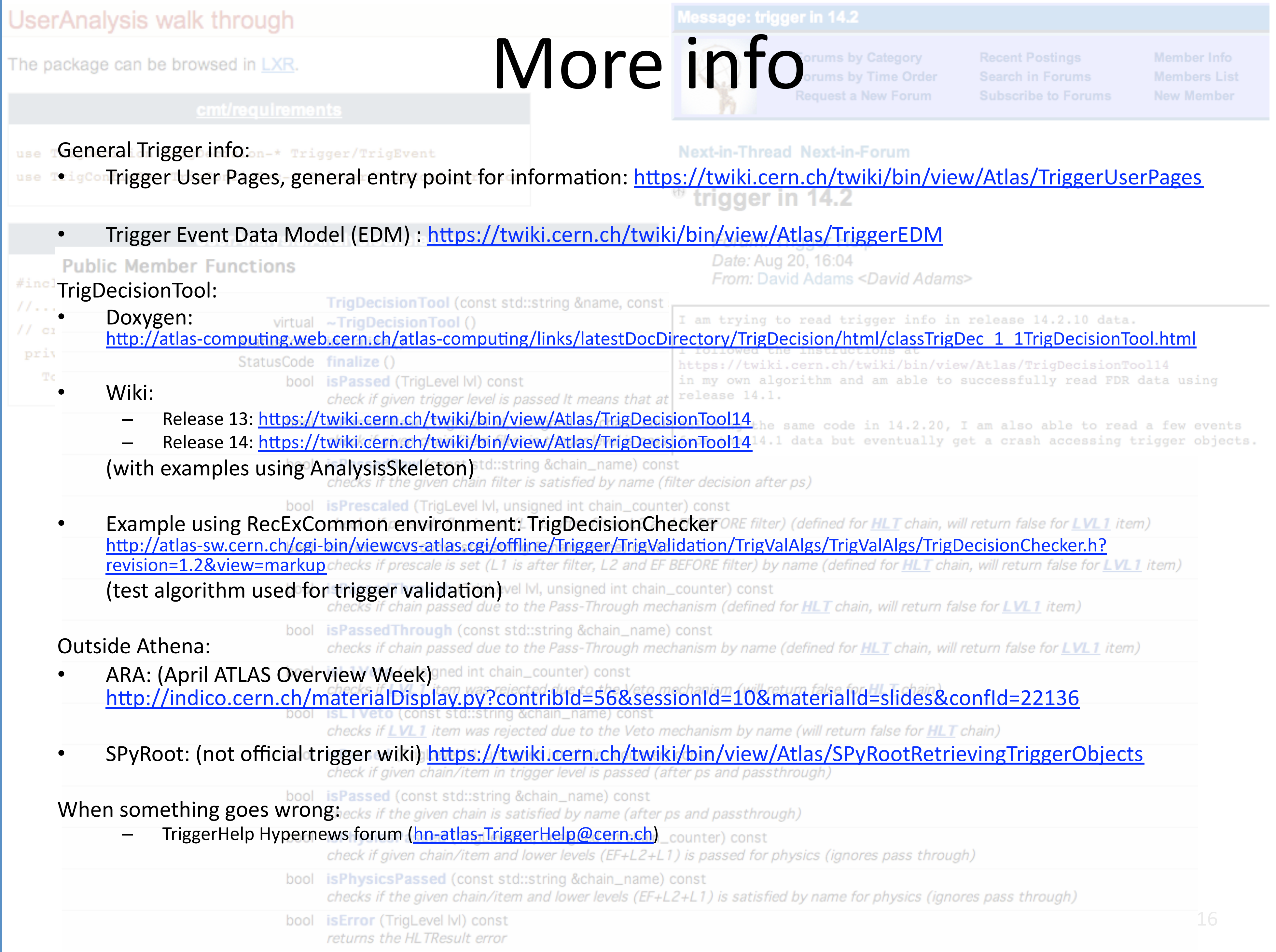
Outlook

- The trigger information available for analysis is now quite complete

- Work is currently ongoing (or at least in the pipeline):
  - Providing/improving access to trigger data outside Athena
  - Slimming navigation for inclusion in DPDs
    - Will be included without slimming in 2008
  - Cleaning up the TrigDecisionTool interface
  - Providing "navigation" links between offline objects and trigger navigation as a common feature

- The first priority: make the trigger run with real data!

  …And use this experience to find what else we need



More info

General Trigger info:

- Trigger User Pages, general entry point for information: https://twiki.cern.ch/twiki/bin/view/Atlas/TriggerUserPages
- Trigger Event Data Model (EDM) : https://twiki.cern.ch/twiki/bin/view/Atlas/TriggerEDM

TrigDecisionTool:

- Doxygen: http://atlas-computing.web.cern.ch/atlas-computing/links/latestDocDirectory/TrigDecision/html/classTrigDec_1_1TrigDecisionTool.html
- Wiki:
  - Release 13: https://twiki.cern.ch/twiki/bin/view/Atlas/TrigDecisionTool14
  - Release 14: https://twiki.cern.ch/twiki/bin/view/Atlas/TrigDecisionTool14

  (with examples using AnalysisSkeleton)
- Example using RecExCommon environment: TrigDecisionChecker http://atlas-sw.cern.ch/cgi-bin/viewcvs-atlas.cgi/offline/Trigger/TrigValidation/TrigValAlgs/TrigValAlgs/TrigDecisionChecker.h?revision=1.2&view=markup

  (test algorithm used for trigger validation)

Outside Athena:

- ARA: (April ATLAS Overview Week) http://indico.cern.ch/materialDisplay.py?contribId=56&sessionId=10&materialId=slides&confId=22136
- SPyRoot: (not official trigger wiki) https://twiki.cern.ch/twiki/bin/view/Atlas/SPyRootRetrievingTriggerObjects

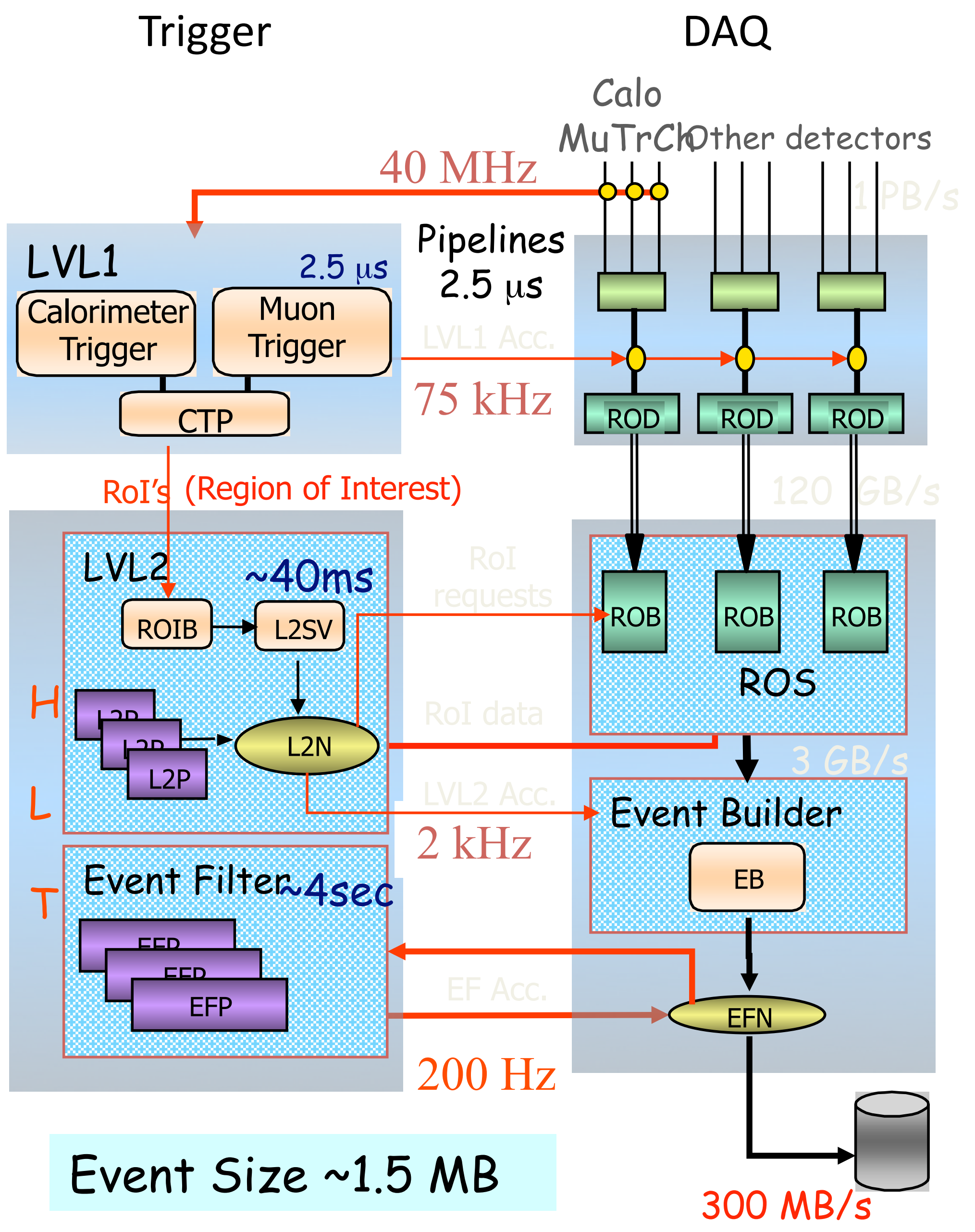
When something goes wrong:

- TriggerHelp Hypernews forum (hn-atlas-TriggerHelp@cern.ch)

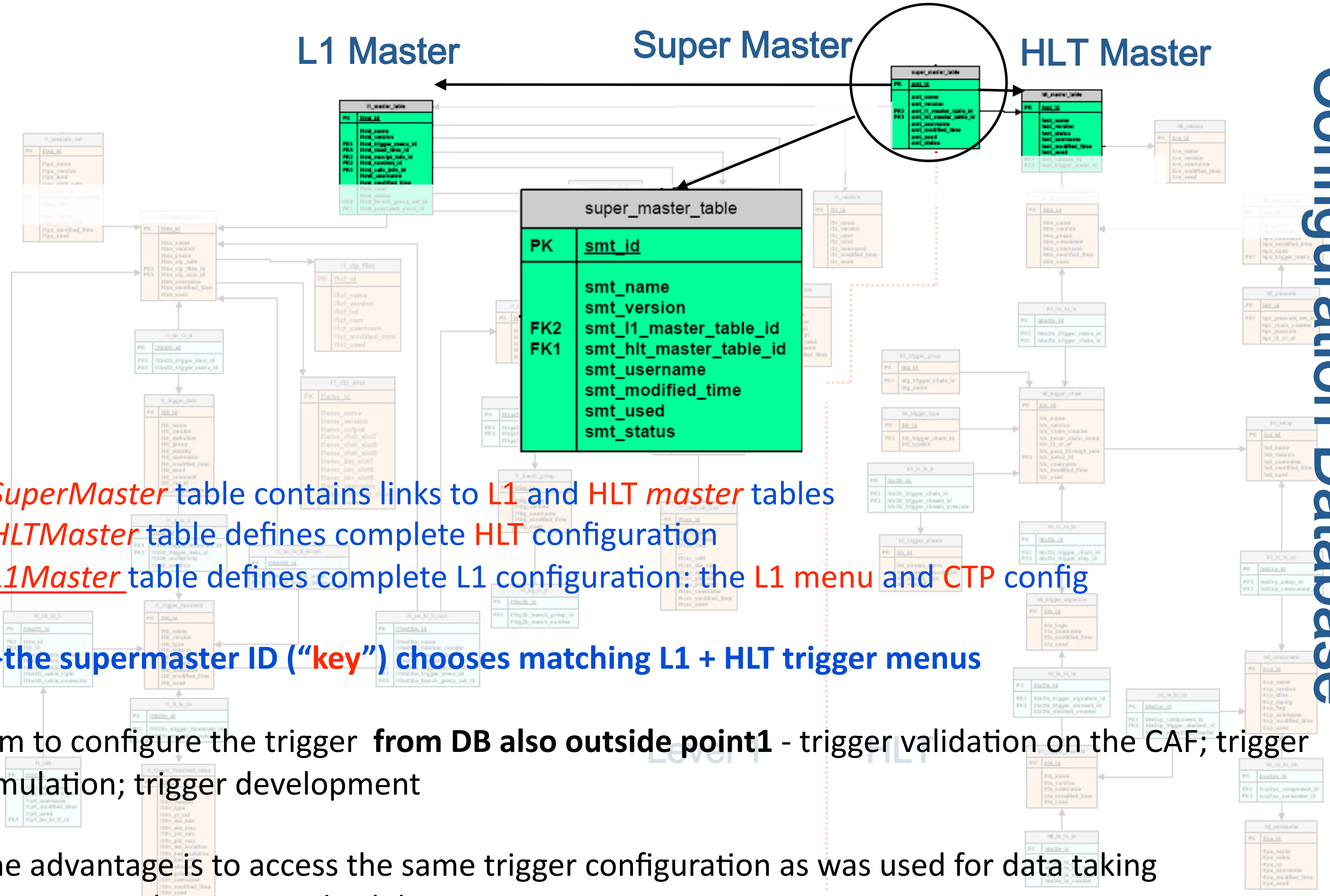
Backup

- Three trigger levels:
- Level 1:
  - Hardware based
  - Calorimeter and muons only
  - Latency 2.5 μs
  - Output rate ~75 kHz
- Level 2: ~500 farm nodes(*)
  - Only detector **”Regions of Interest” (RoI) processed - Seeded by level 1**
  - Fast reconstruction
  - Average execution time ~40 ms(*)
  - Output rate up to ~2 kHz
- Event Builder: ~100 farm nodes(*)
- Event Filter (EF):~1600 farm nodes(*)
  - Seeded by level 2
  - Potential full event access
  - Offline algorithms
  - Average execution time ~4 s(*)
  - Output rate up to ~200 Hz

(*) 8CPU (four-core dual-socket farm nodes at ~2GHz



Configuration Database



super_master_table	
PK	smt_id
	smt_name
	smt_version
FK2	smt_l1_master_table_id
FK1	smt_hlt_master_table_id
	smt_username
	smt_modified_time
	smt_used
	smt_status

- *SuperMaster* table contains links to L1 and HLT *master* tables
- *HLTMaster* table defines complete HLT configuration
- *L1Master* table defines complete L1 configuration: the L1 menu and CTP config

⇒**the supermaster ID ("key") chooses matching L1 + HLT trigger menus**

Aim to configure the trigger **from DB also outside point1** - trigger validation on the CAF; trigger simulation; trigger development

The advantage is to access the same trigger configuration as was used for data taking

➔ easy to achieve reproducibility

➔TriggerDB to be available at Tier0/1 (Oracle), and Tier2 (SQlite)

➔First running version in 14.2.10