AOD Trigger Analysis in release 13

Simon George
Ricardo Goncalo

Royal Holloway
University of London

US-ATLAS Analysis Jamboree BNL 6-10 August 2007

Outline

- Outline of analysis techniques
- Recap: trigger analysis with Rel.12
  - Trigger menu
  - TriggerDecision
- Rel.12 vs Rel.13
  - TriggerDecision vs TrigDecisionTool
  - Trigger menu changes
  - Details of TrigDecisionTool
- Analysis recipes
  - Athena-based AOD recipes
  - Basic TrigDecisionTool
  - Advanced TrigDecisionTool: Navigation
  - Re-run trigger decision
  - Other tools
- Rel. 13 status
  - What works and what doesn’t
- Trigger menus (point to TAPM)
- Conclusions

Trigger Analysis techniques

- **Things you can do with AOD in Athena.**
- Check chain status
  - Did the signature(s) I'm interested in pass or fail?
- Look at objects reconstructed by the trigger
  - Dig a bit deeper into why the chain passed or not
  - Compare with offline reconstruction
  - See AOD content talk
- Change the trigger cuts
  - Modify the properties of the hypothesis algorithms
  - Re-run the decision
  - Tune the cuts to optimise rate/efficiency

Recap: what exists in 12.0.6

- 12.0.6-7 used for CSC production;
  - Use CSC-06 configuration:

```
TrigT1ConfigVersion="CSC-06"
TrigHLTConfigVersion="CSC-06"
```

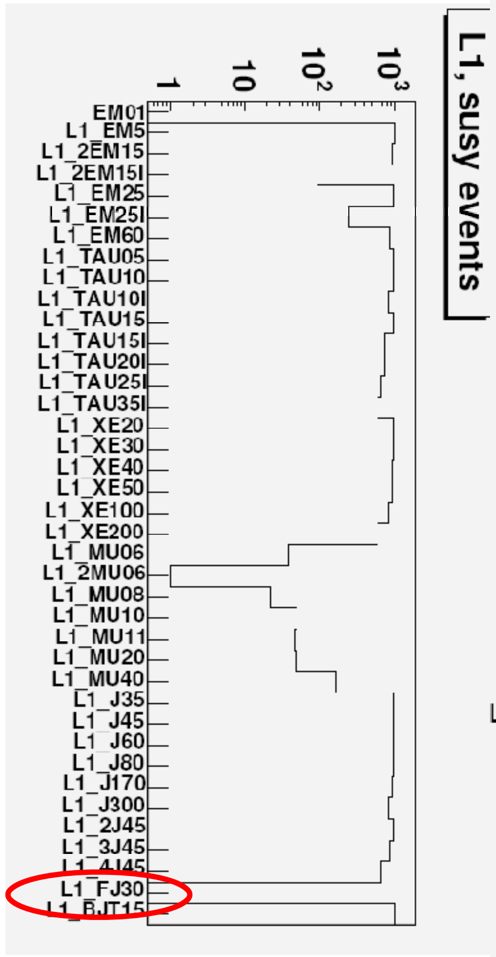
- Stream tests use somewhat different menu (STR-01)
- Main physics trigger signatures:

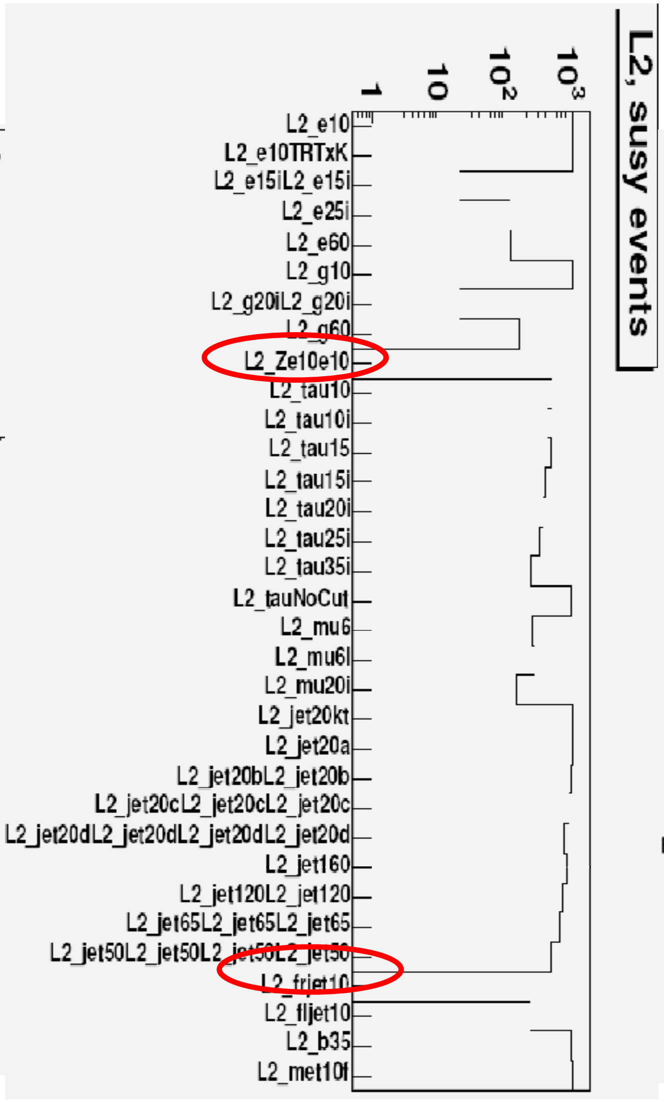
Slice	**HLT signatures**	**Starting from L1 items:**	**Comments**
Electron	2e15i, e25i, e60	2EM15, 2EM25, EM60	No isolation in L1 items; e25i ~realistic
Photon	2g20i, g60	2EM15I, EM60	Start from L1 items with isolation
Muon	mu6, mu20i	MU06, MU20	No isol; mu20i ~realistic; L1 p_T ordering
Tau	tau10i, tau15i, tau20i, tau25i, tau35i	TAU10i, TAU15i, TAU20i, TAU25i, TAU35i	
Jet	j160, 2j120, 3j65, 4j50	J45, 2J45, 3J45, 4J45	L1_J45 not realistic
ETmiss	met10	TAU05	Starts from L1 tau

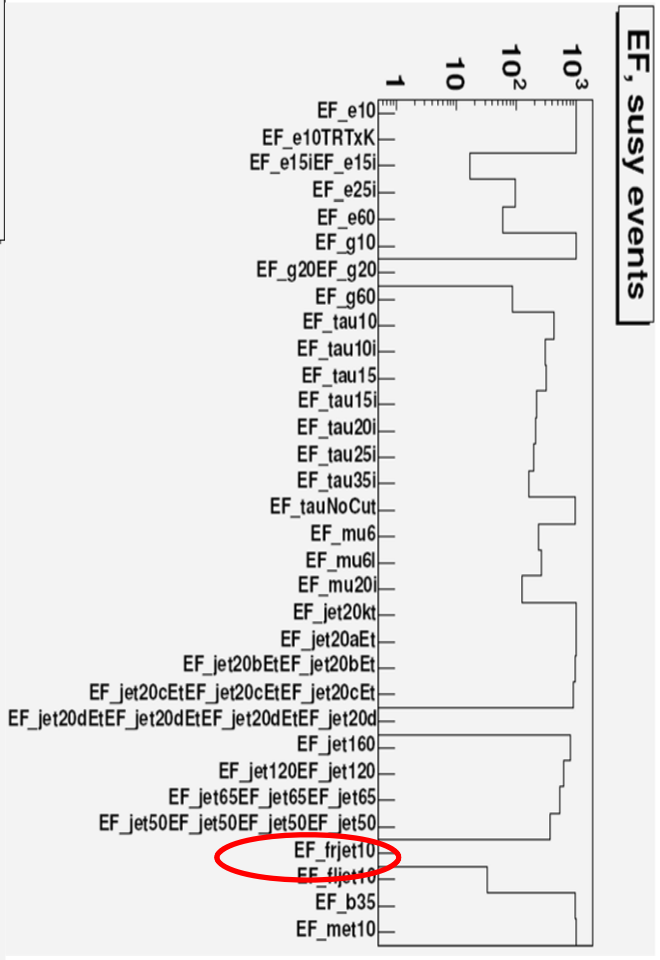
- In addition, technical or "expert" signatures for performance studies
  - tauNoCut, e10, jet20…
  - Needed in practice to allow trigger rerunning (must produce trigger objects)

What exists in 12.0.6 (cont.)

- Full list of trigger signatures, including technical signatures







What doesn’t exist in 12.0.6

- Most of the above signatures have not been optimized:
  - This means that the efficiencies and rates are more or less unrealistic
  - Means that asking if trigger passed is meaningless
- Most likely many useful signatures missing in each slice
- No prescale factors
  - These can easily be studied by hand, at least in simple cases
- Some existing signatures are “turned off”
  - b35 : run in “AcceptAll” mode
  - L2_Ze10e10 : e10 “technical” signature not storing TrigElectrons
  - frjet10 : problems with L1 bit encodin/non-optimal calibration
- Mixed triggers not possible
  - tauXX + ETmissYY; muXX + eYY …
- No ETsum or Jet Sum triggers yet

Using TriggerDecision – 12.0.6

- TriggerDecision: AOD/ESD object filled for all signatures

- Retrieve from StoreGate:
  - Default SG key is “`MyTriggerDecision`” (`TrigDecisionMaker/jobOfragment_TriggerDecisionMaker.py`)
- Query TriggerDecision:

```
 TriggerDecision* p_trigger_decision;
 sc = m_storeGate->retrieve(p_trigger_decision,”MyTriggerDecision”);
 if ( p_trigger_decision->isTriggerPassed() )
      m_log << MSG::DEBUG << “Some signature was satisfied!” << 
endreq;
 if ( p_trigger_decision->isDefined(“e25i”) ) {
      bool e_candidate_found = p_td->isPassed(“e25i”);
      m_log << MSG::DEBUG << “e25i signature was satisfied!” << 
endreq;
  }
```

- Filled for whatever signatures were configured
  - Retrieves configuration tables at initialization for L1/L2/EF
- Self-contained
  - No config needed to use it in AOD

Problems with 12.0.6

- TrigDecision
  - In order to be self-contained, each signature (chain) is stored as a string, along with its pass/fail status
  - not the most space-efficient approach and does not scale well as the menu grows
- Navigation
  - Outside of the trigger algorithms themselves, no uniform way to relate features from different algorithms, same RoI
  - Use of pointers in some classes as a workaround caused AOD size issues.
  - No way to know which features relate to successful chains, e.g. which egammas passed the e25i trigger?

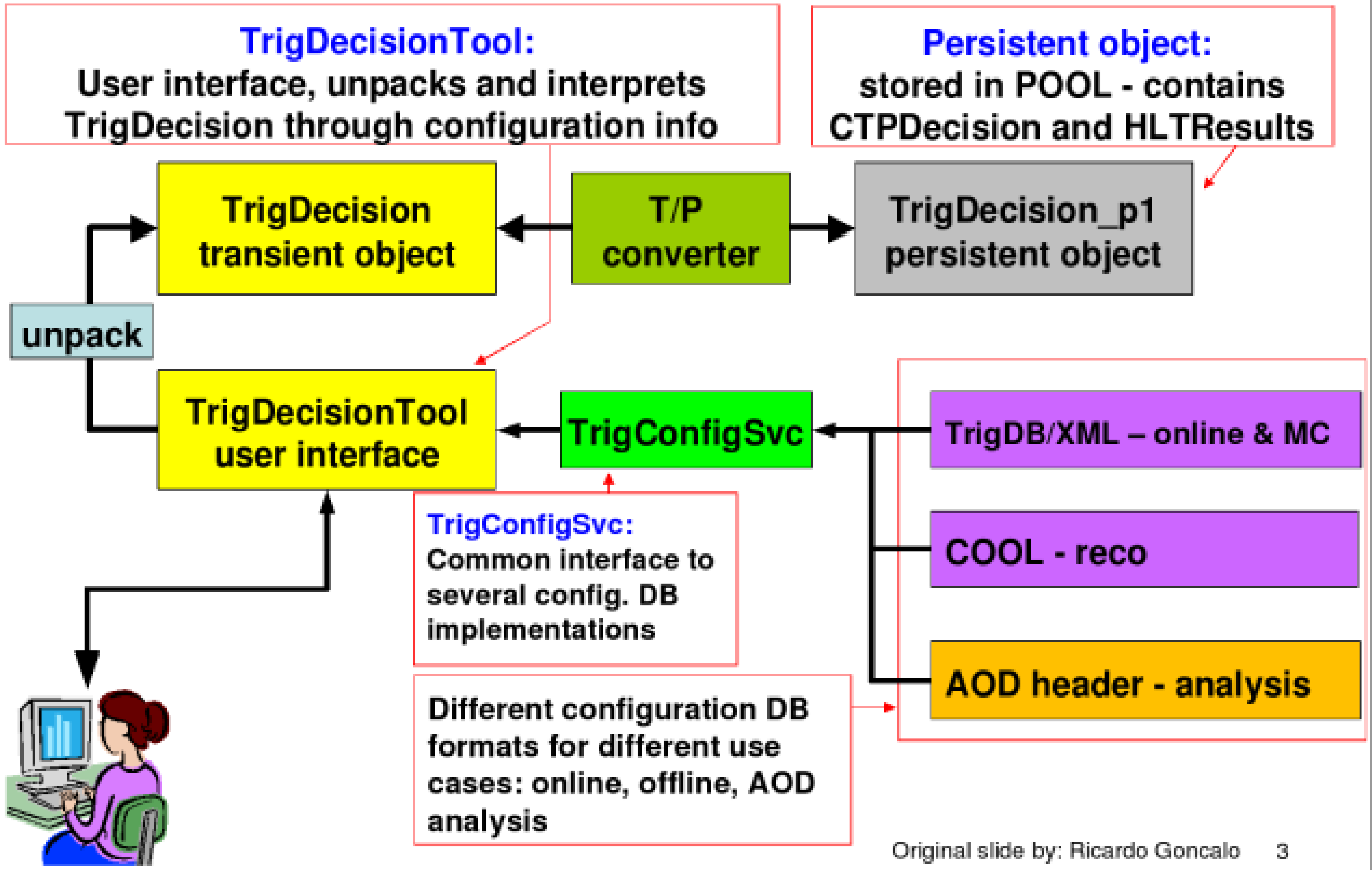
Tutorial for 12.0.6

- https://twiki.cern.ch/twiki/bin/view/Atlas/TriggerAnalysisTutorial1205
- You can still do a lot despite the noted deficiencies.

Release 12 vs. release 13

- TrigDecision/TrigDecisionTool replace TriggerDecision
  - What's new?
    - Compact: chains stored as bits
    - Use config service to interpret bits as chain names
    - Config saved in AOD (ideally per-run) so still self-contained
    - Access through tool to hide this
    - Navigation from chains to related features
  - Described on following pages
- Menu differences
  - More complete, but FDR menu planned for 13.0.30
  - Caveats about unoptimized triggers still apply
  - Some changes to chains and AOD content
  - New Steering supports most missing features

TrigDecision: Design


TrigDecisionTool:
User interface, unpacks and interprets TrigDecision through configuration info
Persistent object:
stored in POOL - contains CTPDecision and HLTResults
TrigDecision transient object
T/P converter
TrigDecision_p1 persistent object
unpack
TrigDecisionTool user interface
TrigConfigSvc
TrigDB/XML – online & MC
COOL - reco
AOD header - analysis
TrigConfigSvc:
Common interface to several config. DB implementations
Different configuration DB formats for different use cases: online, offline, AOD analysis

Original slide by: Ricardo Goncalo 3

TrigDecisionTool - details

- TrigDecisionTool is the interface for users to analyse the results of the trigger.
  - Brings together event-wise information and configuration information (which is run-wise)
  - Provides access to run-wise info using handles (like human readable chain names) which exist only in configuration.
  - It is a tool because it has to use number of other Athena services and tools.
  - It relies on the ESD/AOD object TrigDec::TrigDecision
- The TrigDec::TrigDecision is an object holding event-wise trigger information.
  - TrigDecision is produced in Athena, after trigger has run from RDO, and saved to AOD & ESD
  - Created by TrigDecisionMaker algorithm
  - Inputs are CTPDecision (LVL1), HLTResult (LVL2 & EF) and the trigger configuration
  - It is T/P separated and its persistent partners are defined in the TrigEventAthenaPool package.
  - **It should not be used directly, but via the TrigDecisionTool**

TrigDecisionTool (cont.)

- There are 4 levels of detail for accessing trigger config information, TrigDecisionDetails.
  1. NONE - if no configuration information is needed - and only simple information can be inspected
  2. CHAINS - where only chain information is needed - also no configuration is needed
  3. CONFIG - when the configuration is needed and user friendly methods are enabled
  4. FULL - when also trigger objects need to be accessed

- Reference documentation for TrigDecisionTool:
  – http://atlas-computing.web.cern.ch/atlas-computing/links/nightlyDevDirectory/AtlasOffline/latest_doxygen/InstallArea/doc//TrigDecision/html/classTrigDec_1_1TrigDecisionTool.html

- Simple, self-contained object TrigDec::TrigDecisionRecord also available
  – Originally made for SAN
  – Could be used in AOD for direct ROOT access

Analysis recipes

- Athena-based AOD recipes
- Basic TrigDecisionTool
- Advanced TrigDecisionTool: Navigation
- Re-run hypotheses
  - Similar to rel 12 tutorial, not covered today.
- Other tools – TrigDecisionTool clients

Reading AOD directly in ROOT?

- The recipes presented today all read AOD within Athena

- Reading AOD with ROOT is experimentally available in 13.0.X
- Not yet possible/supported for the trigger
- But progress is being made
- Plan to work in it this week
- Probably for rel 14

Recipes – RG slides go here

- Preliminary Instructions:
- 1) Set up 13.0.20
- 2) Check out PhysicsAnalysis/AnalysisCommon/UserAnalysis
- https://twiki.cern.ch/twiki/bin/view/Atlas/UserAnalysis
- and check you can make it and run an example so it is working
- 3) workaround for bug: check out and build latest TrigDecision???
- 4) Copy requirements, TrigAnalysisExample (.h and .cxx); make
- 5) find an AOD file & register it:
- e.g. RTT or one made yourself.
  pool_insertFileToCatalog /path/to/myAOD.pool.root
- FCregisterLFN -p /path/to/myAOD.pool.root -l myAOD.pool.root
- Tip: run checkFile.py to check the contents of the AOD file. You need to know what you are looking for, especially if it is a bare class, std::vector or another container.
- 6) Modify AnalysisSkeleton_topOptions.py for input AOD file and:
- # Trigger example algorithm
- TrigAnalysisExample = Algorithm("TrigAnalysisExample")
- TrigAnalysisExample.OutputLevel = INFO
- theApp.TopAlg += ["TrigAnalysisExample"]
- 7) run Athena
- **Coding notes:**
  - **templated code => you will get lots of DEBUG messages about unpacking and navigating which you did not write, but appear to be from your algorithm.**

How-to

- 1. Instantiate a TrigDecisionTool

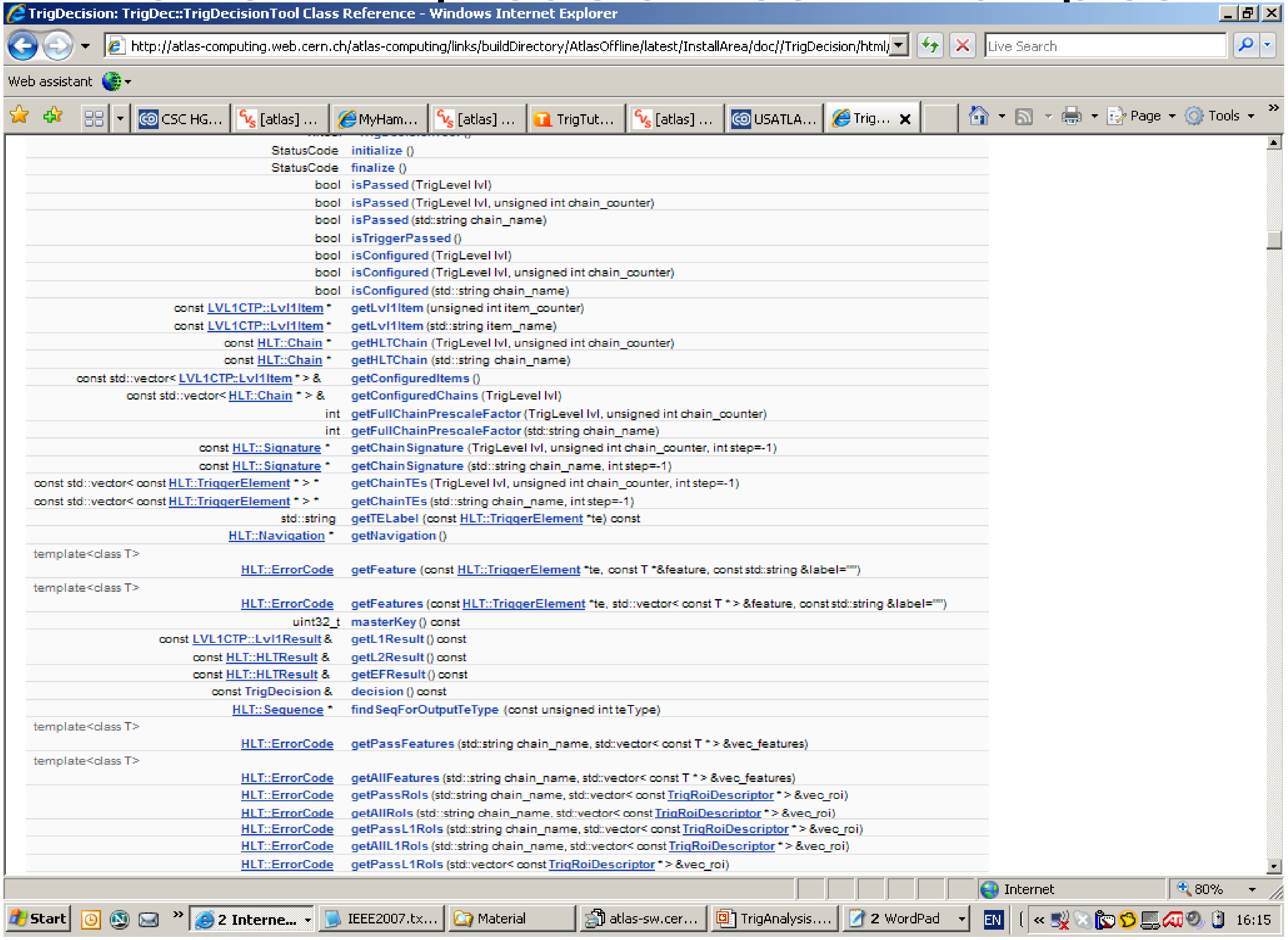
```
using namespace TrigDec;
TrigAnalysisExample::TrigAnalysisExample(const std::string& name,
                                         ISvcLocator* pSvcLocator)
  : Algorithm(name, pSvcLocator),
    m_storeGate("StoreGateSvc",name),
    m_trigDec("TrigDec::TrigDecisionTool",this),
    m_log(0)
{}
```

- 2. Access the TrigDecisionTool methods
  - To ask if trigger passed (*), which signatures were successful, etc

OR

- 3. Use the navigation to access trigger objects

The new TrigDecisionTool - Examples


TrigDecision: TrigDec::TrigDecisionTool Class Reference - Windows Internet Explorer
http://atlas-computing.web.cern.ch/atlas-computing/links/buildDirectory/AtlasOffline/latest/InstallArea/doc//TrigDecision/html/
Live Search
Web assistant
Page
Tools
Internet
80%
Start
16:15

StatusCode	initialize ()
StatusCode	finalize ()
bool	isPassed (TrigLevel lvl)
bool	isPassed (TrigLevel lvl, unsigned int chain_counter)
bool	isPassed (std::string chain_name)
bool	isTriggerPassed ()
bool	isConfigured (TrigLevel lvl)
bool	isConfigured (TrigLevel lvl, unsigned int chain_counter)
bool	isConfigured (std::string chain_name)
const LVL1CTP::Lvl1Item *	getLvl1Item (unsigned int item_counter)
const LVL1CTP::Lvl1Item *	getLvl1Item (std::string item_name)
const HLT::Chain *	getHLTChain (TrigLevel lvl, unsigned int chain_counter)
const HLT::Chain *	getHLTChain (std::string chain_name)
const std::vector< LVL1CTP::Lvl1Item * > &	getConfiguredItems ()
const std::vector< HLT::Chain * > &	getConfiguredChains (TrigLevel lvl)
int	getFullChainPrescaleFactor (TrigLevel lvl, unsigned int chain_counter)
int	getFullChainPrescaleFactor (std::string chain_name)
const HLT::Signature *	getChainSignature (TrigLevel lvl, unsigned int chain_counter, int step=-1)
const HLT::Signature *	getChainSignature (std::string chain_name, int step=-1)
const std::vector< const HLT::TriggerElement * > *	getChainTEs (TrigLevel lvl, unsigned int chain_counter, int step=-1)
const std::vector< const HLT::TriggerElement * > *	getChainTEs (std::string chain_name, int step=-1)
std::string	getTELabel (const HLT::TriggerElement *te) const
HLT::Navigation *	getNavigation ()
template<class T> HLT::ErrorCode	getFeature (const HLT::TriggerElement *te, const T *&feature, const std::string &label="")
template<class T> HLT::ErrorCode	getFeatures (const HLT::TriggerElement *te, std::vector< const T * > &feature, const std::string &label="")
uint32_t	masterKey () const
const LVL1CTP::Lvl1Result &	getL1Result () const
const HLT::HLTResult &	getL2Result () const
const HLT::HLTResult &	getEFResult () const
const TrigDecision &	decision () const
HLT::Sequence *	findSeqForOutputTeType (const unsigned int teType)
template<class T> HLT::ErrorCode	getPassFeatures (std::string chain_name, std::vector< const T * > &vec_features)
template<class T> HLT::ErrorCode	getAllFeatures (std::string chain_name, std::vector< const T * > &vec_features)
HLT::ErrorCode	getPassRoIs (std::string chain_name, std::vector< const TrigRoiDescriptor * > &vec_roi)
HLT::ErrorCode	getAllRoIs (std::string chain_name, std::vector< const TrigRoiDescriptor * > &vec_roi)
HLT::ErrorCode	getPassL1RoIs (std::string chain_name, std::vector< const TrigRoiDescriptor * > &vec_roi)
HLT::ErrorCode	getAllL1RoIs (std::string chain_name, std::vector< const TrigRoiDescriptor * > &vec_roi)
HLT::ErrorCode	getPassL1RoIs (std::vector< const TrigRoiDescriptor * > &vec_roi)

Event pass/fail info

- Simply find if each level passed (*)

```
(*m_log) << MSG::INFO << endreq;
 (*m_log) << MSG::INFO << "Exercise 0: Overall trigger decision:" << endreq;
 (*m_log) << MSG::INFO << "Pass state = " << (m_trigDec->isTriggerPassed() ? "pass" : "fail") << endreq;
 (*m_log) << MSG::INFO << "Level 1 was configured " << (m_trigDec->isConfigured(L1) ? "pass" : "fail") << endreq;
 (*m_log) << MSG::INFO << "Level 1 was successful " << (m_trigDec->isPassed(L1) ? "pass" : "fail") << endreq;
 (*m_log) << MSG::INFO << "Level 2 was configured " << (m_trigDec->isConfigured(L2) ? "pass" : "fail") << endreq;
 (*m_log) << MSG::INFO << "Level 2 was successful " << (m_trigDec->isPassed(L2) ? "pass" : "fail") << endreq;
 (*m_log) << MSG::INFO << "Event Filter was configudred " << (m_trigDec->isConfigured(EF) ? "pass" : "fail") << endreq;
 (*m_log) << MSG::INFO << "Event Filter was configudred " << (m_trigDec->isPassed(EF) ? "pass" : "fail") << endreq;
```

```
TrigAnalysisExample                INFO Exercise 0: Overall trigger decision:
TrigAnalysisExample                INFO Pass state = pass
TrigAnalysisExample                INFO Level 1 was configured pass
TrigAnalysisExample                INFO Level 1 was successful pass
TrigAnalysisExample                INFO Level 2 was configured pass
TrigAnalysisExample                INFO Level 2 was successful pass
TrigAnalysisExample                INFO Event Filter was configudred pass
TrigAnalysisExample                INFO Event Filter was configudred pass
```

(*) For simulated data (at least for now) this doesn’t really mean anything

Find if a signature passed

TrigDecisionTool:: isPassed(TrigLevel level,std::string chain)
TrigDecisionTool:: isPassed(TrigLevel level,std::string chain)
Note: templated code… doing a lot of stuff in the background

```
(*m_log) << MSG::INFO << endreq;
(*m_log) << MSG::INFO << "Exercise 1: Trigger element" << endreq;
(*m_log) << MSG::INFO << "Level 2: m_trigDec->isPassed(“L2_e25i”) "
        << (m_trigDec->isPassed("L2_e25i") ? "true" : "false") << endreq;
(*m_log) << MSG::INFO << "L2_e25i TriggerElement found" << endreq;
```

```
TrigAnalysisExample                          INFO Exercise 1: Trigger element
TrigAnalysisExample.TrigDec::TrigDecisionTool...  DEBUG Reading HLTResult for Lvl1ID=0 Event is Accepted=1 passedThrough=0
       HLTStatus=1554 size of rawResult=815 words (=3260 bytes)
TrigAnalysisExample.TrigDec::TrigDecisionTool...  DEBUG Found 18 chains in HLTResult
TrigAnalysisExample.TrigDec::TrigDecisionTool...  DEBUG Counter = 64 success (raw) = 1 pass-through = 0 prescaled = 0 lastActiveStep = 3
                    name =
TrigAnalysisExample.TrigDec::TrigDecisionTool...  DEBUG Counter = 74 success (raw) = 1 pass-through = 0 prescaled = 0 lastActiveStep = 2
                    name =
TrigAnalysisExample.TrigDec::TrigDecisionTool...  DEBUG Counter = 400 success (raw) = 1 pass-through = 1 prescaled = 0 lastActiveStep = 3
                    name =
TrigAnalysisExample.TrigDec::TrigDecisionTool...  DEBUG Counter = 320 success (raw) = 0 pass-through = 1 prescaled = 0 lastActiveStep = 1
                    name =
```

… and 130 DEBUG lines later…

```
TrigAnalysisExample.TrigDec::TrigDecisionTool     DEBUG Chain L2_e25i found in L2
TrigAnalysisExample                          INFO Level 2: m_trigDec->isPassed(L2,L2_e25i) false
```

New stuff! …first a silly example

- Get the HLT::Chain from TrigDecisionTool

```
const HLT::Chain* p_chn = m_trigDec->getHLTChain("L2_e25i");
 unsigned int code = 0;
 if (p_chn) {
   code = p_chn->getChainCounter();
   (*m_log) << MSG::INFO << "Trigger element for L2_e25i has counter id " << code << endreq;
   (*m_log) << MSG::INFO << "Level 2: m_trigDec->isPassed(L2," << code << ")? = "
           << (m_trigDec->isPassed("L2_e25i") ? "true" : "false") << endreq;
   (*m_log) << MSG::INFO << "L2_e25i chain counter id = " << code << endreq;
 } else {
   (*m_log) << MSG::INFO << "Could not retrieve L2_e25i chains" << endreq;
 }
```

```
TrigAnalysisExample.TrigDec::TrigDecisionTool     DEBUG Chain L2_e25i found in L2
TrigAnalysisExample                               INFO Trigger element for L2_e25i has counter id 67
TrigAnalysisExample.TrigDec::TrigDecisionTool     DEBUG Item L2_e25i not found.
TrigAnalysisExample.TrigDec::TrigDecisionTool     DEBUG Chain L2_e25i found in L2
TrigAnalysisExample                               INFO Level 2: m_trigDec->isPassed(L2,67)? = false
TrigAnalysisExample                               INFO L2_e25i chain counter id = 67
```

More…prescale factors

- TrigDecisionTool::prescaleFactor()

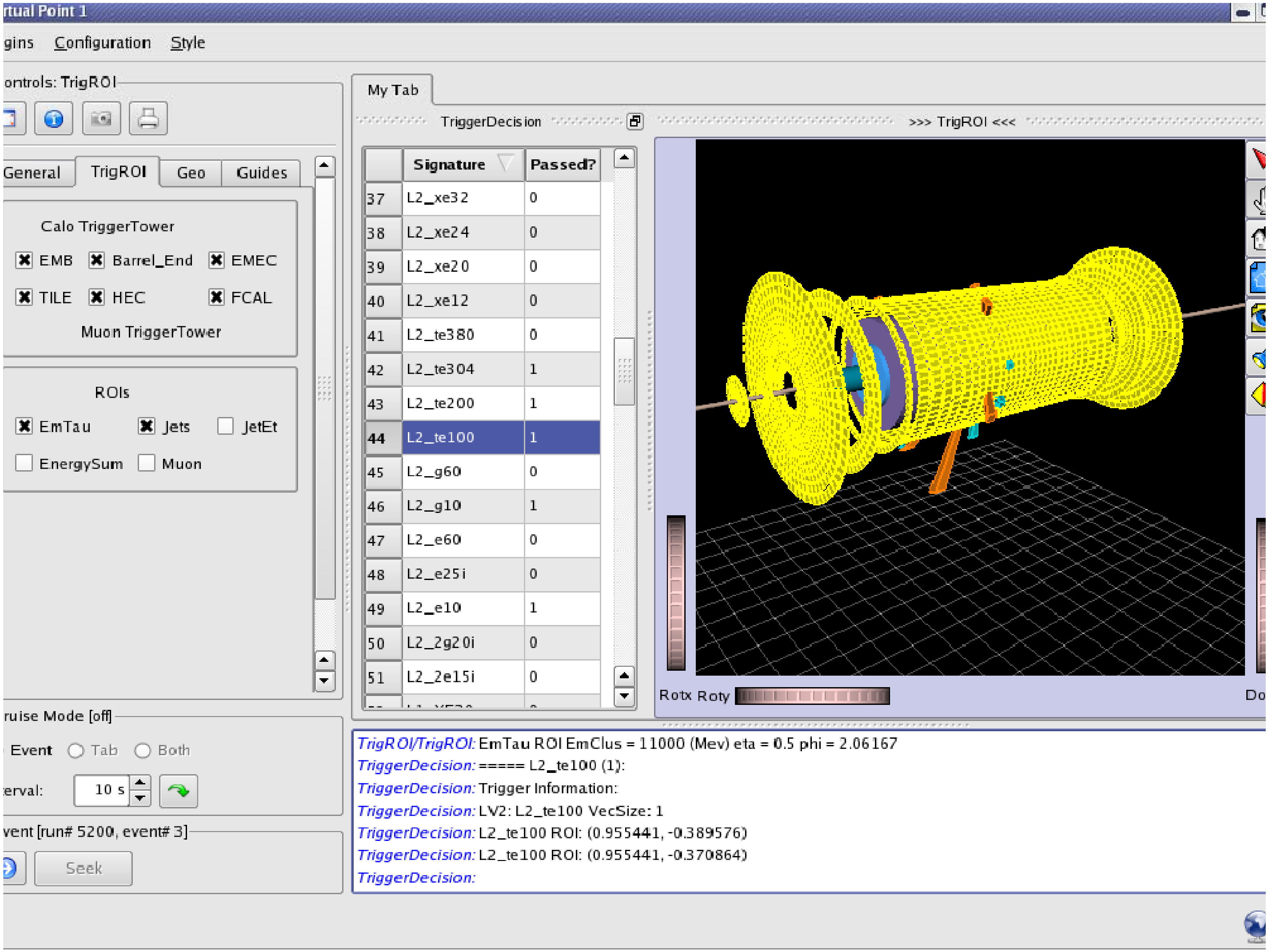
```
TrigAnalysisExample                                 INFO 177 items in L1
TrigAnalysisExample                                 INFO |      L1 item      | prescale | pass |
TrigAnalysisExample                                 INFO |            L1_1EM1|         1|      pass|
TrigAnalysisExample                                 INFO |           L1_2EM15|         1|      pass|
TrigAnalysisExample                                 INFO |          L1_2EM15i|         1|      fail|
TrigAnalysisExample                                 INFO |           L1_2EM25|         1|      fail|
TrigAnalysisExample                                 INFO |          L1_2EM25i|         1|      fail|
TrigAnalysisExample                                 INFO |           L1_1EM60|         1|      fail|
TrigAnalysisExample                                 INFO |            L1_XE20|         1|      fail|
TrigAnalysisExample                                 INFO |            L1_1MU6|         1|      fail|
TrigAnalysisExample                                 INFO |            L1_2MU6|         1|      fail|
TrigAnalysisExample                                 INFO |            L1_1MU8|         1|      fail|
TrigAnalysisExample                                 INFO |           L1_1MU10|         1|      fail|
TrigAnalysisExample                                 INFO |           L1_1MU11|         1|      fail|
TrigAnalysisExample                                 INFO |           L1_1MU20|         1|      fail|
TrigAnalysisExample                                 INFO |           L1_1MU40|         1|      fail|
TrigAnalysisExample                                 INFO |           L1_1TAU5|         1|      fail|
TrigAnalysisExample                                 INFO |          L1_1TAU10|         1|      pass|
TrigAnalysisExample                                 INFO |         L1_1TAU10i|         1|      pass|
TrigAnalysisExample                                 INFO |          L1_1TAU15|         1|      fail|
TrigAnalysisExample                                 INFO |         L1_1TAU15i|         1|      fail|
TrigAnalysisExample                                 INFO |         L1_1TAU20i|         1|      fail|
TrigAnalysisExample                                 INFO |         L1_1TAU25i|         1|      fail|
TrigAnalysisExample                                 INFO |         L1_1TAU35i|         1|      fail|
TrigAnalysisExample                                 INFO |   L1_1TAU17I_XE30|         1|      fail|
TrigAnalysisExample                                 INFO |           L1_BGRP0|       100|      fail|
INFO Configured HLT menu - master key=0
INFO |level|   chain name   | id |prescale fact|passthr.fact|P/T|  lower  chain   | id | pass |after prsc|
INFO |  L2 |          L2_e10|  64|            1|           0|  0|             EM01|  Y|  Y| pass | pass |
INFO |  L2 |         L2_e25i|  67|            1|           0|  0|             EM01|  N|  N| fail | fail |
INFO |  L2 |          L2_e60|  68|            1|           0|  0|             EM01|  N|  N| fail | fail |
INFO |  L2 |        L2_2e15i|  73|            1|           0|  0|             EM01|  N|  N| fail | fail |
INFO |  L2 |          L2_g10|  74|            1|           0|  0|             EM01|  Y|  Y| pass | pass |
INFO |  L2 |        L2_2g20i|  76|            1|           0|  0|             EM01|  N|  N| fail | fail |
INFO |  L2 |          L2_g60|  77|            1|           0|  0|             EM01|  N|  N| fail | fail |
INFO |  L2 |        tau10_L2| 320|            1|           1|  1|             EM01|  N|  Y| fail | fail |
INFO |  L2 |       tau10i_L2| 321|            1|           1|  1|             EM01|  N|  Y| fail | fail |
INFO |  L2 |        tau15_L2| 322|            1|           1|  1|             EM01|  N|  Y| fail | fail |
INFO |  L2 |       tau15i_L2| 323|            1|           1|  1|             EM01|  N|  Y| fail | fail |
INFO |  L2 |       tau20i_L2| 324|            1|           1|  1|             EM01|  N|  Y| fail | fail |
INFO |  L2 |       tau25i_L2| 325|            1|           1|  1|             EM01|  N|  Y| fail | fail |
INFO |  L2 |       tau35i_L2| 326|            1|           1|  0|             EM01|  N|  N| fail | fail |
```

V

```
TrigAnalysisExample.TrigDec::TrigDecisionTool     DEBUG Chain EF_e25i found in EF
TrigAnalysisExample                              INFO Got a vector of 0 TEs.
TrigAnalysisExample.TrigDec::TrigDecisionTool     DEBUG Chain L2_e25i found in L2
TrigAnalysisExample                              INFO Chain L2_e25i: Counter = 67 success (raw) = 0 pass-
      through = 0 prescaled = 0 lastActiveStep = 0              name = L2_e25i passed: 0
TrigAnalysisExample.TrigDec::TrigDecisionTool     DEBUG Chain L2_e25i found in L2
TrigAnalysisExample                              INFO For L2_e25i, 1 TEs found.
TrigAnalysisExample.TrigDec::TrigDecisionTool     DEBUG Chain L2_e25i found in L2
TrigAnalysisExample                              INFO For L2_e25i, prescale factor is 1
TrigAnalysisExample                              INFO TE is active
```

Trigger data via other analysis tools

- These tools are clients of TrigDecisionTool
- Event displays
  - Atlantis (TrigJiveXML)
  - VP1
  - Status: both working with new TrigDecision (we think)
- Analysis tools
  - EventView
    - Status: to be done for rel.13
  - CBNTAA
    - Status: to be done for rel.13
- Trigger rate tool
  - Status: being done
- Your analysis
  - Starts here…



Calo TriggerTower

[✓ EMB] [✓ Barrel_End] [✓ EMEC]

[✓ TILE] [✓ HEC] [✓ FCAL]

Muon TriggerTower

ROIs

[✓ EmTau] [✓ Jets] JetEt

EnergySum Muon

ruise Mode [off]

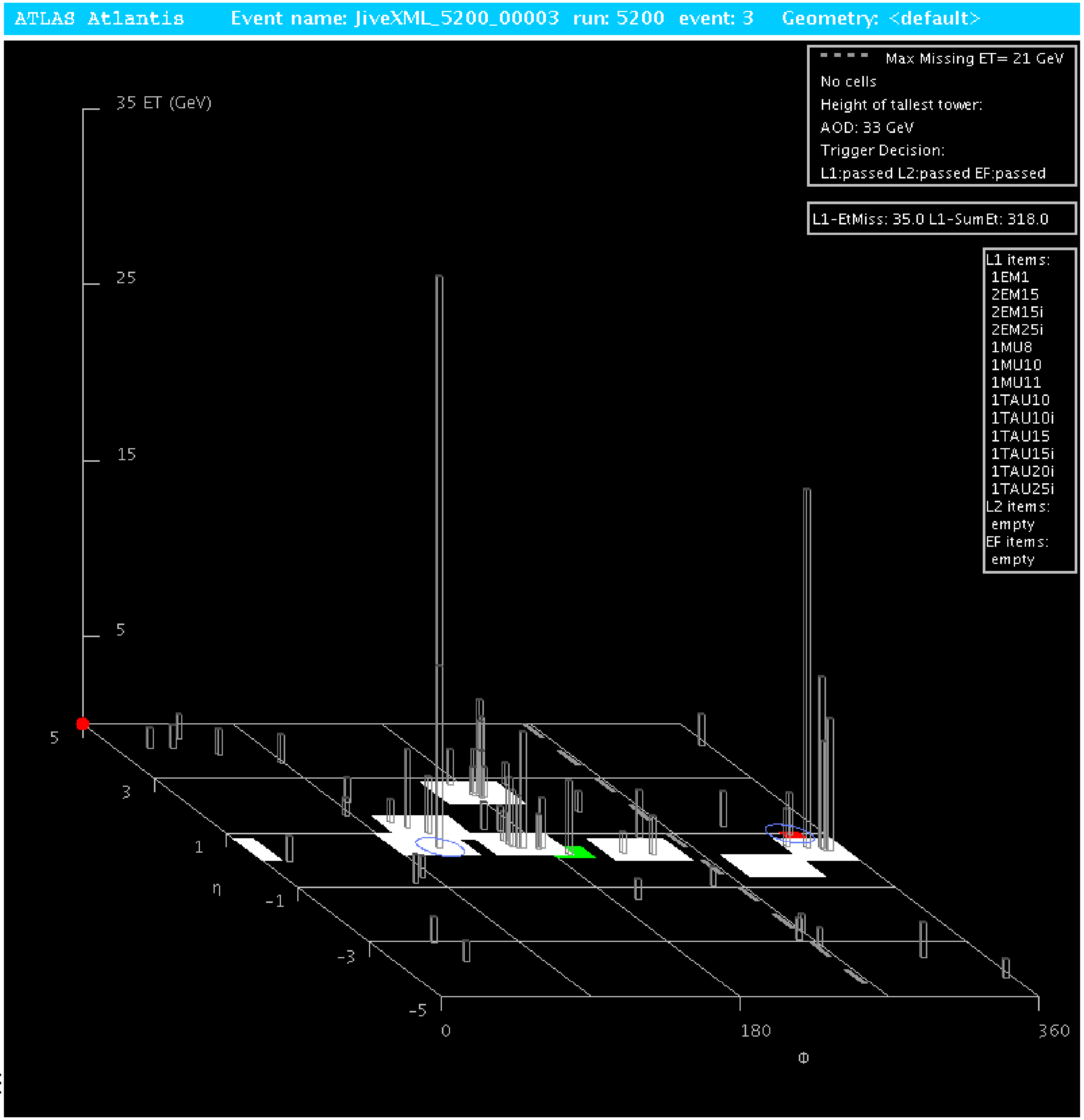
Event Tab Both

erval: 10 s

vent [run# 5200, event# 3]

	Signature	Passed?
37	L2_xe32	0
38	L2_xe24	0
39	L2_xe20	0
40	L2_xe12	0
41	L2_te380	0
42	L2_te304	1
43	L2_te200	1
44	L2_te100	1
45	L2_g60	0
46	L2_g10	1
47	L2_e60	0
48	L2_e25i	0
49	L2_e10	1
50	L2_2g20i	0
51	L2_2e15i	0

```
TrigROI/TrigROI: EmTau ROI EmClus = 11000 (Mev) eta = 0.5 phi = 2.06167
TriggerDecision: ===== L2_te100 (1):
TriggerDecision: Trigger Information:
TriggerDecision: LV2: L2_te100 VecSize: 1
TriggerDecision: L2_te100 ROI: (0.955441, -0.389576)
TriggerDecision: L2_te100 ROI: (0.955441, -0.370864)
TriggerDecision:
```

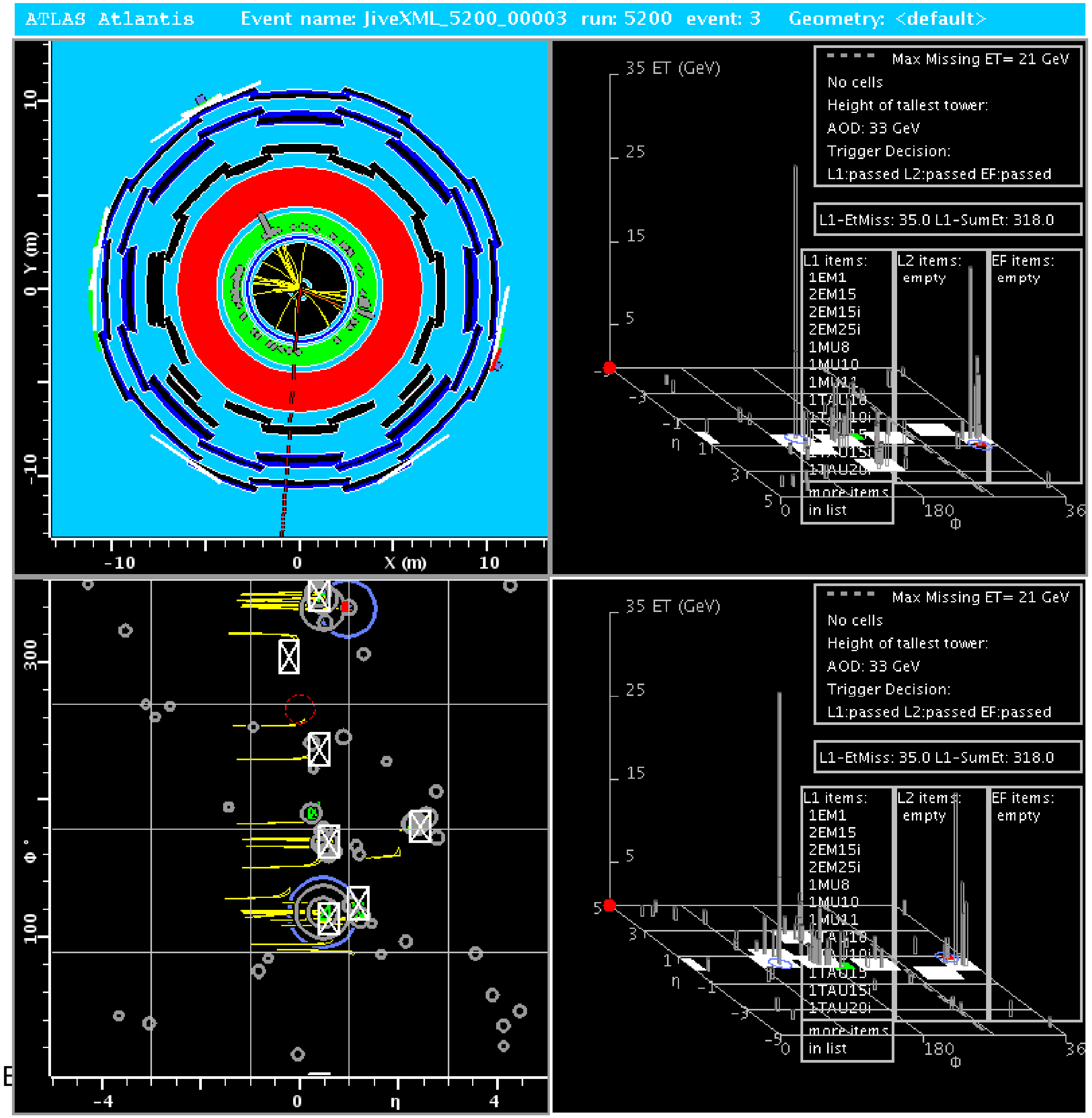


ATLAS Atlantis Event name: JiveXML_5200_00003 run: 5200 event: 3 Geometry: <default>

Max Missing ET= 21 GeV
No cells
Height of tallest tower:
AOD: 33 GeV
Trigger Decision:
L1:passed L2:passed EF:passed

L1-EtMiss: 35.0 L1-SumEt: 318.0

L1 items:
1EM1
2EM15
2EM15i
2EM25i
1MU8
1MU10
1MU11
1TAU10
1TAU10i
1TAU15
1TAU15i
1TAU20i
1TAU25i
L2 items:
empty
EF items:
empty


ATLAS Atlantis Event name: JiveXML_5200_00003 run: 5200 event: 3 Geometry: <default>

Max Missing ET= 21 GeV
No cells
Height of tallest tower:
AOD: 33 GeV
Trigger Decision:
L1:passed L2:passed EF:passed

L1-EtMiss: 35.0 L1-SumEt: 318.0

L1 items:	L2 items:	EF items:
1EM1	empty	empty
2EM15		
2EM15i		
2EM25i		
1MU8		
1MU10		
1MU11		
1TAU10		
1TAU10i		
1TAU15		
1TAU15i		
1TAU20i		
more items in list		

Max Missing ET= 21 GeV
No cells
Height of tallest tower:
AOD: 33 GeV
Trigger Decision:
L1:passed L2:passed EF:passed

L1-EtMiss: 35.0 L1-SumEt: 318.0

L1 items:	L2 items:	EF items:
1EM1	empty	empty
2EM15		
2EM15i		
2EM25i		
1MU8		
1MU10		
1MU11		
1TAU10		
1TAU10i		
1TAU15		
1TAU15i		
1TAU20i		
more items in list		

Release 13 status for TrigDecisionTool

- Some known problems with TrigDecisionTool and related code
  - List them… or too technical, ref twiki?
  - (configured chains is actually “started”; crash when trying to get L2 features (fix in head); …)
- It works well enough to try it.
- Feedback is very useful at this stage.

Conclusions

- TrigDecisionTool, and the steering, configuration and navigation code behind it, provide a major step up in functionality for analysis of the trigger performance.
- Now would be a great time to try it out and provide feedback.