My diploma thesis (current &prospective progress)

By Gordon Fischer

Hamburg 25.08.2006

Schedule

1. Overview: Trigger Steering

2. My current work

3. Future steps

important items

-*Trigger Element (TE):* activated by a specific algorithm(e.g.FEX-Algo) → "hypothesis" that has to be confirmed or rejected → e.g. ‚e25i' is a electron with 25 GeV transverse-energy and isolated

-*Signature:* a combination of requested TE using AND, OR & NOT (2e25i means e25i AND e25i)

-*Sequences:* describes the way TE transferred to another TE using algos.

more important items

-signature table: a list of signatures

-sequence table: a list of sequences

-Trigger Chain: one signature per step, definition of all TC is given by the configuration

Signatures

Signatures are a combination of active TEs that identify the *signature* of a desired physics process.

e50i + e50i

-Signatures are used by the Steering Decision at every Trigger step to select wanted and promptly reject unwanted events

-They are checked against the number of existing active TEs to determine if they are *Satisfied*. The ones that are fulfilled are recorded to be used later by the Steering Result

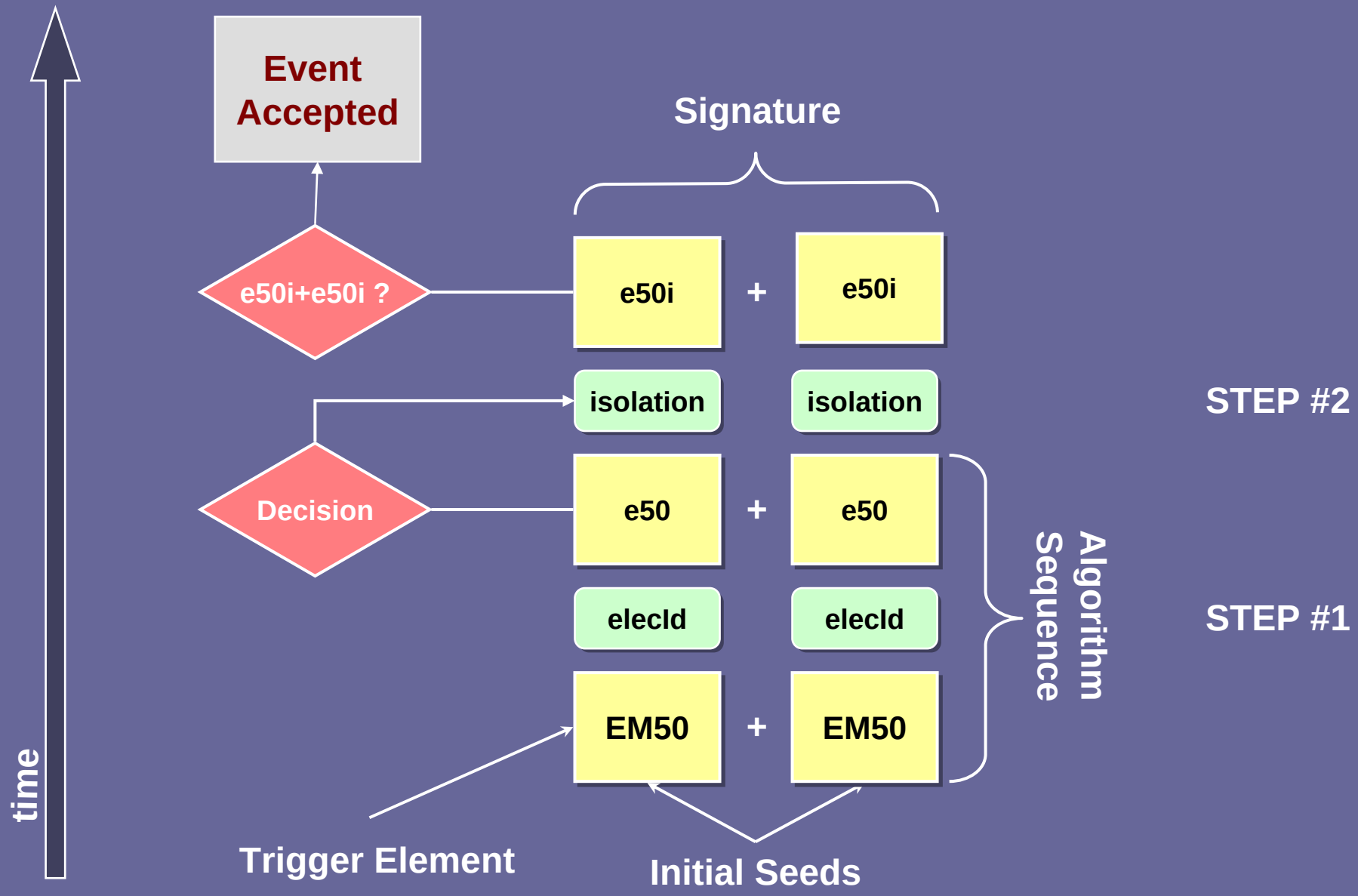
-The Signature Table in a run is defined in configuration files and identify the Physics signals that will be searched

1.Trigger Steering

- The Steering is one building block of the HLTSSW → AKA the „core software“
- Steering takes ownership of the event and unpacks the LvL1 Result recreating the RoI → for each RoI a TE is created
- TE →RoI via data navigation link „uses“
- At the LvL2 the TE passes the selection →externalized and included in the LvL2 Result

  → sent to one of the EF Farms nodes

1.Trigger Steering



My current work: What i have to do

-Assembling information about the rates
 or theTiming Information

-Evaluate dead time

-In my thesis i have to contrive monitoring
 histograms that controls the Lvl2 data flow

-e.g.:

  i) count Events per Step

  ii)count Events per Chains (that means how
    goes a certain event through the LvL2)

current work (cont.)

-we have installed the AtlasTrigger Software

-acquireing the Trigger Steering Software

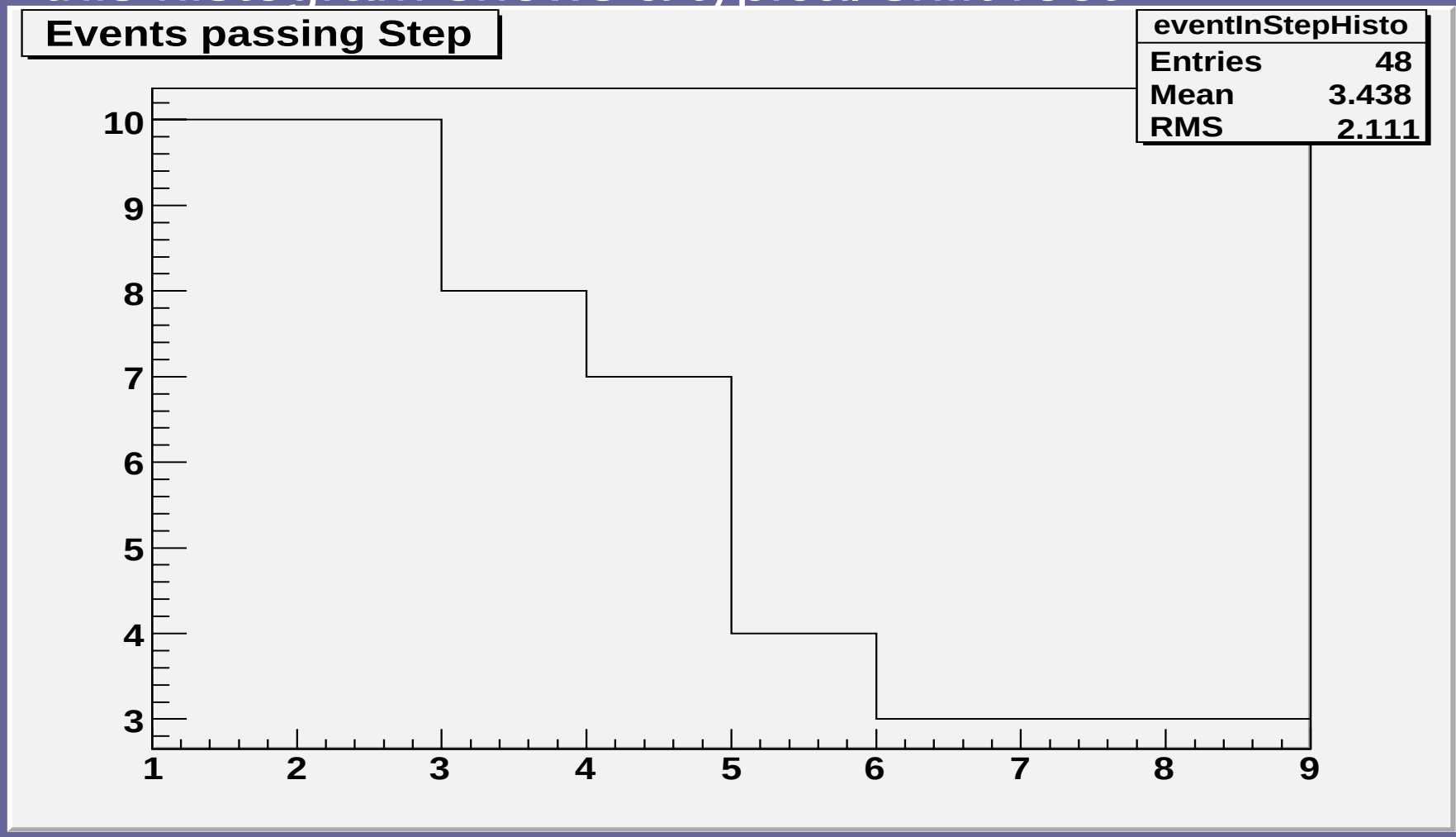
-using different packages like:

- ~TriggerRelease
- ~Trigger/TrigSteer/TrigSteering
- ~Trigger/TrigSteer/TrigInterfaces
- ~Trigger/TrigMonitoring/TrigSteerMonitoring

My current work

-producing histograms like„events passing step“

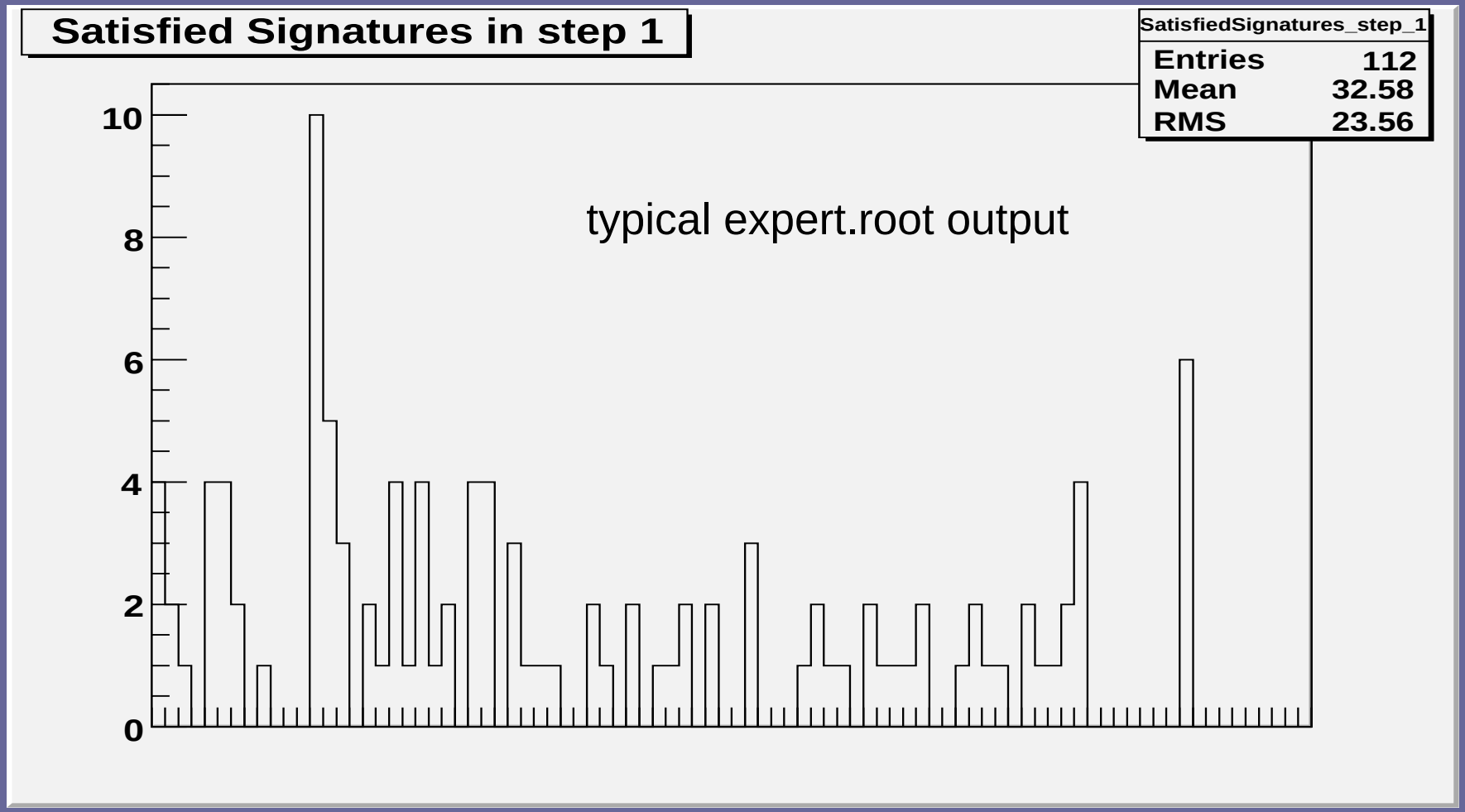
-this histogram shows a typical shift.root



My current work

-TrigSteering contains the StepDecision.cxx that has to be changed

-Using root (what else?)



Future steps

-now: change of strategy!

-stepdecision.cxx will be discarted

-adopting new programs like TrigSteer.cxx and Chain.cxx

-adopting a new approach with 2d-Histograms(TriggerChains vs. Steps or TriggerChains vs. TE)

-therefore counting the TE in every chain for every step

Future Steps

-At present discussion with experts at CERN (Tomasz Bold, Till Eifert and W.Wiedenmann)

-How and where can we integrate the monitoring histograms in the code:

-to be favourite: TrigSteer.cxx or Chain.cxx

-the following 3 slides shows a part of a output created by pureNewSteering_jobOption.py

-the executing software is athena

- -- HLTChainList printout ------------------------------
- -- HLTChain printout ----------------------------------
- -- chain_id: **em25i_L2** | chain_counter: 1 | level: L2 | m_lower_chain_id: **EM25i** prescale: 1 | forced_accept: 0
- -- HLTSignature printout ------------------------------
- -- label: **em25i_L2_1** | signature_counter: 1 | logic: 1
- -- outputTE(s) : (40925, **em25i**)
- --
- -- HLTSignature printout ------------------------------
- -- label: **em25i_L2_2** | signature_counter: 2 | logic: 1
--- outputTE(s) : (62177, **em25i‘**)

- -- HLTSequence printout --------------------------------
- -- inputTEs   : (42914, **EM25i**)
- -- algorithms : PESA::dummyAlgo/Egamma/L2
- -- outputTE   : (40925, **em25i**)
- --
- -- HLTSequence printout --------------------------------
- -- inputTEs   : (40925, **em25i**)
- -- algorithms : PESA::dummyAlgo/EgammaAdv/L2
- -- outputTE   : (62177, **em25i‘**)

///////// Start of HLT Processing in L2 /////////
 //
LvlConverterAlg_L2 DEBUG activating all configured chains.
LvlConverterAlg_L2 DEBUG initial Navigation node created.
LvlConverterAlg_L2 DEBUG Simulated Lvl1 event: EM15i,**EM25i**;EM15i,EM25i;MU6,MU20;MU6;MU6;J50
LvlConverterAlg_L2 DEBUG creating RoINode with attached RoIDescriptor with id: 0
LvlConverterAlg_L2 DEBUG creating TriggerElement EM15i with id: 32975
LvlConverterAlg_L2 DEBUG creating TriggerElement **EM25i** with id: 42914
LvlConverterAlg_L2 DEBUG creating RoINode with attached RoIDescriptor with id: 1
LvlConverterAlg_L2 DEBUG creating TriggerElement EM15i with id: 32975
LvlConverterAlg_L2 DEBUG creating TriggerElement **EM25i** with id: 42914
LvlConverterAlg_L2 DEBUG creating RoINode with attached RoIDescriptor with id: 2
LvlConverterAlg_L2 DEBUG creating TriggerElement MU6 with id: 7814
LvlConverterAlg_L2 DEBUG creating TriggerElement MU20 with id: 65524
LvlConverterAlg_L2 DEBUG creating RoINode with attached RoIDescriptor with id: 3

```
/////////  Executing next Step  /////////
 ////////////////////////////////////////
TrigSteer_L2       DEBUG Executing chain #1 (id em25i_L2)  step 0
TrigSteer_L2       DEBUG Signature em25i_L2_1: executing
TrigSteer_L2       DEBUG Sequence: executing
Egamma_L2_L2           INFO Executing this dummyAlgo Egamma_L2_L2 for types 42914 -> 40925
Egamma_L2_L2           INFO Executing this dummyAlgo Egamma_L2_L2 for types 42914 -> 40925
TrigSteer_L2       DEBUG Signature: found 2 active TEs, require 1 -> satisfied!
TrigSteer_L2       DEBUG Signature: done, satisfied
TrigSteer_L2       DEBUG Executing chain #2 (id mu20_L2)  step 0
TrigSteer_L2       DEBUG Signature mu20_L2_1: executing
TrigSteer_L2       DEBUG Sequence: executing
Muon_L2_L2           INFO Executing this dummyAlgo Muon_L2_L2 for types 65524 -> 1091
TrigSteer_L2       DEBUG Signature: found 1 active TEs, require 1 -> satisfied!
TrigSteer_L2       DEBUG Signature: done, satisfied
TrigSteer_L2       DEBUG Executing chain #3 (id 2em15i_L2)  step 0
TrigSteer_L2       DEBUG Signature 2em15i_L2_1: executing
TrigSteer_L2       DEBUG Sequence: executing
TrigSteer_L2       DEBUG Signature: found 2 active TEs, require 2 -> satisfied!
TrigSteer_L2       DEBUG Signature: done, satisfied
TrigSteer_L2       DEBUG Executing chain #4 (id 2mu6_L2)  step 0
TrigSteer_L2       DEBUG Signature 2mu6_L2_1: executing
TrigSteer_L2       DEBUG Sequence: executing
TrigSteer_L2       DEBUG Signature: found 3 active TEs, require 2 -> satisfied!
TrigSteer_L2       DEBUG Signature: done, satisfied
TrigSteer_L2       DEBUG Executing chain #5 (id j200_L2)  step 0
TrigSteer_L2       DEBUG Signature j200_L2_1: executing
TrigSteer_L2       DEBUG Sequence: executing
TrigSteer_L2       DEBUG Signature: found 0 active TEs, require 1 -> failed!
TrigSteer_L2       DEBUG Chain j200_L2 failed
.
.
.
.
TrigSteer_L2       DEBUG Executing chain #9 (id Z_L2)  step 0
TrigSteer_L2       DEBUG Signature Z_L2_1: executing
TrigSteer_L2       DEBUG Signature: found 2 active TEs, require 2 -> satisfied!
TrigSteer_L2       DEBUG Signature: done, satisfied
TrigSteer_L2       DEBUG Executing chain #10 (id H_L2)  step 0
TrigSteer_L2       DEBUG Signature H_L2_1: executing
TrigSteer_L2       DEBUG Signature: found 2 active TEs, require 4 -> failed!
TrigSteer_L2       DEBUG Chain H_L2 failed
TrigSteer_L2       DEBUG
```

The Last Slide → Coffee Break

```
/////////  Executing next Step  /////////
 ////////////////////////////////////////
TrigSteer_L2       DEBUG Executing chain #9 (id Z_L2)  step 2
TrigSteer_L2       DEBUG Signature Z_L2_3: executing
TrigSteer_L2       DEBUG Sequence: executing
ZFinder_L2_L2         INFO Executing this newDummyAlgo2To1 ZFinder_L2_L2
ZFinder_L2_L2         INFO Executing this newDummyAlgo2To1 ZFinder_L2_L2 for types
40925 -> 23634
ZFinder_L2_L2         INFO Executing this newDummyAlgo2To1 ZFinder_L2_L2 for types
40925 -> 23634
TrigSteer_L2       DEBUG Signature: found 1 active TEs, require 1 -> satisfied!
TrigSteer_L2       DEBUG Signature: done, satisfied
TrigSteer_L2       DEBUG Chain Z_L2 finished successfully
ResultBuilder_L2    DEBUG Event passed.
ResultBuilder_L2    DEBUG
```

Appendix1 -Algorithms

FEX (Feature Extraction Algorithms):

Extracts results → create TE

Cannot provide the Steering with any decision result

HYPO:

validate physical hypotheses based on the past features

results used by the steering to drive the HLT dataflow

CORE:

Monitoring the operations performed by the steering

Appendix2-Algorithms

Algorithms are responsible for the Reconstruction. The Steering calls them on a conditional basis

- The set of algorithms that can potentially be executed in a run is configured based on XML files
- Dynamical information in terms of existing Trigger Elements determine if and how many times a particular algorithm has to be executed
- Algorithms signal success condition to the Steering by activating the input Trigger Element