CMS Tutorial

Data/Trig,Objests,Cfg/Crab, PAT & others

Xiangwei Meng

Institute of High Energy Physics, CAS, Beijing

Nov. 07 , 2008

Contents

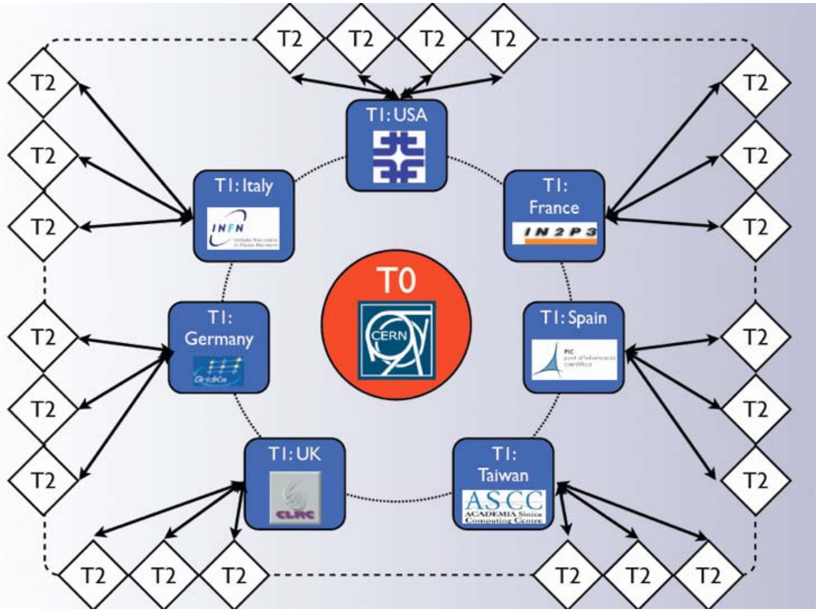
CMS Tutorials

The CMS Offline WorkBook
The CMS Offline SW Guide
CMSSW Reference Manual
CMS TWiki

- Computing model & data flow
- Trig menu
- Physics Objects
- User anal @T2
- Cfg/PYTHON & DBS/Crab
- CMSSW & PAT
- Fast simu production
- Others

The CMS computing model

- **Tier 0 (at CERN)**
  - Receiving data from online system and splitting into *Primary Datasets*
  - Prompt calibration and reconstruction
  - Distribution of RECO and AOD to Tier 1
- **Tier 1 (7 computing centres)**
  - Providing storage for received data
  - Re-Reconstruction, Calibration
  - Central *skimming*
- **Tier 2 (~40 smaller centres)**
  - Service for local community
  - CMS wide analysis resources: User analysis
  - MC production & simulation



Prompt data

- **CMS detector -UXC at LHC Point 5 (P5), Cessy, France**
- **the trigger and data acquisition system P5 underground CR –UCX and P5 surface CR –SCX**

- **T0 -CMS Computing Center, Meyrin, CH -for prompt reconstruction**

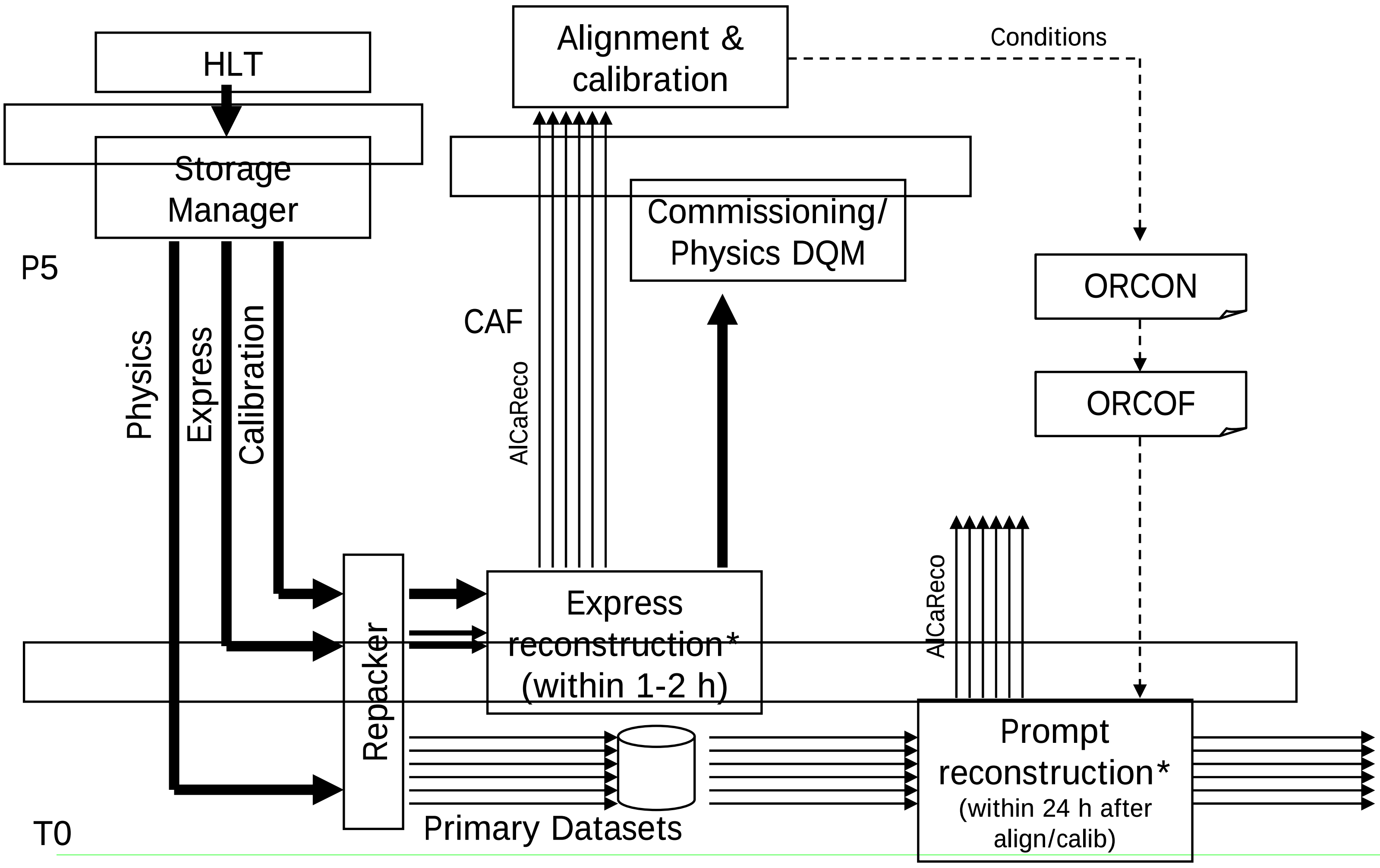
PromptReconstruction
Certain datasets are configured to get reconstructed
– BeamHalo, ZeroBias, TriggerTest, Cosmics, BarrelMuon, EndcapsMuon

– Primary Dataset (PD)
– Processing Era (PE): defined ad-hoc currently
– Processing Info (PI): information on processing
– Processing Version (PV): usually starts as -v1
– Data Tier (DT): RAW, RECO, ALCARECO, (AOD)

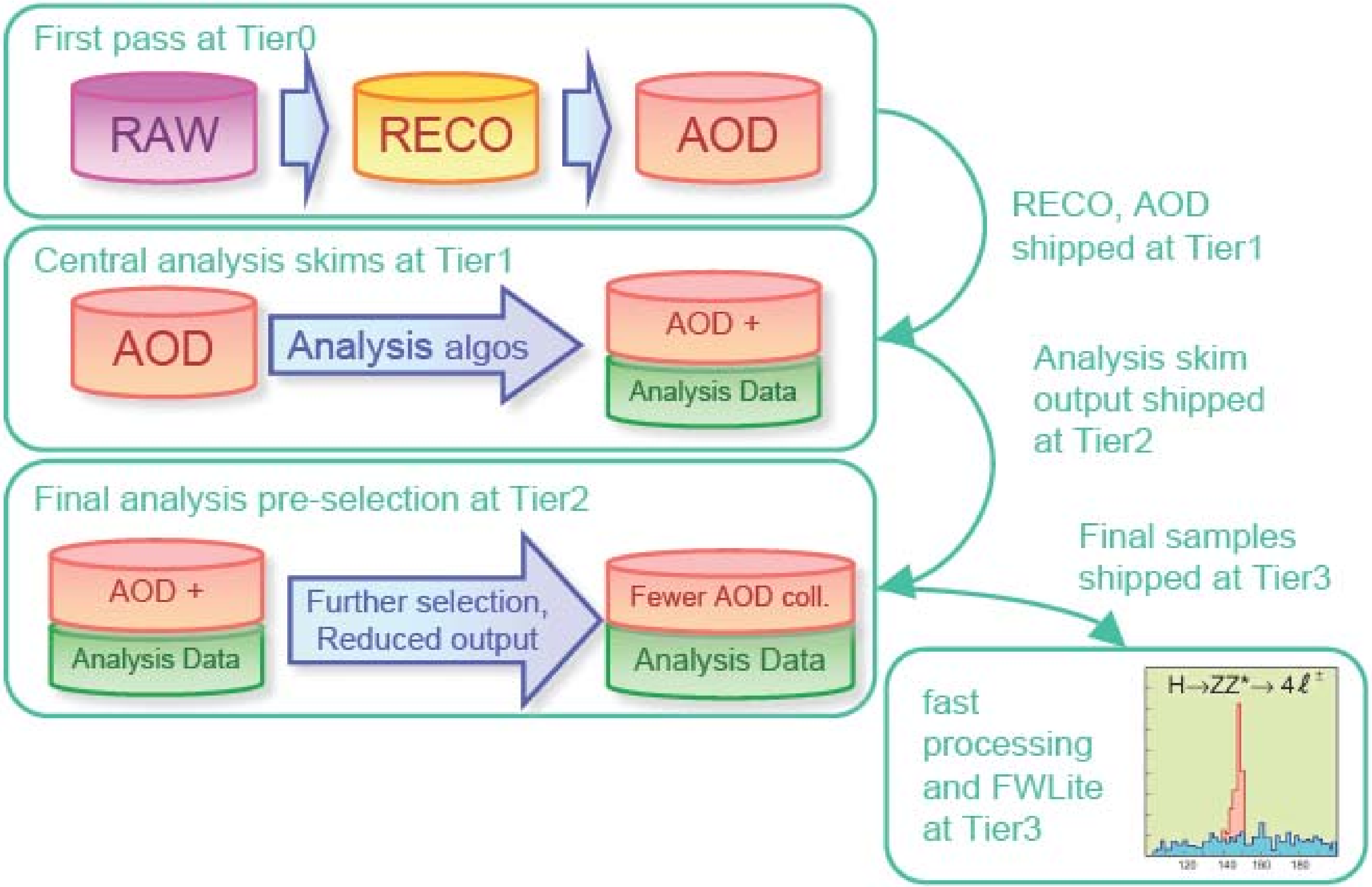
Dataset Names
/<PD>/<ProcEra><PI><ProcVer>/<DataTier>
/BeamHalo/BeamCommissioning08_CRUZET4_V5P_StreamALCARECOTkAlCosmics_v3/ALCARECO

Logical File Names
/store/data/<PE>/<PD>/<DT>/<PI><PV>/<SPLIT>/<UUID>.root
/store/data/BeamCommissioning08/BeamHalo/RECO/v1/000/062/236/88034D41-7780-DD11-A3F6-000423D6A6F4.root

CMS Prompt Calibration Loop



Data Skimming



“HLT exercise”: 1E32 (June 2007)

HLT path	L1 condition	Thresholds (GeV)	HLT Rate (Hz)	Total Rate (Hz)
VBF Double-Jet + $\not\!\!E_T$	A_ETM30	(40, 60)	0.2 ± 0.0	89.0
SUSY 2-jet+$\not\!\!E_T$	A_ETM30	(80,20,60)	2.0 ± 0.1	90.4
Acopl. Double-Jet + $\not\!\!E_T$	A_ETM30	(60, 60)	1.0 ± 0.0	90.4
Single Isolated e	A_SingleIsoEG12	15	17.1 ± 2.3	107.5
Single Relaxed e	A_SingleEG15	17	9.6 ± 1.3	109.3
Double Isolated e	A_DoubleIsoEG8	10	0.2 ± 0.1	109.4
Double Relaxed e	A_DoubleEG10	12	0.8 ± 0.1	109.9
Single Isolated γ	A_SingleIsoEG12	30	8.4 ± 0.7	118.1
Single Relaxed γ	A_SingleEG15	40	2.8 ± 0.2	118.5
Double Isolated γ	A_DoubleIsoEG8	(20,20)	0.6 ± 0.4	119.0
Double Relaxed γ	A_DoubleEG10	(20,20)	1.8 ± 0.5	120.1
High E_T e	A_SingleEG15	80	0.5 ± 0.0	120.4
High E_T e	A_SingleEG15	200	0.1 ± 0.0	120.4
Lifetime b-tag 1-jet	$\diamond$	180	1.3 ± 0.0	120.5
Lifetime b-tag 2-jets	$\diamond$	120	2.1 ± 0.0	121.2
Lifetime b-tag 3-jets	$\diamond$	70	1.7 ± 0.0	121.8
Lifetime b-tag 4-jets	$\diamond$	40	1.8 ± 0.0	122.6
Lifetime b-tag H_T	$\diamond$	470	2.5 ± 0.1	123.1
Single τ	A_SingleTauJet80	15	0.2 ± 0.0	123.2
$\tau + \not\!\!E_T$	A_TauJet30_ETM30	15	1.8 ± 0.2	124.7
Double τ (Calo+Pixel)	A_DoubleTauJet40	15	4.9 ± 0.6	129.4
$e + b$-jet	A_IsoEG10_Jet20	(10, 35)	0.1 ± 0.0	129.4
e + jet	A_IsoEG10_Jet30	(12, 40)	11.6 ± 1.2	135.8
$e + \tau$	A_IsoEG10_TauJet20	(12, 20)	0.2 ± 0.0	135.8
Prescaled e/γ	See Table 3.9		5.0 ± 0.0	140.8
Prescaled μ	See Table 2.4		3.0 ± 0.0	143.8
Min.Bias	A_MinBias_HTT10	—	1.5 ± 0.0	145.3
Pixel Min.Bias	A_ZeroBias	—	1.5 ± 0.0	146.8
Zero Bias	A_ZeroBias	—	1.0 ± 0.0	147.8
Total HLT rate (Hz)				148 ± 4.9

HLT path	L1 condition	Thresholds (GeV)	HLT Rate (Hz)	Total Rate (Hz)
	A_DoubleJet70			
Acopl. Single-Jet + $\not\!\!E_T$	A_ETM30	(100, 60)	1.6 ± 0.0	84.2
Single-Jet + $\not\!\!E_T$	A_ETM30	(180, 60)	2.2 ± 0.1	84.4
Double-Jet + $\not\!\!E_T$	A_ETM30	(125, 60)	1.0 ± 0.0	84.4
Triple-Jet + $\not\!\!E_T$	A_ETM30	(60, 60)	0.6 ± 0.0	84.4
Quad-Jet + $\not\!\!E_T$	A_ETM30	(35, 60)	1.2 ± 0.1	84.6
$H_T + \not\!\!E_T$	A_HTT300	(350, 65)	4.4 ± 0.1	86.2
Single Jet Prescale 10	A_SingleJet100	150	3.5 ± 0.0	87.9
Single Jet Prescale 100	A_SingleJet70	110	1.5 ± 0.0	89.1
Single Jet Prescale 1000	A_SingleJet30	60	0.8 ± 0.4	89.9

Continued on next page . . .

- μ : 50 Hz
- e ɣ : 30 Hz
- jets/MET/Ht: 30 Hz
- τ : 7 Hz
- b-jets: 10 Hz
- x-channels: 20 Hz
- prescaled: 15 Hz
- Total: 150 Hz

~ 60 HLT paths

“iCSA08”: 2E30/2E31 (April 2008)

Status	Path Name	L1 condtition	Threshold [GeV]	L1 Prescale	HLT Prescale	HLT Rate [Hz]	Total Rate [Hz]
new	HLT2PhotonLWonlyPMEt8_L1R_NI	*L1_DoubleEG5*	8	1	1	0.0 ± 0.0	117.1
new	HLT2PhotonLWonlyPMEt10_L1R_NI	*L1_DoubleEG5*	10	1	1	0.0 ± 0.0	117.1
new	HLT2PhotonLWonlyPMEt12_L1R_NI	*L1_DoubleEG10*	12	1	1	0.0 ± 0.0	117.1
1e32	HLT1Photon	*L1_SingleIsoEG12*	30	1	1	0.3 ± 0.2	117.1
1e32	HLT1PhotonRelaxed	*L1_SingleEG15*	40	1	1	0.2 ± 0.2	117.1
1e32	HLT2Photon	*L1_DoubleIsoEG8*	(20,20)	1	1	0.0 ± 0.0	117.1
1e32	HLT2PhotonRelaxed	*L1_DoubleEG10*	(20,20)	1	1	0.0 ± 0.0	117.1
1e32	HLT1EMHighEt	*L1_SingleEG15*	80	1	1	0.0 ± 0.0	117.1
1e32	HLT1EMVeryHighEt	*L1_SingleEG15*	200	1	1	0.0 ± 0.0	117.1
1e32	HLT2ElectronZCounter	*L1_DoubleIsoEG8*	(10,10)	1	1	0.0 ± 0.0	117.1
1e32	HLT2ElectronExclusive	*L1_ExclusiveDoubleIsoEG6*	(6,6)	1	1	0.0 ± 0.0	117.2
1e32	HLT2PhotonExclusive	*L1_ExclusiveDoubleIsoEG6*	(10,10)	1	1	0.0 ± 0.0	117.2
1e32	HLT1PhotonL1Isolated	*L1_SingleIsoEG10*	12	1	1	0.0 ± 0.0	117.2
1e32	HLTB1Jet	◊	180	1	1	0.4 ± 0.2	117.2
1e32	HLTB2Jet	◊	120	1	1	0.8 ± 0.2	117.2
1e32	HLTB3Jet	◊	70	1	1	0.5 ± 0.2	117.2
1e32	HLTB4Jet	◊	40	1	1	0.8 ± 0.2	117.2
1e32	HLTBHT		470	1	1	1.2 ± 0.3	117.3
new	HLTB1Jet120		120	1	1	0.0 ± 0.0	117.3
new	HLTB2Jet60		60	1	1	0.0 ± 0.0	117.3
new	HLTB3Jet40		40	1	1	0.0 ± 0.0	117.3
new	HLTB4Jet30	◊	30	1	1	0.0 ± 0.0	117.3
new	HLTBHT320	◊	320	1	1	0.0 ± 0.0	117.3
1e32	HLT1Tau	*L1_SingleTauJet80*	15	1	1	0.0 ± 0.0	117.3
1e32	HLT1Tau1MET	*L1_TauJet30_ETM30*	15	1	1	0.1 ± 0.0	117.3
1e32	HLT2TauPixel	*L1_TauJet40*	15	1	1	0.4 ± 0.2	117.7
new	HLT1TauRelaxed	*L1_SingleTauJet80*	15	1	1	0.0 ± 0.0	117.7
new	HLT1Tau1METRelaxed	*L1_TauJet30_ETM30*	15	1	1	0.0 ± 0.0	117.7
new	HLT2TauPixelRelaxed	*L1_DoubleTau20*	15	1	1	0.0 ± 0.0	117.7
1e32	HLTXElectronBJet	*L1_IsoEG10_Jet20*	(10,35)	1	1	0.1 ± 0.0	117.7
1e32	HLTXElectron1Jet	*L1_IsoEG10_Jet30*	(12,40)	1	1	1.0 ± 0.2	117.8
1e32	HLTXElectron2Jet	*L1_IsoEG10_Jet30*	(12,80)	1	1	0.0 ± 0.0	117.8
1e32	HLTXElectron3Jet	*L1_IsoEG10_Jet30*	(12,60)	1	1	0.0 ± 0.0	117.8
1e32	HLTXElectron4Jet	*L1_IsoEG10_Jet30*	(12,35)	1	1	0.0 ± 0.0	117.8
1e32	HLTXElectronTau	*L1_IsoEG10_TauJet20*	(12,20)	1	1	0.8 ± 0.3	118.0
1e32	HLTMinBias	*L1_MinBias_HTT10*	-	1	1	0.5 ± 0.0	118.5
1e32	HLTMinBiasPixel	*L1_ZeroBias*	-	1	1	1.5 ± 0.0	120.0
new	HLTMinBiasHcal	◊◊	-	12	2500	2.0 ± 0.0	122.0
new	HLTMinBiasEcal	$L1_SingleEG2, L1_DoubleEG1$	-	20	1500	2.0 ± 0.0	124.0
1e32	HLTZeroBias	*L1_ZeroBias*	-	1	1	2.5 ± 0.0	124.5
		Total HLT rate (Hz)					124.5 ± 3.9

~ 190 HLT paths

Startup menus: 8E29 (Fall 2008)

Path Name	L1 condtition	L1 Prescale	HLT Prescale	HLT Rate [Hz]	Total Rate [Hz]	Avg. Size [MB]	Total Throughput [MB/s]
HLT_Mu13	*L1_SingleMu10*	1	1	0.12 ± 0.03	89.48	1.50	134.22
HLT_Mu15	*L1_SingleMu10*	1	1	0.06 ± 0.02	89.48	1.50	134.22
HLT_Mu15_Vtx2mm	*L1_SingleMu7*	1	1	0.06 ± 0.02	89.48	1.50	134.22
HLT_DoubleIsoMu3	*L1_DoubleMu3*	1	1	0.04 ± 0.02	89.48	1.50	134.22

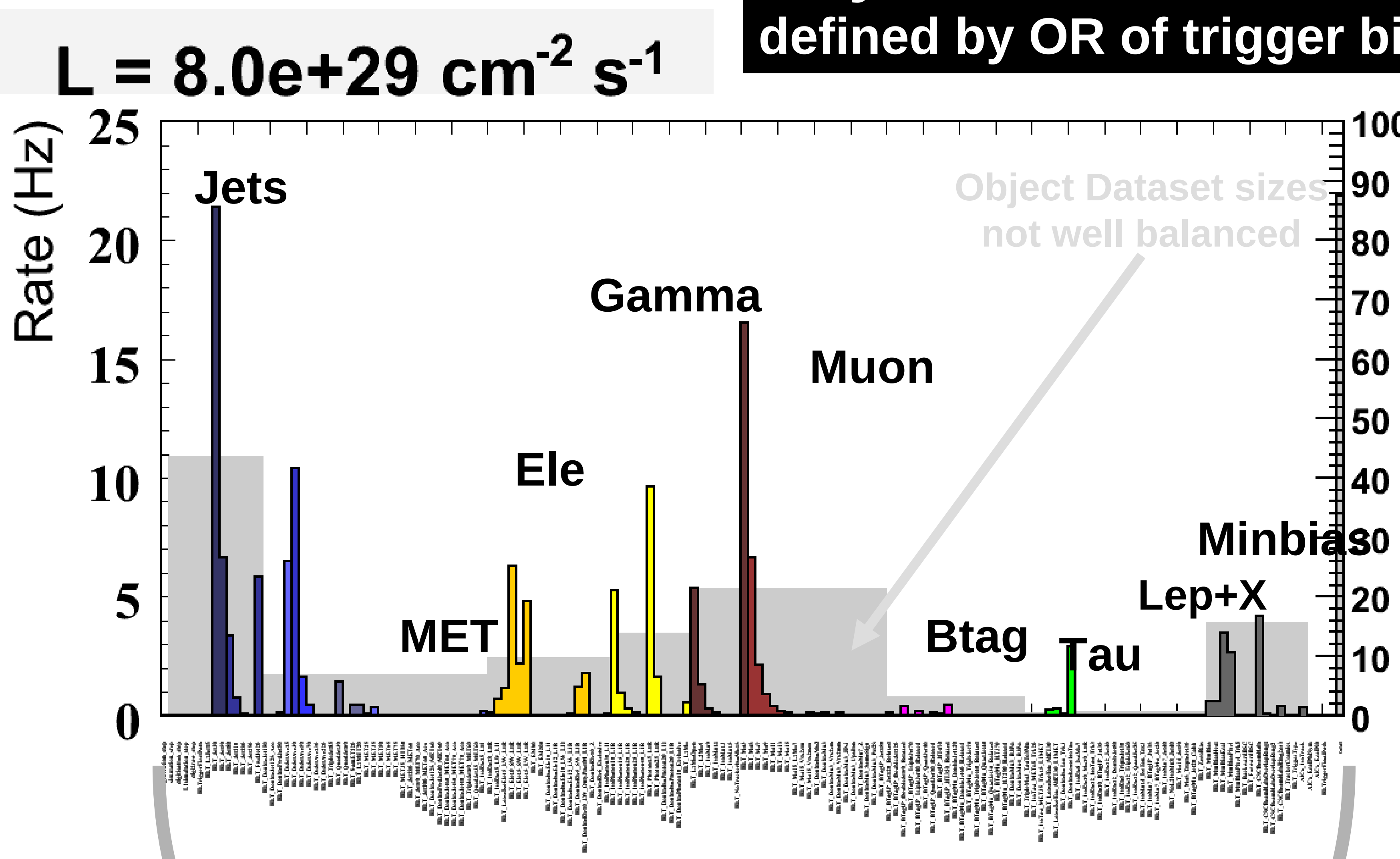
–continued from previous page (L = 8.0e+29)

Path Name	L1 condtition	L1 Prescale	HLT Prescale	HLT Rate [Hz]	Total Rate [Hz]	Avg. Size [MB]	Total Throughput [MB/s]
HLT_IsoEle12_DoubleJet80	*L1_IsoEG10_Jet30*	1	1	0.00 ± 0.00	89.67	1.50	134.50
HLT_IsoElec5_TripleJet30	*L1_EG5_TripleJet15*	1	1	0.07 ± 0.03	89.67	1.50	134.50
HLT_IsoEle12_TripleJet60	*L1_IsoEG10_Jet30*	1	1	0.02 ± 0.02	89.67	1.50	134.50
HLT_IsoEle12_QuadJet35	*L1_IsoEG10_Jet30*	1	1	0.02 ± 0.02	89.67	1.50	134.50
HLT_IsoMu14_IsoTau_Trk3	*L1_Mu5_TauJet20*	1	1	0.00 ± 0.00	89.67	1.50	134.50
HLT_IsoMu7_BTagIP_Jet35	*L1_Mu5_Jet15*	1	1	0.00 ± 0.00	89.67	1.50	134.50
HLT_IsoMu7_BTagMu_Jet20	*L1_Mu5_Jet15*	1	1	0.00 ± 0.00	89.67	1.50	134.50
HLT_IsoMu7_Jet40	*L1_Mu5_Jet15*					1.50	134.50
HLT_NoL2IsoMu8_Jet40	*L1_Mu5_Jet15*					1.50	134.50
HLT_Mu14_Jet50	*L1_Mu5_Jet15*					1.50	134.50
HLT_Mu5_TripleJet30	*L1_Mu3_TripleJet15*					1.50	134.50
HLT_BTagMu_Jet20_Calib	*L1_Mu5_Jet15*	1	1	0.19 ± 0.05	89.67	1.50	134.50
HLT_ZeroBias	*L1_ZeroBias*	15000	1	4.03 ± 0.25	93.68	1.50	140.52
HLT_MinBias	*L1_MinBias_HTT10*	4000	1	1.73 ± 0.16	95.39	1.50	143.08
HLT_MinBiasHcal	*ListTooLong*	5000	1	2.52 ± 0.19	97.90	1.50	146.85
HLT_MinBiasEcal	*L1_SingleEG2ORL1_DoubleEG1*	5000	1	3.75 ± 0.24	101.63	1.50	152.44
HLT_MinBiasPixel	*L1_ZeroBias*	15000	1	2.64 ± 0.20	104.25	1.50	156.38
HLT_MinBiasPixel_Trk5	*L1_ZeroBias*	15000	1	0.32 ± 0.07	104.54	1.50	156.81
HLT_BackwardBSC	*38OR39*	1	1	0.00 ± 0.00	104.54	1.50	156.81
HLT_ForwardBSC	*36OR37*	1	1	0.00 ± 0.00	104.54	1.50	156.81
HLT_CSCBeamHalo	*L1_SingleMuBeamHalo*	2	1	4.12 ± 0.24	108.42	1.50	162.64
HLT_CSCBeamHaloOverlapRing1	*L1_SingleMuBeamHalo*	2	1	0.08 ± 0.03	108.47	1.50	162.71
HLT_CSCBeamHaloOverlapRing2	*L1_SingleMuBeamHalo*	2	1	0.00 ± 0.00	108.47	1.50	162.71
HLT_CSCBeamHaloRing2or3	*L1_SingleMuBeamHalo*	2	1	0.40 ± 0.07	108.63	1.50	162.94
HLT_TrackerCosmics	*24OR25OR26OR27OR28*	1	1	0.00 ± 0.00	108.63	1.50	162.94
AlCa_IsoTrack	*ListTooLong*	1	1	6.79 ± 0.32	114.40	0.21	164.18
AlCa_EcalPhiSym	*ListTooLong*	1	1	50177.63 ± 42.49	50211.02	0.00	214.27
AlCa_EcalPi0	*ListTooLong*	1	1	62.60 ± 0.96	50211.02	0.01	214.27
Total Physics HLT rate (Hz), AlCa triggers not included							108.63 ± 1.24
Total Physics HLT throughput (MB/s), AlCa triggers not included							162.94 ± 1.51
Total HLT throughput (MB/s)							214.27 ± 2.03

Path Name	L1 condtition	L1 Prescale	HLT Prescale	HLT Rate [Hz]	Total Rate [Hz]	Avg. Size [MB]	Total Throughput [MB/s]
HLT_IsoEle12_Jet40	*L1_IsoEG10_Jet30*	1	1	0.07 ± 0.03	89.67	1.50	134.50
HLT_Mu9	*L1_SingleMu7*	1	1	0.50 ± 0.06	89.48	1.50	134.22
HLT_Mu11	*L1_SingleMu7*	1	1	0.19 ± 0.03	89.48	1.50	134.22

~ 160 HLT paths

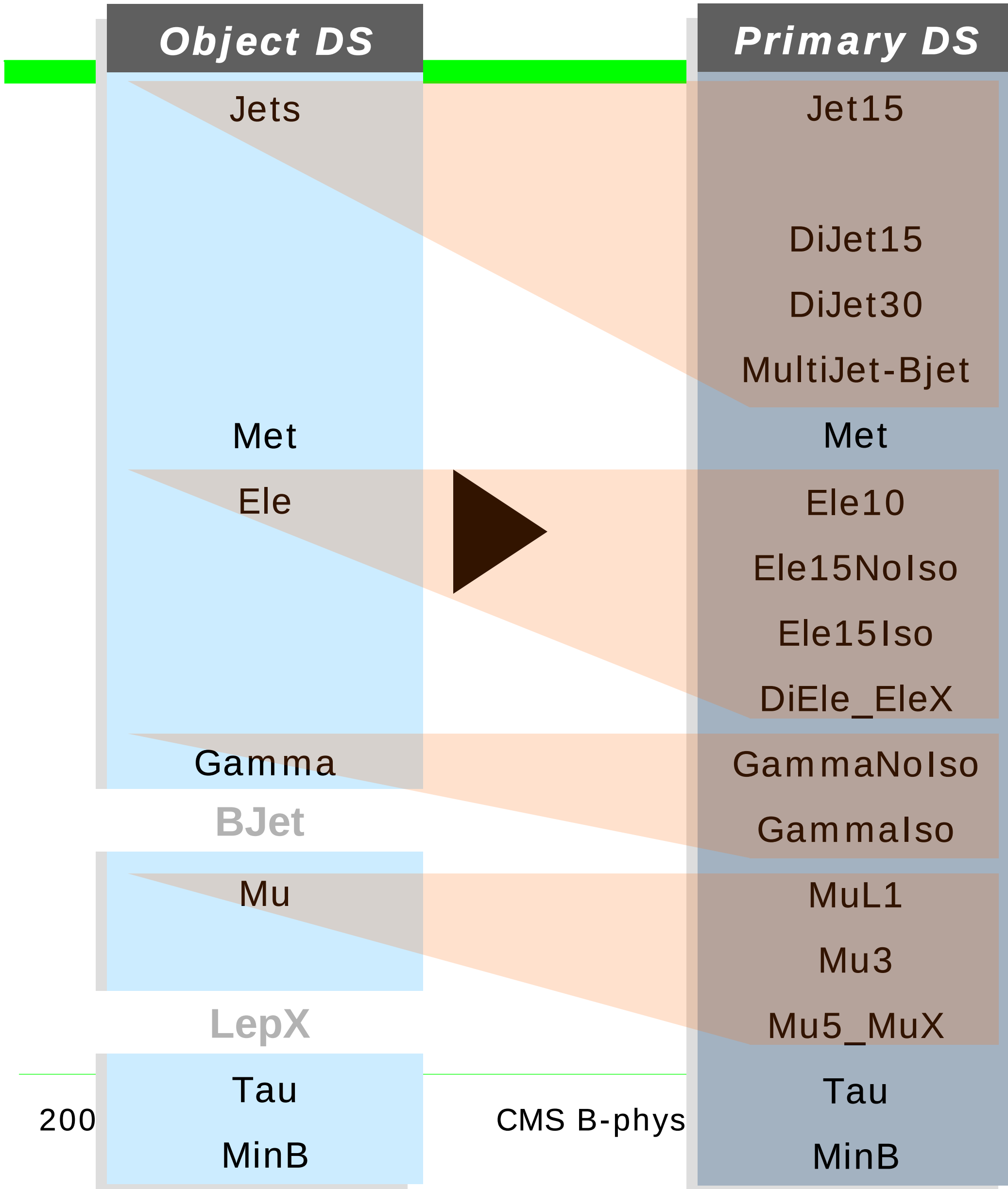
Object Datasets:
defined by OR of trigger bits



A Physics-driven division

(No AlCa trigger considered)

Object → Primary



For rate equalization

Some are too big ⇒ Split

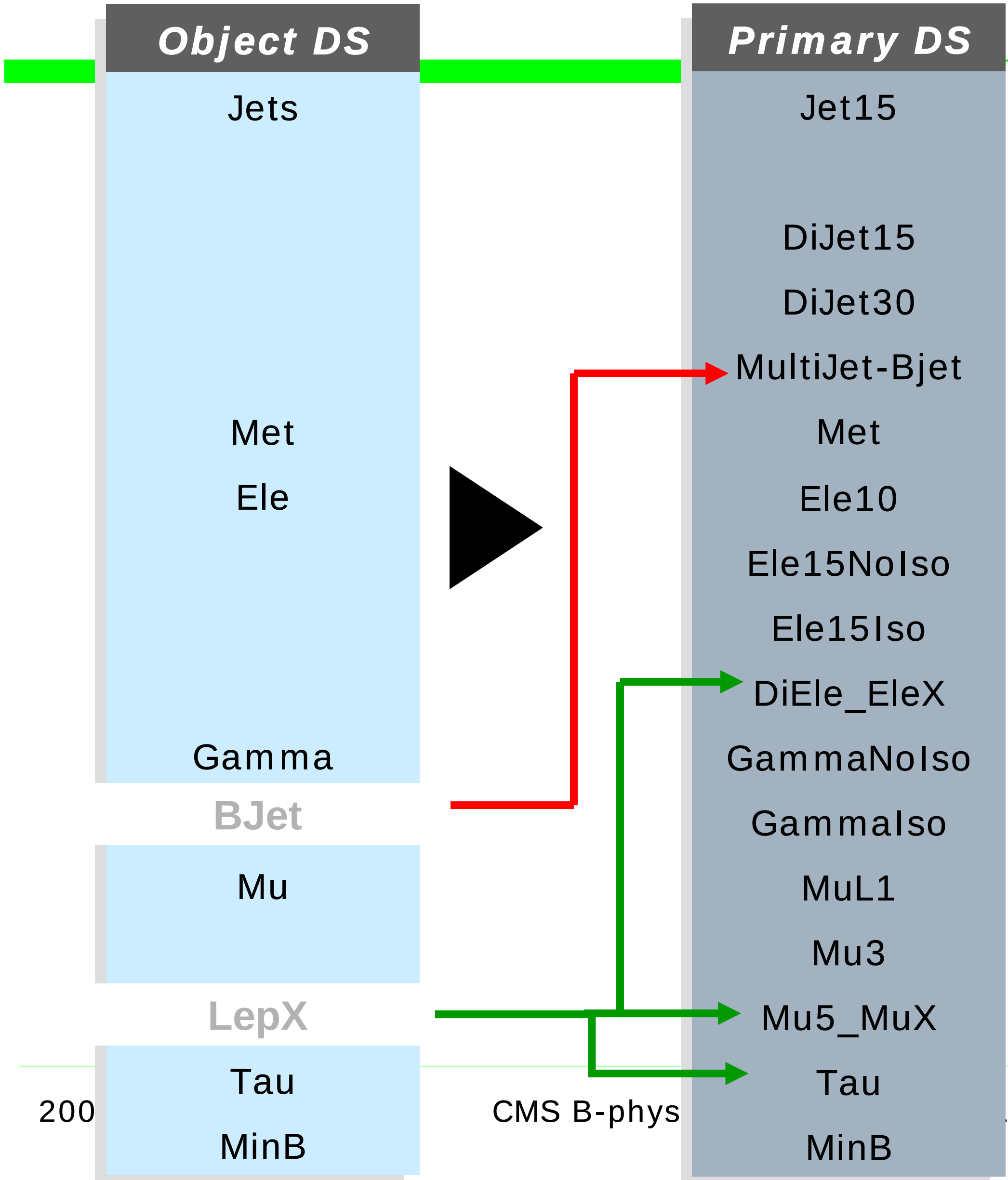
Some are too small ⇒ Merge

From object datasets to primary datasets:

- Splitting based only on trigger bits
- Merge correlated triggers
  - Keep unprescaled triggers together
- Allow for duplication of triggers if meaningful from physics point of view
- Naming provides basic info on content of the PDS

Example here is for
L >> 8E29
For 8E29 only need 8 PD …

Object → Primary



Lepton +X and Bjet datasets have tiny rate.

- Lepton+X: are combined object triggers. Split and absorbed into the 2 relevant lepton datasets
  - same trigger paths appearing in 2 datasets
- Bjet: merged with the MultiJet dataset

Example here is for
L >> 8E29
For 8E29 only need 8 PD …

Startup Core Menu for 8E29

Physics:

- HLT_DoubleEle10_LW_OnlyPixelM_L1R
- HLT_DoubleEle5_SW_L1R
- HLT_DoubleLooseIsoTau
- HLT_Ele10_SW_L1R
- HLT_FwdJet20
- HLT_IsoPhoton10_L1R
- HLT_Jet30 (L1 prscl: 25)
- HLT_Jet50 (HLT prscl: 5)
- HLT_Jet80
- HLT_DiJetAve30
- HLT_L1Jet15 (L1 prscl: 25, HLT prscl: 40)
- HLT_L1MET20 (L1 prscl: 50)
- HLT_L1Mu (HLT prscl: 20)
- HLT_L1MuOpen (HLT prscl: 20)
- HLT_L2Mu9
- HLT_LooseIsoTau_MET30
- HLT_LooseIsoTau_MET30_L1MET
- HLT_MET35
- HLT_Mu3
- HLT_Photon15_L1R

"Zero-/Min-Bias" Triggers

•HLT_ZeroBias (L1 prscl: 15000)
•HLT_MinBias (L1 prscl: 4000)
•HLT_MinBiasHcal (L1 prscl: 5000)
•HLT_MinBiasEcal (L1 prscl: 5000)
•HLT_MinBiasPixel (L1 prscl: 15000)
•HLT_MinBiasPixel_Trk5(L1 prscl: 15000)

AlCaRAW Triggers

•AlCa_EcalPhiSym
•AlCa_EcalPi0
•AlCa_IsoTrack

Technical Trigs. for AlCa

•HLT BackwardBSC
•HLT ForwardBSC
•HLT CSCBeamHalo
•HLT CSCBeamHaloOverlapRing1
•HLT CSCBeamHaloOverlapRing2
•HLT CSCBeamHaloRing2or3
•HLT TrackerCosmics

https://twiki.cern.ch/twiki/bin/view/CMS/WorkBookMuonAnalysis
https://twiki.cern.ch/twiki/bin/view/CMS/SWGuideMuons

reco::Muon

- three different muon reconstruction algorithms are merged in the “muons” collection: GlobalMuons, TrackerMuons, StandaloneMuons
  - The three algorithms are not mutually exclusive - one muon can be in all 3 categories!
  - TrackerMuons are not a subset of GlobalMuons (or vice-versa)

```
bool muon::isStandAloneMuon()
bool muon::isGlobalMuon()
bool muon::isTrackerMuon()
```

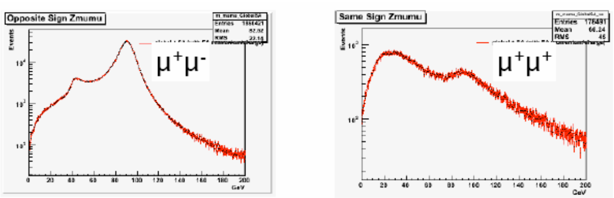
Three possibilities (using p_T as an example)

1. GlobalMuon : muon→pt() returns the p_T from the global refit
2. TrackerMuon and not a GlobalMuon: muon→pt() returns the p_T from the tracker track
3. StandaloneMuon and not a TrackerMuon or GlobalMuon: muon→pt() returns the p_T measured in the muon chambers

To be sure of getting a particular fit, the tracker only, muon system only, and global fit muons can be retrieved explicitly:

```
TrackRef muon::innerTrack()
TrackRef muon::outerTrack()
TrackRef muon::globalTrack()
```

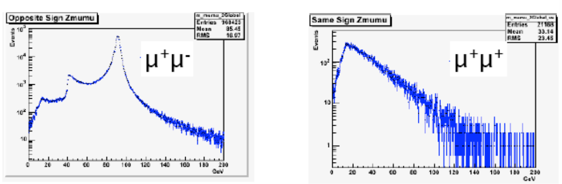
Standalone muons



2_1_7 Z→μμ MC samples
(all plots by A. Fanfani)

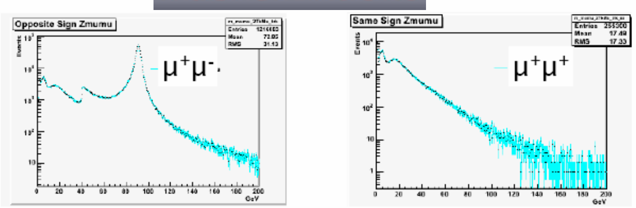
- Muon chamber reconstruction only (no tracker information), requiring at least 2 muon segments

Global muons



- Match a StandaloneMuon to a track in tracker, then perform a global refit using hits from both
- Generally should give the best performance (efficiency/fake-rate/resolution/charge assignment) for muons from ~10GeV to ~1TeV

Tracker muons

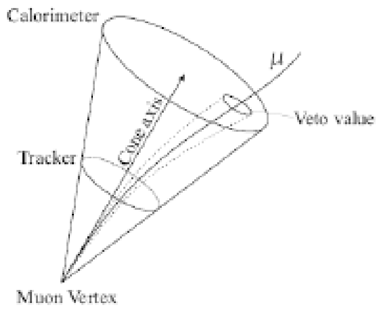


- Match a track to at least 1 segment in the muon chambers - no global refit
- Best efficiency for low p_T muons (J/Ψ→μμ, γγ→μμ, etc.), but also highest fake rate and ambiguities in matching multiple tracks to one muon segment - use only with additional selections/constraints

Muon isolation & selector

- Pre-computed isolations in ΔR cones of 0.3 and 0.5are accessible from the reco::Muon object
- Defined in

DataFormats/MuonReco/interface/Muon.h
DataFormats/MuonReco/interface/MuonIsolation.h
DataFormats/MuonReco/interface/MuonSelectors.h




```
MuonIsolation muon::isolationR03()
MuonIsolation muon::isolationR05()
```

Results of the selector decisions are accessible from the reco::Muon object

```
bool muon::isGood(SelectionType type);
```

For a given isolation cone, several calorimeter and track-based variables are available

- Sum of HCAL E_T
- Sum of ECAL E_T
- Sum of H0 E_T
- Sum of p_T of tracker tracks
- Number of tracker tracks
- Number of jets (sisConeCalo5Jets)

Accessible from the isolation object (using $\Delta R = 0.3$ for example):

```
float muon::isolationR03().hadEt
float muon::isolationR03().emEt
float muon::isolationR03().hoEt
float muon::isolationR03().sumPt
int muon::isolationR03().nTracks
int muon::isolationR03().nJets
```

The selector "type" can be one of 11 different options:

	reco::Muon::All
Alternative way to select Global/ Tracker/Standalone muons	reco::Muon::AllGlobalMuons reco::Muon::AllStandaloneMuons reco::Muon::AllTrackerMuons
Resolves shared segment ambiguities in TrackerMuons	reco::Muon::TrackerMuonArbitrated reco::Muon::AllArbitrated
Cuts on DOCA, X^2, and N(hits)	reco::Muon::GlobalMuonPromptTight
Cuts on maximum penetration depth	reco::Muon::TMLastStationLoose reco::Muon::TMLastStationTight
Cuts on 2-D segment/calorimeter compatibility likelihood	reco::Muon::TM2DCompatibilityLoose reco::Muon:TM2DCompatibilityTight

Muon timing & *In the Calorimeter*

- Muon time-of-flight information is available from the DT's
  - Useful for rejection of non-collision backgrounds, "exotic" heavy charged particle searches, etc.
- Defined in DataFormats/MuonReco/interface/MuonTime.h

MuonTime muon::time()

Several variables are accessible from "time()" for both inward and outward going particles:

float muon::time().inverseBeta

float muon::time().inverseBetaErr

float muon::time().freeInverseBeta

float muon::time().freeInverseBetaErr

float muon::time().timeAtIpInOut

float muon::time().timeAtIpInOutErr

float muon::time().timeAtIpOutIn

float muon::time().timeAtIpOutInErr

- Calorimeter compatibility - use energy deposited in ECAL, HCAL, and HO to check for compatibility with a minimum ionizing particle (MIP)
  - Complementary to information from tracker & muon system
- Defined in RecoMuon/MuonIdentification/interface/MuonCaloCompatibility.h

Result is summarized in a likelihood variable

$$L_{\mu}/(L_{\mu}+L_{not\ \mu})$$

float muon::caloCompatibility()

TeV Muons,CaloMuons &CosmicMuons

- Special treatment is needed for very high p_T muons ($Z' \rightarrow \mu\mu$, etc.)
  - Muon system contributes to the resolution, must deal with showering muons
- Maps stored in RECO/AOD link global muons to refits done with altered hit content
  - Improves resolution & non-Gaussian tails

"refit_name" can be one of three refit algorithms:

1. "default"
2. "first hit"
3. "picky"

- Typical usage:

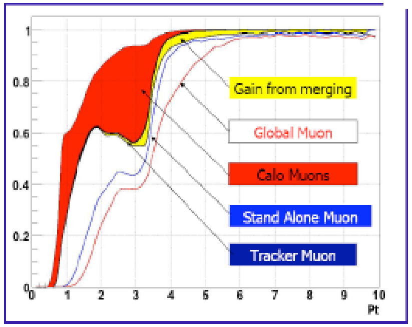
```
edm::Handle<reco::TrackToTrackMap> tevMap;
event.getByLabel("tevMuons","refit_name",tevMap);
```

- New algorithm for CMSSW_2_X - not included in reco::Muon objects
- Tracker track (that was not reconstructed by any other muon algorithm) matched to MIP signature in the calorimeter - no muon system information at all!
- Defined in:

DataFormats/MuonReco/interface/CaloMuon.h

- Typical usage:

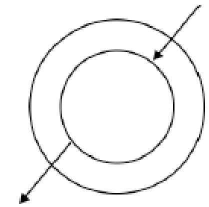
```
edm::Handle<reco::CaloMuonCollection> muons;
event.getByLabel("caloMuons",calomuons);
reco::CaloMuonCollection::const_iterator calomuon;
```



(D. Kovalskyi)

- Extremely high efficiency/low purity for low p_T muons

- Dedicated reconstruction is used for muons not originating from the IP
  - Momentum direction for all muons is set as downward, two legs are found if the muon crosses upper & lower hemispheres



- Can be reconstructed either with tracker + muon system ("globalCosmicMuons" collection) or with muon system only ("cosmicMuons" collection)
- Typical usage:

```
edm::Handle<reco::TrackCollection> muons;
event.getByLabel("cosmicMuons",muons);
reco::TrackCollection::const_iterator muon;
```

For some closely related topics not covered here:
• PAT muons
https://twiki.cern.ch/twiki/bin/view/CMS/EWKPatDefaults21X#Muons
• HLT muons
https://twiki.cern.ch/twiki/bin/view/CMS/MuonHLT

T. Speer
Brown University

Offline vertex reconstruction I

- Vertex Reconstruction:
  - Vertex Finding: Identification of vertices and assignment of tracks to vertices, with possible estimate of vertex position
    - Primary vertex reconstruction
    - Vertex finding in Jets
  - Vertex Fitting: Most precise estimate of the vertex position and track parameters at vertex from a set of tracks

- Two collections:
  - OfflinePrimaryVertices: Default primary vertex reconstructions
  - OfflinePrimaryVerticesWithBS: Primary vertex reconstructed, imposing the offline beam spot as a constraint in the fit of the vertex position.
- We advise for now to use the OfflinePrimaryVertices collection
- If no reconstructed vertex is found in an event:
  - A vertex based on the beam-spot is put into the event
  - flag isFake() is set to true
  - Contains no tracks, chi^2 =0, ndof = 0, and the

- Only primary vertex search is part of Standard Sequence and stored
- Further searches (secondary) have to be done by the user, selecting tracks of interest
  - It could be part of other sequences, e.g. conversions, V0 search, etc

  - Flags on the validity/type of vertex:
    - isValid
    - isFake: whether it is made from tracks, or from BeamSpot
- If the user performs a vertex fit/search himself, he may use also the TransientVertex, which provides more information

- Several algorithms available:
- VertexFitters: SWGuide
  - Kalman Filter: LSM fitter
  - Adaptive Vertex Fitter: soft-assignment, iterative, re-weighted LS fit
  - TrimmedKalmanVertex Fitter: hard-assignment, iterative LS fit
  - Gaussian-Sum Filter: Gaussian mixture of *pdf*s
  - Adaptive Gaussian-Sum Filter
- Vertex finders: SWGuide
  - AdaptiveVertexReconstructor
  - TrimmedKalmanVertexFinder
  - MultiVertexFit: Concurent Multi-Vertex Fit
  - TertiaryTracksVertexFinder
- Kinematic fit: fit with constraints (SWGuide)

Offline vertex reconstruction II

TransientTrack

- Default reco::Track not suitable for most higher-level algorithms (e.g. vertex, b/τ -tagging)
  - no access to magnetic field (no propagation!)
- Use Tracks through reco::TransientTrack
- In your application, build TT through TransientTrackBuilder:

```
//get the builder from the EventSetup:
edm::ESHandle<TransientTrackBuilder> theB;
iSetup.get<TransientTrackRecord>().get("TransientTrackBuilder",theB);
//do the conversion:
vector<TransientTrack> t_tks = (*theB).build(trackCollection);
```

Vertex Fitting and finding

- The object with which the user interacts is a VertexFitter or a VertexReconstructor:

```
KalmanVertexFitter fitter;
TransientVertex myVertex = fitter.vertex (vectorOfTransientTrack)
```

```
KalmanTrimmedVertexFinder finder;
vector< TransientVertex > vertices = finder.vertices (vectorOfTransientTrack)
```

ConfigurableVertexFitter

- Simplified usage through ConfigurableVertexFitter:
  - VertexFitter, that can be fully configured at runtime
  - Concrete VertexFitter used chosen at runtime through PSet

```
class ConfigurableVertexFitter : public VertexFitter {
  ConfigurableVertexFitter ( const edm::ParameterSet & );
```

- Documentation: SWGuide
- Fitters currently available:
  - KalmanFilter, Adaptive filter
- Examples PSet:

```
PSet vertexreco = {
  string fitter = "avf"
  double sigmacut =3.0
  double Tini=256
  double ratio=0.25 }
```

The configurables depend on the choice of the fitter!

ConfigurableVertexReconstructor

- Simplified usage through ConfigurableVertexReconstructor:
  - VertexReconstructor, that can be fully configured at runtime.
  - Concrete VertexReconstructor used chosen at runtime through PSet:

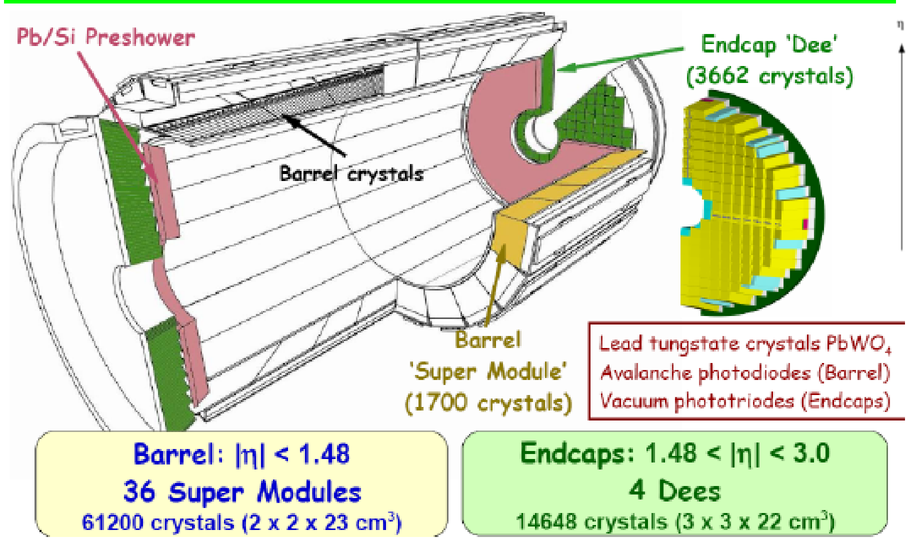
```
class ConfigurableVertexReconstructor :
      public VertexReconstructor {
  ConfigurableVertexReconstructor(const edm::ParameterSet&);
```

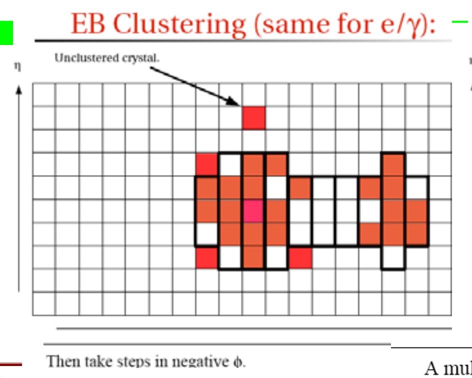
- Documentation: SWGuide
- Finders currently available:
  - Adaptive reconstructor, MultiVertexFitter, TrimmedKalmanVertexFinder

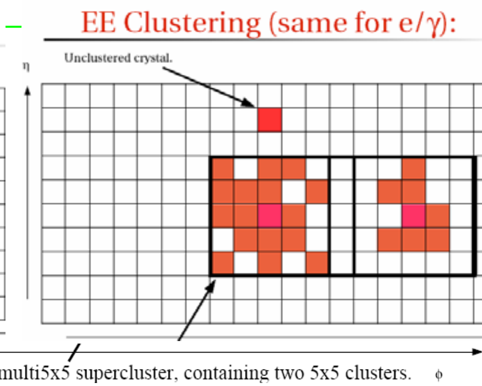
```
PSet vertexreco = {
  string finder = "avr"
  double primcut = 1.8
  double seccut = 6.0
  double minweight = 0.5 }
```

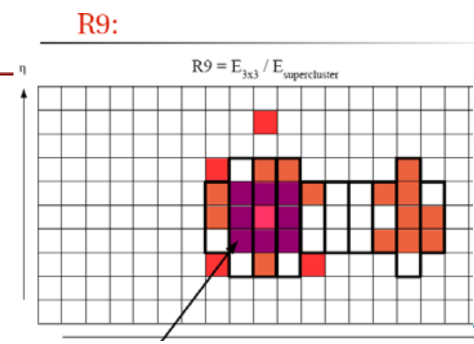
The configurables depend on the choice of the finder!

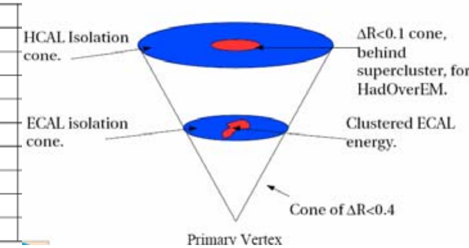
Reco::Photon











- In order to form physics objects, we cluster the crystals together. In order to make sure all of the energy is gathered up, we use 'supercluster' algorithms which are capable of making clusters of clusters.
  - Different algorithms used for barrel and endcap.

RecHits → SuperClusters → Corrected SuperClusters → reco::Photon + reco::PhotonID

SuperClusters → Clustered RecHits

- All the additional associated information about a reco::Photon is calculated and stored within reco::PhotonID.
- Things stored in reco::PhotonID:
  - Isolation in ECAL, HCAL and Tracker.
  - Fiducial flags.
  - Shower shape.
  - Basic Quality flags.

HadOverEM:

- Take the energy of your supercluster.
- Look in the HCAL behind the cluster position (currently in a cone of $\Delta R < 0.1$).
- The HCAL energy used is the tower behind the cluster, and therefore HadOverEM = $E_{HCAL}/E_{SuperCluster}$.
- Stored in reco::Photon.

Conversions:

- There is a dedicated algorithm for starting from ECAL clusters and attempting to reconstruct conversion tracks.
  - This really deserves it's own talk.
- However, the results are associated to the reco::Photon in the PhotonProducer, and thus I mention it here. The reco::Conversion object contains information about the tracks, a vertex (if available), and what clusters were used.

Isolation:

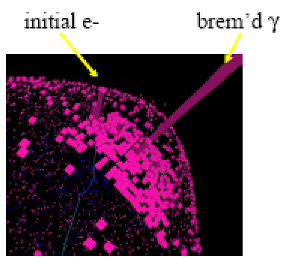
- ECAL Isolation: Based on sum of RecHits in a cone about the SuperCluster direction. A 'slice' in η excludes the clustered energy, and an inner cone excludes the real energy which is not clustered.
- HCAL Isolation: In a hollow cone of 0.4-0.1, excluding region already used for HadOverEM calculation.
- Track isolation: In both a solid cone and a 0.4-0.04 hollow cone.

Shower Shape:

- Currently the only shower shapes included are the R9 variable (covered earlier), and the energy weighted width in η, which is stored in the supercluster.
- More variables are going to be added:
  - $\sigma_{\eta\eta}$, the covariance of the 5x5 array.
  - 1x5 and 2x5 energies for ratios.

Reco::Electron

- Electrons are caracterized by a high E_T SuperCluster matched in position and momentum with a track
- Electron object can be more complicated due to showering in the traker material
  - Several subclusters (grouped into a supercluster)
  - Eventualy several tracks from photon conversions
- Electron object therefore
  - has a particle behaviour
  - is a composed object
  - handles combined information



« showering electron »

Super Cluster, Trajectory Seed → Electron Seed → Electron TrackCandidate → Electron Gsf track → GsfElectron

Seeding:
- Std track seeds
- SC driven pixel match filter
  - E_T, H/E

Trajectory building:
- CTF builder
- Electron loss modeling
- No Chi2 cut
- Reduced #candidates/layer

Gsf track fit:
- Electron loss modeling
- Mode of the gaussian mixture used for p_{ele}
- Brem fraction

Preselection:
- E/p
- dEta, dPhi

Amb. resolution

Electron seeding

- Electron seed reconstruction starts from standard tracker seeds and superclusters
  - Filtered by E_T and H/E
- Pixel match filter applied on standard tracker seeds
- Match the first hit backpropagating from supercluster E_T and position to the beam spot
  - Large windowz (z spread)
  - E_T dependent phi window
- Match the second hit propagating from the matched first hit up to the second hit layer
  - Tight windows

- A TrajectorySeed contains:
  - a TSOS
    - By convention defined on the outermost layer
  - a vector of RecHits
  - a propagation direction
- An ElectronPixelSeed adds:
  - Ref to SuperCluster
- Unlike TrajectorySeeds, ElectronPixelSeeds are stored in RECO data tier

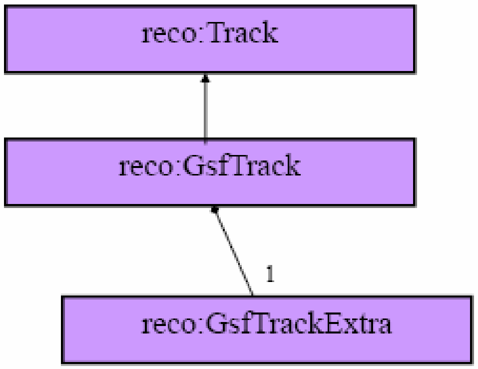
TrajectorySeed ← reco:ElectronPixelSeed ← (1) reco:SuperCluster

Electron tracking

- Electron reconstruction uses Ckf trajectory building with dedicated propagators and parameters
  - Energy loss modeling
  - No Chi2 cut
  - Reduced number of candidates per layer
- Gsf fit is used to evaluate track parameters
  - Energy loss modeling
  - Multicomponent TSOS
  - Further electron reconstruction makes use of mode estimate

Gsf track object

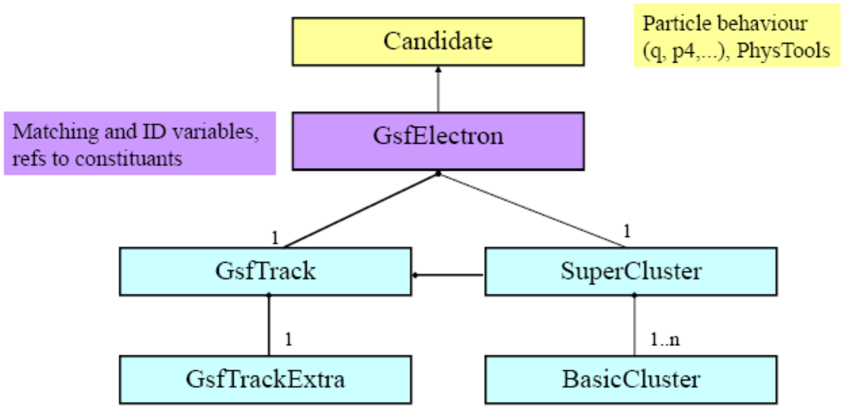
- A Ttrack contains (see Tracking tutorial):
  - Ref. position on the track
  - Momentum at this position
  - 5D curvilinear covariance matrix from the fit
  - Charge, Chi2 and ndof
  - Hit patterns (in which layers the track has hits)
- Through the TrackExtra (only in RECO)
  - Inner and outer track parameters with covariance matrix
  - A reference to the TrajectorySeed
  - Vector of Refs to RecHits
  - a propagation direction
- A GsfTrack adds:
  - Charge, momentum and momentum covariance from mode



- GsfTrackExtra is the standard « extra » track extension for the GsfTrack
  - Inner and outer multicomponent states

Electron preselection

- GsfElectrons are constructed from a reconstructed GsfTrack and a loose association with it's corresponding supercluster
  - Associated at the electron seeding stage
- The supercluster-gsf track preselection is loose so to keep high efficiency at the RECO stage
  - Compatible with affordable fake rate
  - Purity can be increased at a later stage applying electron ID
- Preselection is based on
  - Supercluster E_T
  - Delta eta between the SC position and track extrapolation
  - Delta phi between the SC position and track extrapolation
  - E/P
  - H/E

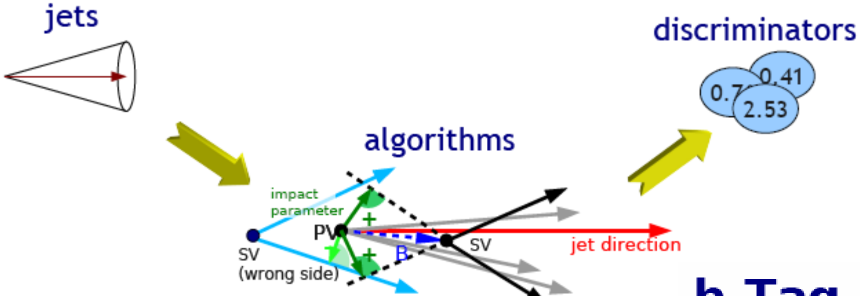


b-Tag

b-quarks significantly differ from light flavour quarks by:
- mass: m = 4.2 GeV
- lifetime: $\tau \approx 1.5$ ps → ~1.8mm *(at 20 GeV)* before decay
- decay: weak, mostly into c-quarks *(→ 3rd decay)* → 20% into leptons
- tracks: high decay multiplicity, significant displacement
- Secondary vertices *(SV)*: tracks intersecting at a common vertex

What is a b-Tag?
- Information associated to a **jet** *(in principal works for any jet)*
- Many different algorithms available
  - Universal / Specialized
  - Robust (for early data) / Complex
  - Different purity regions
- Information associated with jets
  - Tracks, vertices, leptons, …

Going to describe structure for CMSSW 1.7.X and newer (2.1.X)




```
reco::JetTag

reco::BaseTagInfo
reco::TrackIPTagInfo
reco::SecondaryVertexTagInfo
reco::SoftLeptonTagInfo
```

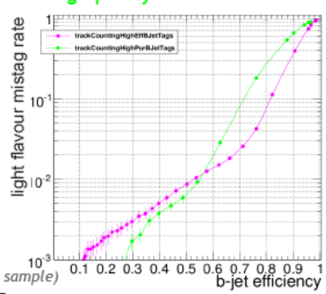
CMSSW package: `DataFormats/BTauReco`

b-Tag Discriminators

"Track Counting"

- A "simple" algorithm
  → suitable for early data
- Based on track impact parameters (IP)
  - 2nd highest (signed) IP significance in jet → high efficiency
  - 3rd highest (signed) → high purity
- discriminator is set to -100 in case too few tracks are available *(likely to be a bad jet)*

- `trackCountingHighEffBJetTags`
- `trackCountingHighPurBJetTags`

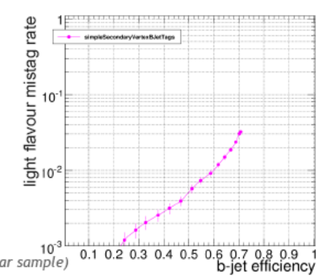


(2_1_8 on a TTbar sample)

"Simple Secondary Vertex"

- Another "simple" algorithm
  → suitable for early data *(most robust against misalignment)*
- Based on reconstruction of a Secondary Vertex in the jet
- discriminator is flight distance significance *(secondary vertex separation measure)*
- discriminator is set to -1 in case no vertex is found
- b-Jet efficiency limited by vertex finding!
  → will only reach ~70% with ideal detector

- `simpleSecondaryVertexBJetTags`



(2_1_8 on a TTbar sample)

"Jet Probability" and "Combined Secondary Vertex"

- More sophisticated algorithms
  → high performance
  → need to be "calibrated" first
  (need data from Conditions Database)

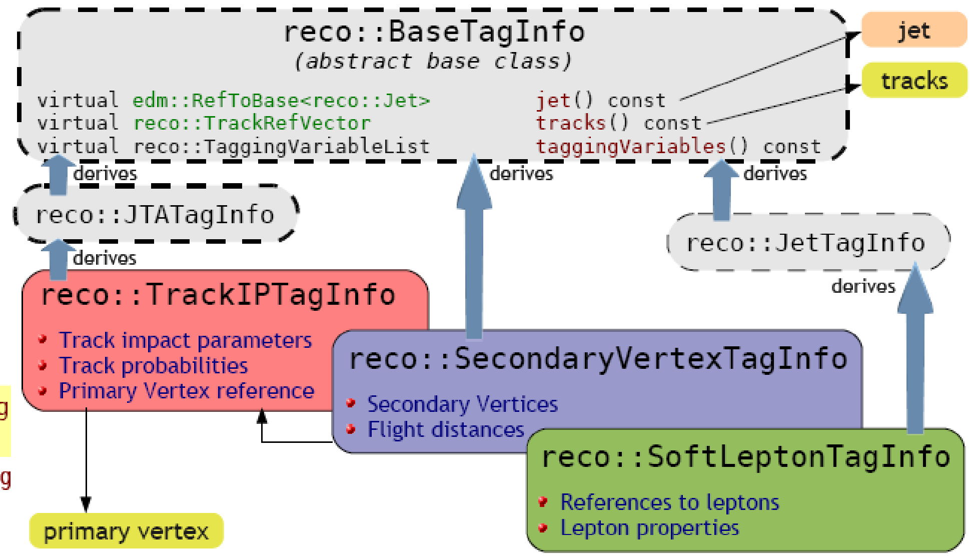
"Jet Probability"

- Using track IP PDF's in categories
- Computes an overall probability
- Two algorithm flavours:
- JetProbabilityBJetTags
  an sensible choice
- jetBProbabilityBJetTags
  high b-jet efficiency variant
  (best used on dijet events)

"Combined Secondary Vertex"

- Combines Track and Vertices
- Overall best performing
- A discriminator < 0 indicates a non-taggable jets
- combinedSecondaryVertexBJetTag
  a sensible choice
- combinedSecondaryVertexMVABJetTag
  variant employing a neural network

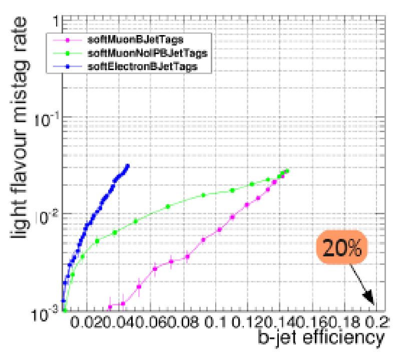
Intermediate collections with information used for tagging



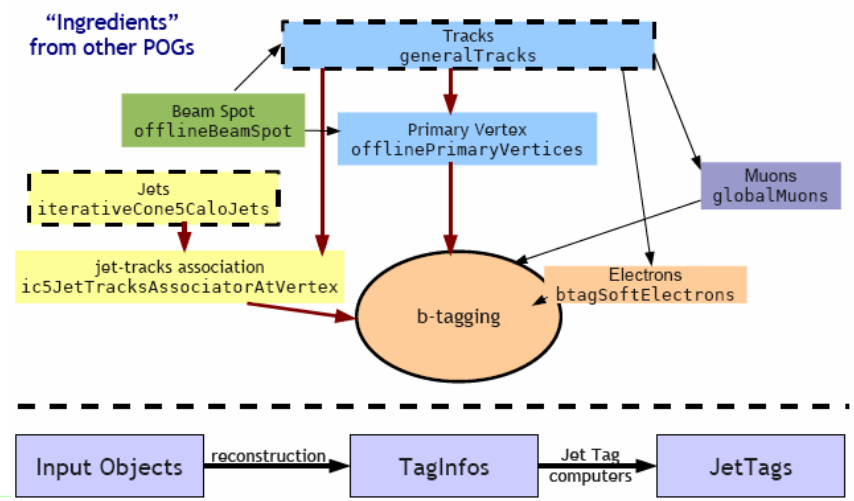
"Soft Lepton"

- Lepton-based algorithms
  → limited by lepton b-decay probability
  → fairly independent of detector alignment
- SoftMuonBJetTags
  uses "globalMuon" muons in jet
- SoftMuonNoIPBJetTags
  same, but without muon impact parameter
- SoftElectronBJetTags
  uses electrons instead of muons

  → depends on "btagSoftElectrons"
  dedicated electron ID
  for non-isolated electrons



Standard Sequence



Taus

Taus: a general introduction

- Mass 1.78 GeV/c^2,
- Lifetime 0.29 ps
- Decay modes:

Hadronic decays

Decay	Final state	BR(%)
Leptonic 35.2	$e+\nu_e+\nu_\tau$	17.8
	$\mu+\nu_\mu+\nu_\tau$	17.4
1-Prong	$\pi+\nu_\tau$	10.9
	$\pi+\nu_\tau+\geq 1\pi^0$	35.8
3-Prong	$\pi\pi\pi+\geq 0\pi^0+\nu_\tau$	13.9

τ-jets characteristics

- Very collimated and isolated
- Low multiplicity
- Electromagnetic and hadronic energy deposits

BUT

- a significant fraction of the τ momentum escapes undetected with the ν_τ

The TAU object

Hadronic TAU-jet candidate object

a jet + a collection of tracks associated to it + a discriminator which is the result of the tagging:

YES it's a tau NO it isn't a tau

TAU

CaloTau | PFTau

PF = ParticleFlow

CaloJets

Tracks

PFJets

PFCandidates (muons, electrons, charged and neutral hadrons, photons) starting from tracks and energy clusters

The TAU object

Hadronic TAU-jet candidate object

a jet + a collection of tracks associated to it + a discriminator which is the result of the tagging:

YES it's a tau NO it isn't a tau

TAU

CaloTau | PFTau

PF = ParticleFlow

CaloJets

Tracks

PFJets

PFCandidates (muons, electrons, charged and neutral hadrons, photons) starting from tracks and energy clusters

The TAU object

PF = ParticleFlow

PFTau → Fixed signal cone

PFTau → Shrinking signal cone

2 standard configurations for producing PFTau objects with different definition of signal and isolation cone sizes:

PFRecoTauProducer: fixed size signal cone 0.07 in ΔR space.

PFRecoTauProducerHighEfficiency: size signal cone that varies inversely with the E_T of the associated jet $5/E_T$: higher efficiency (in the low p_T region) at a cost of increased QCD isolation fake rate.

PF Candidates

Optimal combination of information from all subdetectors used

- Muons:
  - Reconstructed tracks and ECAL/HCAL clusters compatible with muon chambers
- Electrons:
  - Tracks and ECAL clusters
  - Energy losses and possible bremsstrahlung photons
- Calibrated HCAL cluster is compared with track momentum
  - Charged hadron: if E and p_T are compatible
  - Neutral hadron or photon: if E and p_T are incompatible
- Neutral hadrons: unassociated (to tracks) HCAL clusters
- Photons: unassociated isolated ECAL clusters

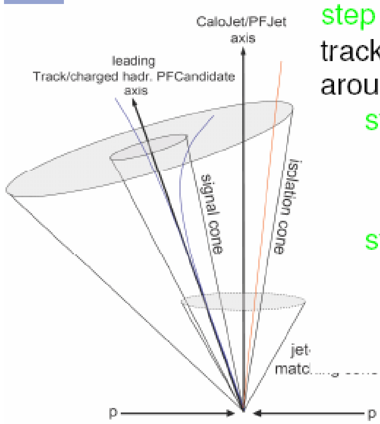
The complete list of particles can be used to derive physics objects such **PFJets**, which improves the performance compared to previous algorithms

Software Architecture

- The code is located in three 3 CMSSW packages :
- `DataFormats/TauReco` containing the classes defining objects, stored for each event and manipulated by an analyst,
  - a 1st layer: PFTauTagInfo objects used in the elaboration o the 2nd layer objects,
  - a 2nd layer: analyst-dedicated PFTau objects,
  - a discriminant layer, (AssociationVector) mapping discriminant output for PFTau objects
- `RecoTauTag/RecoTau` package, in which are contained the EDProducers of the former objects
- `RecoTauTag/TauTagTools` package, in which are containec possibly useful methods.

The PFTauDiscriminatorByIsolation

For each hadronic τ-jet candidate the discriminator returns 0 or 1 as the response of a procedure based on the following steps:



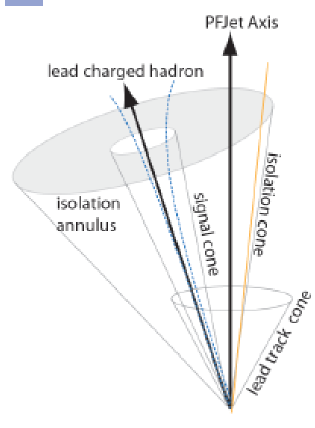
step 1 - ask for a leading (p_T>5GeV/c) track in a matching cone (ΔR=0.10) around the jet axis

step 2 - ask for 0/few (0) track(s) in an ($\Delta R_{innercone}$=0.10, $\Delta R_{outercone}$=0.50) isolation annulus around the leading element

step 3 may ask for 0/few (0) gamma PFCandidate(s) in an ($\Delta R_{innercone}$=0.15, $\Delta R_{outercone}$=0.50) isolation annulus around the leading element

/RecoTauTag/RecoTau/data/CaloRecoTauDiscriminationByIsolation.cfi
/RecoTauTag/RecoTau/data/PFRecoTauDiscriminationByIsolation.cfi

Tau base reconstruction: the isolation algorithm

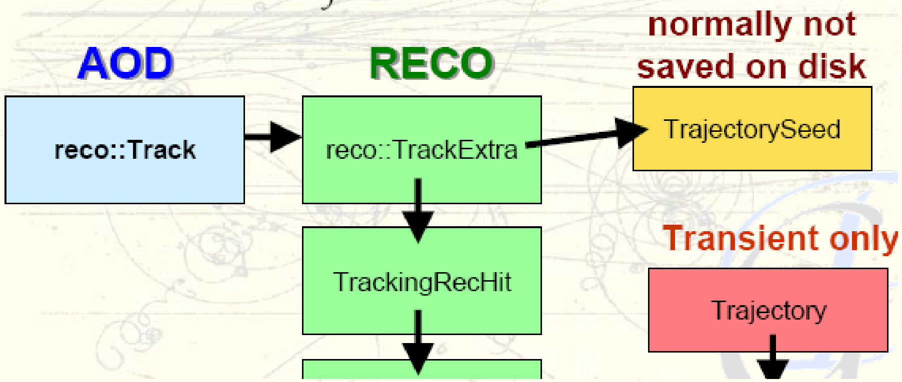


(courtesy of E. Friis)

- PF-based Jet reconstruction (p_T>15 GeV/c)
- At least one charged hadron with p_T > 5 GeV/c at a ΔR<0.1 from jet direction in the (η,ϕ) plane
- Signal cone and isolation annulus definition around the leading track, typical values are for the signal cone ΔR=0.07 and for the isolation annulus ΔR=0.45
- No charged hadron or photon candidates above a p_T threshold (1 GeV/c for the charged, 1.5 GeV/c for the gamma cand) in the isolation annulus

Track

- In CMSSW, information about tracks is split into different objects:



Track Quality

- The generalTracks come from the merging of different track collections, removing duplicates.
- By using the TrackQuality you can select subsamples optimized for different efficiency/purity trade-offs
  - **all generalTracks**: highest efficiency, fakes 5-20% depending on the pseudorapidity and pt range
  - "**highPurity**": the tightest selection; fake rate ~2%, with an efficiencty 2-5% lower than the inclusive generalTracks
  - "tight", "goodIterative": intermediate selections.
- Example usage in CMSSW:

```
if (track.quality(Track::highPurity)) { … }
```

1. Pixel-only track reconstruction is performed, followed by pixel vertex finding.
2. Seeds are reconstructed from pixel hit triplets, and from mixed hit pairs (pixels + some parts of TIB, TID and TEC to increase efficiency).
3. The two seed collections are merged
4. A first step of CKF pattern recognition is performed inside-out to produce track candidates. Then, an outside-in step is done to recollect hit overlaps on seeding layers
5. The KF fitting and smoothing procedure is applied to produce the tracks. At this step, outlyer hits with large chi2 are discared. The RungeKutta propagator is used to remove any bias from non uniform magnetic field.
6. The AnalyticalTrackSelector is used to classify the tracks into different qualities
7. The high purity sample is then used as a first step for a three-step iterative tracking
8. In the second and third step
   - Clusters associated to the tracks are removed from the input collection (but large clusters from merged hits are not removed)
   - The full chain is repeated, but with looser cuts: hits from clusters, seeds from hit, track candidates from seeds, tracks from candidates, and a filter to
9. Tracks from all the second and third step of iterative tracking are merged with the generalTracks, removing duplicates by looking at the fraction of shared hits.
10. The output track collection is the **generalTracks**

Particle Flow

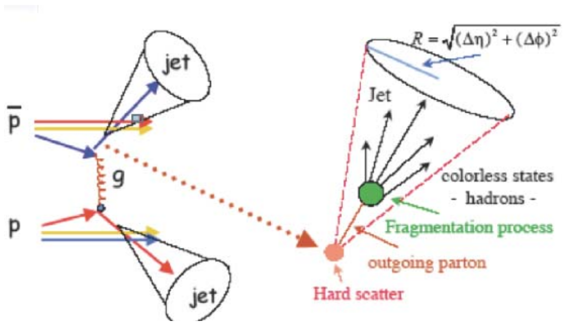
- electron BEING IMPLEMENTED
- muon BEING IMPLEMENTED

The core of the particle flow algorithm reconstructs photons, charged hadrons, and neutral hadrons and consists in

- calorimeter clustering (ECAL, HCAL, PS),
  - produces PFClusters
- track reconstruction and extrapolation to the calorimeters,
  - produces PFRecTracks
- reconstructing blocks of topologically connected elements (tracks, ECAL clusters, HCAL clusters, PS clusters)
  - produces PFBlock
- analyzing these blocks to reconstruct particles.
  - produces PFCandidates

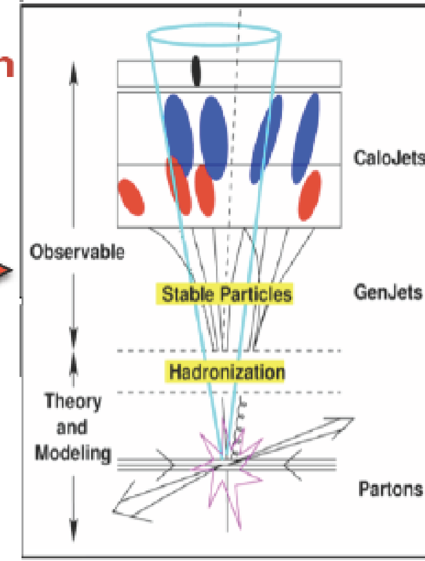
- **Fast simulation**
- **Full simulation**
- **PAT**

Jets & MET



Jet Algorithms: Introduction

- Jets are expected to represent the underlying partons
  - Yet, a jet is what the jet algorithm defines it to be
- Desired properties of jet algorithms:
  - Same behavior at detector/particle/parton levels
  - Infrared and collinear (IRC) safe (ie. not sensitive to adding soft particles or splitting a 4-vector into two smaller)
  - Not too sensitive to details of detector, pileup at high lumi, non-perturbative processes
  - Reliable calibration
  - Computationally efficient
- Major classes of jet algorithms:
  - Cone: cluster objects close in angle
    - Simple shape, unless jets overlap
  - KT: cluster objects close in relative p_T
    - Irregular shape



Algorithms

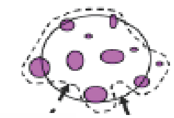
Iterative Cone

- Iteratively searches for stable cones of size $R=\sqrt{(\Delta\eta^2 + \Delta\phi^2)}$
- "Cookie cutter" – input objects assigned to a jet are removed before the next iteration
- No splitting/merging
- Seed based, not IRC safe
  - Seed ET > 1 GeV

Midpoint Cone

- Also seed based
- Adds extra seeds between stable cones ("midpoints")
- Not IRC safe beyond NLO
- Does not remove "used" inputs
- Applies splitting/merging
- Leaves unclustered energy

- **SISCone**
  - Seedless IRC Safe Cone algorithm
  - Searches for ALL stable cones
  - Applies splitting/merging
  - IRC safe to all orders
  - No unclustered inputs
- (Fast) **kT**
  - Uses sequential recombination of 4-vectors based on relative kT
  - Controlled by the jet separation parameter D (determines jet "size")
  - IRC safe to all orders
- All algorithms cluster 4-vectors and can be used at detector, particle, and parton levels.
- For calorimeter inputs, all algorithms use Scheme B thresholds for cell energies and tower ET > 0.5 GeV cut



Jet Flavors

* **CaloJets:** jets produced from CaloTowers
* **GenJets:** jets produced from generator level MC particles
* **PFJet:** jets produced from Particle Flow candidates.
* **BasicJets:** jets produced from arbitrary collection of Candidate inherited objects.

Every jet contains information about its 4-momentum.
This is the quantity to use for generic kinematics analysis.

CaloJet also contains information about the fraction of energy deposited in a particular calorimeter compartment or group of compartments: ECAL, HCAL, HB(barrel), HO(outer), HE(endcap), HF(forward).

GenJet contains information about fraction from different types of generated particles, i.e. from charged hadrons.

PFJet conains information about energy contribution and number of contribuing different types of Particle Flow objects.

MET: negative transverse vector sum

(example CaloMET: sum over uncorrected projective Calo Towers)

$$\vec{E}_T = -\sum_n (E_n \sin\theta_n \cos\phi_n \hat{\mathrm{i}} + E_n \sin\theta_n \sin\phi_n \hat{\mathrm{j}}) = -E_x \hat{\mathrm{i}} - E_y \hat{\mathrm{j}}$$

- Collections
  - CaloMETCollection (~6kB/event)
    - met - *raw caloMET from caloTowers (scheme-B thresholds)*
    - metNoHF - *raw caloMET from caloTowers (scheme-B). Note: HF not used in SET or MET quantities*
    - metOpt - *raw caloMET from caloTowers (optimized thresholds)*
    - metOptNoHF - *raw caloMET from caloTowers (optimized thresholds). HF not used in SET or MET*
  - METCollection (~6kB/event)
    - htMetSC5 - *MET from sisCone5CaloJets*
    - htMetSC7 - *MET from sisCone7CaloJets*
    - htMetKT4 - *MET from kt4CaloJets*
    - htMetKT6 - *MET from kt6CaloJets*
    - htMetIC5 - *MET from iterativeCone5CaloJets*
    - genMetFromIC5GenJets - *MET from iterativeCone5GenJets*
  - GenMETCollection (~3kB/event)
    - genMet - *MET from genCandidatesForMET (no BSM, neutrinos, or muons in negative vector Et sum)*
    - genMetNoNuBSM - *MET from genCandidatesForMETNoNuBSM (no BSM, prompt neutrinos, prompt muons in negative vector*
      - NOTE: Bug in this collection present from 200 up to 210 - took particles for MET calculation from faulty list. Remedied for

MuonMET (Met corrections for Global Muons)

$$\mathrm{MET} = -\sum_{i=1}^{\text{towers}} E_T^i - \sum^{\text{muons}} p_T^\mu + \sum_{i=1}^{\text{deposit towers}} E_T^i.$$

Corrects MET for muon response, since the muons deposit only small energies in the calorimeters, typically a few GeV being equivalent to minimum ionizing particle energy (at a wide range of muon momentum)

-> The MET in events with muons thus needs to be corrected for muon momenta and muon energy deposition in the calorimeters;

TypeI MET (See CMS AN-2007/041 for details)

* Corrects JES using : iterativeCone5CaloJets , midPointCone5CaloJets, and midPointCone7CaloJets

$$E_T{}^{\mathrm{corr}} = E_T - \sum_{i=1}^{\mathrm{N_{jets}}} \left[\underbrace{p_{T_i}^{\mathrm{corr}}}_{\text{CorrectedJet}} - \underbrace{p_{T_i}^{\mathrm{raw}}}_{\text{RawJet}} \right],$$

... where the sum runs over all the jets with raw pt raw greater than a *jetPTthreshold* and EMF less than *jetEMfracLimit*. (to exclude electrons)

Tau MET corrections

$$\Delta \vec{E}_T = \sum \underbrace{\vec{E}_T^{\mathrm{cal}}}_{\text{CaloJet}} - \underbrace{\vec{E}_T^{\tau}}_{\text{PFTau}} - \underbrace{\vec{E}_T^{\mathrm{UE\ res}} - \vec{E}_T^{\mathrm{PU\ res}}}_{\text{Not icluded @ the moment ..}},$$

* Tau jets are over-corrected by standard jet corrections ...
* ... but can be measured accurately using particle Flow

User Analysis on Tier-2s

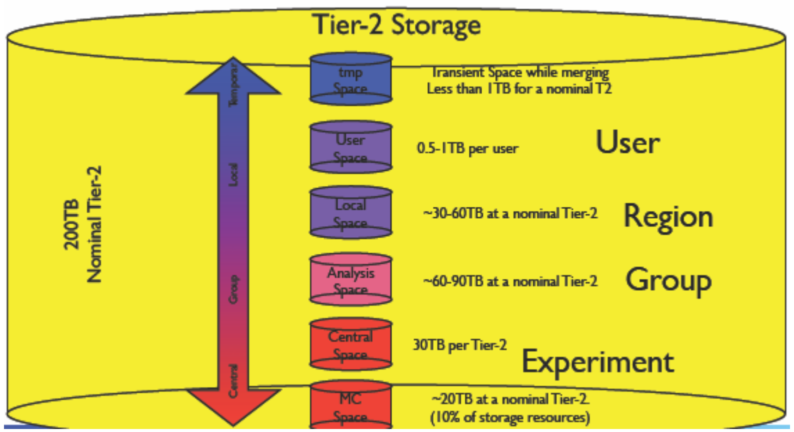
The CAF at CERN is a very valuable resources

- It will have access to the really prompt reconstruction and calibration samples the quickest
- It's useful for low latency analysis and some other very high profile tasks involving data promptly
- Unfortunately it's small

How is the Storage managed?

Storage at Tier-2 centers is broken into 6 pieces

- Transient and unmanaged to more persistent and centrally managed



All numbers are for a nominal Tier-2

Central Space 30TB

- Intended for RECO samples of Primary Datasets.
  - In 2008 we had expected to be able to store 2 copies of MC and data sample using the identified T2 space

Physics Group Space 60-90TB

- Assigned to 1-3 physics groups. Space allocated by physics data manager. The site data manager still approves the request, but only to ensure the group is below quota

Local Storage Space 30TB-60TB

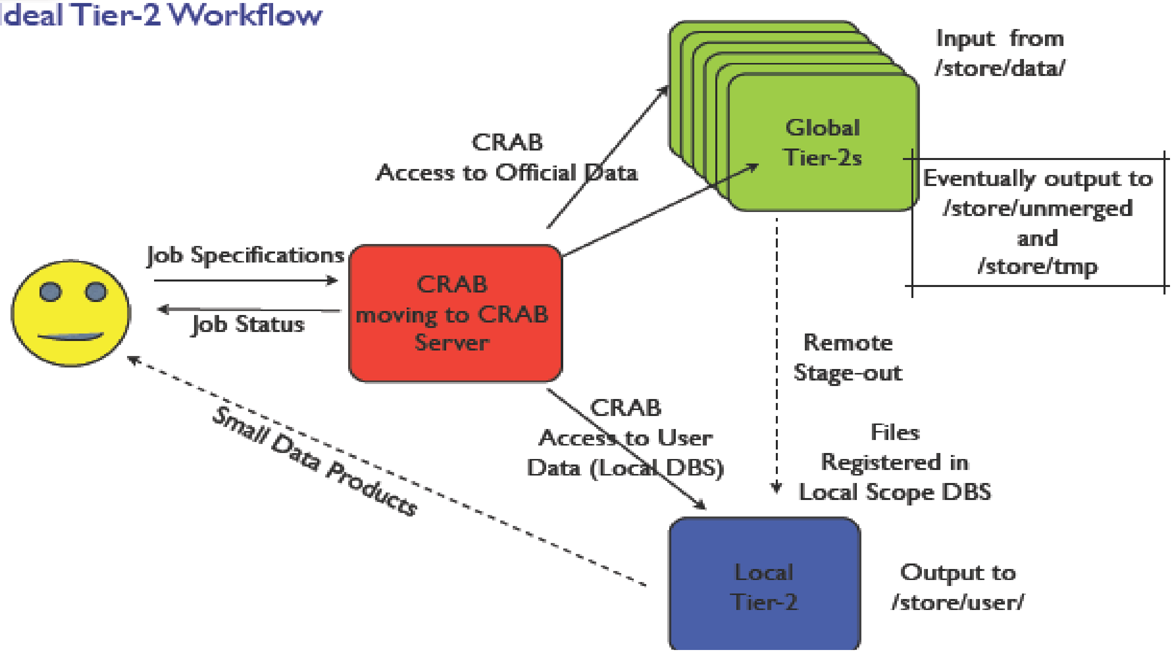
- Controlled by the local storage manager. Intended to benefit the geographically associated community

User Space 0.5-1TB per person in the geographically associated community

- controlled by individuals

Tier-2 Analysis Workflow

Ideal Tier-2 Workflow



How does it impact you?

- CRAB will continue to submit your jobs to where ever publishes the data, but we are trying to ensure there are good support connections between the sites and the groups they are supporting
- Someone in the analysis group you work in is empowered to ask that data samples be moved into the space controlled by the group
  - You should be able to ask that person to replicate samples that are interesting to you

Toturial: configuration file --PYTHON time

Motivation

- Many tools handling configurations are written in Python
- In the long run easier maintenance
- More flexibility for the end user, which means for you!
- Old config be deprecated

http://indico.cern.ch/conferenceDisplay.py?confId=37576

the CMS-specific configuration file language

↓

The Python coding language

https://twiki.cern.ch/twiki/bin/view/CMS/WorkBookConfigFileIntro

https://twiki.cern.ch/twiki/bin/view/CMS/SWGuideAboutPythonConfigFile

An example

```
import FWCore.ParameterSet.Config as cms
import Foo.Bar.data.somefile

process = cms.Process("RECO")

process.source = cms.Source("PoolSource",
        fileNames = cms.untracked.vstring("test.root") )
process.tracker = cms.EDProducer("TrackFinderProducer")

process.out = cms.OutputModule("PoolOutputModule",
fileName = cms.untracked.string("test2.root") )

#add the contents of Foo.Bar.data.somefile to the process
process.extend(Foo.Bar.data.somefile)

process.p = cms.Path(process.tracker * process.out)
```

old config

```
include "RecoJets/JetProducers/data/CaloTowerSchemeB.cfi"
```

Python config

```
from RecoJets.JetProducers.CaloTowerSchemeB_cfi import *
```

old config

```
process = foo {
   [...]

   module udsfilter = JetFilter{ ...}

   module jets = JetReco{ ...}


   [...]

   replace udsfilter.jetType = 2

   [...]

   path mypath = (jets, udsfilter)


}
```

python config

```
process = cms.Process("foo")
[...]

udsfilter = cms.EDFilter("JetFilter",...)
process.udsfilter = udsfilter

process.jets = cms.EDProducer("JetReco",...)

[...]

process.udsfilter.jetType = 2

[...]

process.mypath = cms.Path(process.jets*udsfilter)
```

```
$ cmsRun config.cfg
```

old config

```
$ cmsRun config_cfg.py
```

Python

```
type name = value
```

→

```
name = type( value )
```

```
module foo = FooFilter {}
```

→

```
foo = EDFilter("FooFilter")
```

```
process.foo = EDFilter("SomeFilter",
                       MeanX = double(0),
                       MeanY = double(0),
                       MeanZ = double(0),
                       log = untracked.bool(True))
```

```
module foo = SomeFilter {
                       double MeanX = 0
                       double MeanY = 0
                       double MeanZ = 0
                       untracked bool log = true}
```

old config	python config
source	Source
module	EDAnalyzer, EDFilter, EDProducer, OutputModule
service, es_source, ...	Service, ESSource, ...

Reminder:
Since 1_5_0 there are stricter rules what kind of modules are allowed in Paths and EndPaths!

old cfg

```
path p1 = {foo, bar & foobar }
```

precedence of
* over +
maybe more intuitive!

python

```
p1 = Path( foo * bar + foobar )
```

new feature:
can be modified

old config

```
module udsfilter = JetFilter{ ...}
replace udsfilter.jetType = 2
```

python config

```
udsfilter = cms.EDFilter("JetFilter",...)
udsfilter.jetType = 2
```

```
replace musician += "Madonna"
replace painters += { "Picasso", "da Vinci"}
```

```
musician.append("Madonna")
painters.extend( ("Picasso", "da Vinci") )
composers.pop() # don't like Madonna
```

old config

```
block Record = {
   untracked string composer = "Beethoven"
}
```

Python config

```
Record = cms.PSet(
composer = cms.untracked.string( "Beethoven" )
)
```

old config

```
include "SomePlace/Record.cff"

module MyModule = SomeFilter {
    using Record
}
```

Python config

```
from SomePlace.Record_cff import Record

MyModule = EDFilter("SomeFilter",
                    Record
                   )
```

The central process configuration is slightly different

- Things get only run if attached to the process object
- This gives you more flexibility of selecting what you want
- Whole file contents get added via process.extend(...)
  (equivalent to the old include "...")
- Individual objects by assignment to the process:
  process.pathName = cms.Path(process.a * process.b)

- Even "complete" processes from another file can be modified

```
from Configuration.Examples.RecoExample_cfg import process

process.source.fileNames = ("file://myfile.root")
```

Things you should know

- Each file starts with
  import FWCore.ParameterSet.Config as cms
- Comments start with a sharp ("#")
- Leading spaces are important / indentation is part of the syntax
- Don't forget the comma between parameters
- In a scram environment "python <config.py>" can check if your configuration file is well formed
- To translate an old config yourself use "cfg2py.py <old.cf?>"
  (if you translate a longer .cfg file you should get a coffee first)

CMSSW file input& output, Trigger Bits

input

```
process.source = cms.Source("PoolSource", ...)
```

PROCESS

output

```
process.out = cms.OutputModule("PoolOutputModule",

  fileName = cms.untracked.string("output.root"),
```

```
cms.untracked.vstring(

    "keep *",
    "drop *_*_*_HLT",
    "keep FEDRawDataCollection_*_*_*"
)
```

Output files are written via the PoolOutputModule

```
cms.OutputModule("PoolOutputModule",
    outputCommands = RECOEventContent.outputCommands,
    fileName = cms.untracked.string('TTbar_cfi_GEN_SIM_DIGI.root'),
    SelectEvents = cms.untracked.PSet(
        SelectEvents = cms.vstring('*Electron:HLT')
    )
)
```

selectEvents TWiki

https://twiki.cern.ch/twiki/bin/view/CMS/SWGuideEDMPathsAndTriggerBits

Accessing event data

Data inside the event are called "Product"

moduleLabel : productInstanceLabel : processName

```
# by module and default product label
 Handle<TrackVector> trackPtr;
 iEvent.getByLabel("tracker", trackPtr );

# by module and product label
 Handle<SimHitVector> simPtr;
 iEvent.getByLabel("detsim", "pixel" ,simPtr );

# by type
 vector<Handle<SimHitVector> > allPtr;
 iEvent.getByType( allPtr );

# by Selector
 ParameterSelector<int> coneSel("coneSize",5);
 Handle<JetVector> jetPtr;
 iEvent.get( coneSel, jetPtr );
```

At every position in the job one can inspect what products can be accessed:

```
process.dump = cms.EDAnalyzer('EventContentAnalyzer')
process.p = cms.Path(... + dump + ...)
```

```
++Event     0 contains 161 products with friendlyClassName, moduleLabel and productInstanceName:
...
++recoElectrons "siStripElectronToTrackAssociator" "siStripElectrons"
++recoGenJets "Fastjet10GenJets" ""
++recoGenJets "Fastjet10GenJetsNoNu" ""
++recoGenJets "Fastjet6GenJets" ""
++recoGenJets "Fastjet6GenJetsNoNu" ""
...
```

Given a file a similar thing can be done at command line:

```
edmDumpEventContent <filename>
```

Crab: Cms Remote Analysis Builder

- Every configuration in CRAB is set through the directives reported in the crab.cfg file
  - Organized as key = value pairs
  - Grouped in macro-sections [CRAB], [CMSSW], [USER], …
- A minimal and a full template for crab.cfg are in $CRABPATH/crab.cfg, $CRABPATH/full_crab.cfg
- Inline documentation (crab –h) guides you to set attributes

- Essential step to re-run with Grid and CRAB
- To take care before to publish your data
  - You must be registered in SiteDB
  - You must know the local DBS instance where to publish
  - You must know a Tier2 StorageElement where to store data

Publication

- Moreover
  - crab.cfg must contain the publication directives BEFORE creation The .root must be an EDM file
- https://twiki.cern.ch/twiki/bin/view/CMS/SWGuide CrabForPublication

- Create the CRAB project (by default crab.cfg)
  - crab –create
- Submit your jobs
  - crab –submit <all | n | rng > [–c <crab_prj>]
- Track the jobs progress
  - crab –status [–c <crab_prj>]
  - alternative use of the CRABSERVER web interface
- When jobs get done, retrieve data:
  - crab –getoutput <all | rng > [-c <crab_prj>]
  - output will store in <crab_prj/res>

- If you need to kill some job
  - crab –kill <all | n | rng > [–c <crab_prj>]
- Get post-mortem infos
  - crab –postMortem <all | rng > [-c <crab_prj>]
- Resubmit
  - crab –resubmit <all | rng > [-c <crab_prj>]
- Publish your results, if you need to share them
  - crab –publish [-c <crab_prj>]
- Clean the obsolete CRAB project
  - crab –clean [-c <crab_prj>]

DBS: Dataset Bookkeeping System

- DBS & DLS-- CMS data discovery service
- StandAlone/Server: o(100 jobs)/large tasks
- user data Crab publication to DBS

--publish the results of analysis to their local DBS

  - 1.SiteDB registration: https://twiki.cern.ch/twiki/bin/view/CMS/SiteDBForCRAB
  - StorageElement: local Tier2 /store/(user LFN namespace) in addtion to /store/(mc LFN namespace)
  - a private DBS instance
  - publish data at the job creation time

dataset name and LFN(logical file name)
/<primarydataset>/<yourHyperNewsusername>-<publish_data_name>-<PSETHASH>/USER
*<PSETHASH> is calculated from pset.cfg
/store/user/<yourHyperNewsusername>/<primarydataset>/<publish_data_name>/<PSETHASH>/<output_file_name>

** if the copy of output problem, the logical file name will be
/copy_problems/<output_file_name>

***if the publish option selected in crab.cfg, the StorageElement directory where to copy will be
/<storage_path>/<yourHyperNewsusername>/<PrimaryDS>/<publish_data_name>/<PSETHASH>/

DBS private data publication

```
[USER]
copy_data = 1
storage_element = t2-srm-02.lnl.infn.it
storage_path = /srm/managerv1?SFN=/pnfs/lnl.infn.it/data/cms/store/user

publish_data=1
storage_element = cmsdcache.pi.infn.it
storage_path = /srm/managerv1?SFN=/pnfs/pi.infn.it/data/cms/store/user
```

```
publish
1. create and submit all jobs
2. retrieve all the outputs
then you can issue:
  crab -publish
to check published:
the script InspectDBS2.py located in the python dir of CRAB:
 ./InspectDBS2.py --DBSURL=<dbs_url_for_publication> --
datasetPath=<name_of_your_dataset>

to delete a dataset
./DBSDeleteData.py --DBSURL=<dbs_url_for_publication> --
datasetPath=<name_of_your_dataset>
```

```
run analysis
[CMSSW]
datasetpath=<primarydataset>/<publish_data_name>/USER
### DBS/DLS options
dbs_url = <dbs_url_for_publication>#DBS URL : http://cmssrv17.fnal.gov:8989/DBS108LOC1/servlet/DBSServlet
```

Offline ConditionsData/FrontierConditions

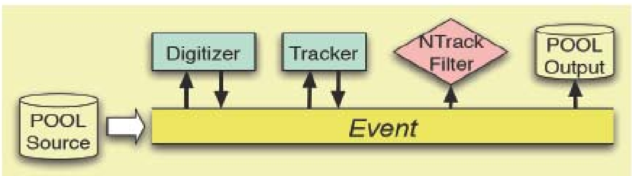
Conditions data for calibration and alignment are defined in ORCOF(Off-line Reconstruction Conditions DB Off-line). From the 200 release onwards, the set of database tags which together define the offline conditions data are collected together in a Global Tag, which is itself stored in the database.

- Triggers Path for Startup Conditions
  - **AlCaReco streams for collision data**
    - 3 TkAlJpsiMuMu Tracker alignment
      HLT_DoubleMu3_JPsi, HLT_DoubleMu4_BJPsi
- Start-up scenario (aka "SurveyLASCosmics" scenario) laser alignment system
- **Global Tag 21X**
  - **COSMMC_21X** for COSMICS MC. Expected calibration and alignment conditions at startup
  - **IDEAL_V9** Ideal/trivial conditions - perfectly aligned and calibrated detector
  - **STARTUP_V7** Expected calibration and alignment conditions at startup
- **Global Tag 20X**
  - **CSA08_S156** Conditions from S156 (10pb-1) calibration and alignment in CSA08
  - **CSA08_S43** Conditions from S43 (1pb-1) calibration and alignment in CSA08
  - **IDEAL_V2**
  - **STARTUP_V2**
- **FakeConditions**

CMSSW & PhysicsAnalysisToolkit

The CMS Event Data Model

- the core of the CMS software is often referred to as "the framework"
- based upon the Event Data Model (EDM)
  - *the central concept is the "event"*
  - *an event is immutable: existing information in the event cannot be changed*
  - *during processing exchange of information between modules happens only via the event*
  - *the event is processed along a "path", an ordered list of modules*
- working with modules
  - *EDProducer, EDFilter, EDAnalyzer, OutputModule (and EDLooper)*
  - *modules to be arranged in sequences and to go into paths*



The different software components

- the main CMS software: CMSSW
  - *contains everything from the core framework to analysis code*
  - *also contains simulation, FWLite, Iguana*
- daily workhorse: ROOT
- code compilation and external software configuration: SCRAM
- code management and reference: CVS, LXR, Doxygen
- job submission to the grid: CRAB
- data storage: ROOT, POOL

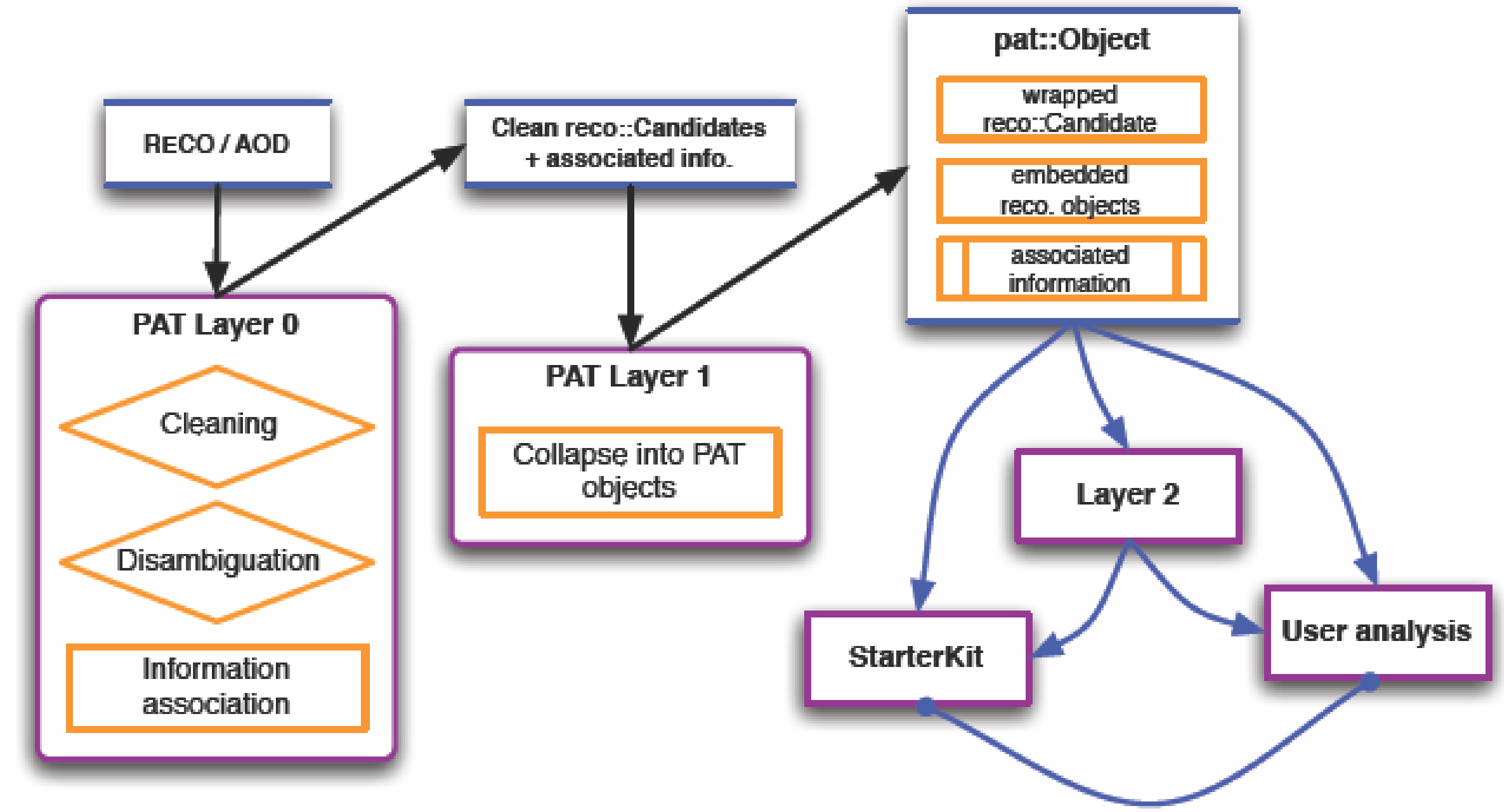
Analysis within the CMS Computing Model

- event data is organized in so-called **data-tiers**
  - *RAW: output HLT; primary archive format; input offline reconstruction*
  - *RECO: offline reconstructed objects*
  - *FEVT: RAW + RECO*
  - *AOD: format for physics analysis; subset of RECO*
  - *GEN: Monte-Carlo information*
  - *SIM: simulated energy depositions (simhits)*

The PAT in layers

- a multi-layered approach was needed to provide both maximal flexibility and user-friendliness within the constraints of the EDM
- event interpretation
  - *"Layer 0": cleaning and disambiguation ➡ event interpretation + additional analysis-level tasks (e.g. MC matching)*
  - *"Layer 1": creation of PAT objects that collapse externally associated information ➡ no algorithmic tasks*
- event hypothesis
  - *"Layer 2": event hypothesis dependent tasks ➡ provide possibility for re-tuning event interpretation*
- analysis
  - *"Starter Kit": for data exploration and plotting*
  - *"paste-your-analysis-here"*

PAT: 3 layers



RECO / AOD
Clean reco::Candidates
+ associated info.
pat::Object
wrapped
reco::Candidate
embedded
reco. objects
associated
information
PAT Layer 0
Cleaning
Disambiguation
Information
association
PAT Layer 1
Collapse into PAT
objects
Layer 2
StarterKit
User analysis

RooFit, RooStats, RooStatsCms

- Intra-experiment common tools for statistics: RooStats
  - developers: Kyle Cranmer (ATLAS), Gregory Schott (CMS), Lorenzo Moneta (ROOT), Wouter Verkerke (RooFit)
  - endorsed by CMS statistics committee (we are their technical arm)
  - built on top of RooFit
    - consequence: a built-in preference for RooFit
    - not necessarily a bad thing, many CMSers used it before

- Common RooStats still gelling together
  - Gregory and Danilo Piparo provided RooStatsCms
    - move some tools to RooStats – but keep some for CMS
    - distribution: CVS checkout right now, ship with CMSSW in the future

- Concern: long-term support
  - need to develop a base of (semi-)expert users of RooFit & RooStats

CMS Offline, Computing, Trigger

Fullsim production done (200 Mevt) with CMSSW 2_1

1. CMSSW 2_2

i) New Particle Flow algorithm and data format.

ii) Re-digitization and re-reco production

iii) Fast simulation (> 0.5 B evts)

iv) Analysis for CRAFT and Global Runs data

v) new PAT version for analysis data produced with 2_1 and 2_2 releases.

2. CMSSW 3_0 (Jan/Feb 09)

Data format changes allowed,

GEANT4, Root, etc

3. Trigger Tables up to 10^{32}

New procedures being put in place

Primary Data Sets

CMS Physics Plan

- **Prior to Sep 19: everything and everyone had been directed towards the imminent arrival of data. Now:**
- **Continue work on early publications**
  - Aim to have full drafts ahead of time (data-taking)
- **Continue the 900 GeV analyses – to completion**
- **Start analyses with the 10 TeV samples from CMSSW_2_1**
  - Recall: they have 3.8 T, 10 TeV and new tracker format (backwards incompatible)
- **Restart Monte Carlo analysis approvals? Yes.**
  - Given that we have a few months now ahead, we can update some of our results (especially for 10 TeV)
    - Need to tend to needs of younger collaborators working on theses and limited-term position
  - To restart physics studies: planning a fastsim production
    - Assume 10 TeV, inst. Lumi up to 10^{32}, $50pb^{-1}$ like before. As soon as fastsim is ready. We'll go with CMSSW_2_2.

CMS Physics Goals

- **Consolidate our state: there are parallel activities related to "tools" and the "how-to" do physics**
  - **We should deploy the PAT throughout all groups**
  - **We have to co-organize the "feedback loops" that are currently taking too long:**
    - Release➔validation➔bugfind➔bufix➔re-release
    - Also check/improve on the "analysis turn-around"
  - **We need to complete the definitions of the trigger menus for the different luminosities**
  - **We need to complete the definitions of the primary datasets**
  - **We need to run analysis (extensively, for long periods of time, with many people) at the Tier-2's**

Fast Simulation basic concepts

- Speed: 400ms (sim and tk) + 2-3s (reco and HLT) per event
- One single production workflow: Sim+L1+HLT+Reco
- Reprocessing is meaningless:
  - Easier to re-simulate from scratch or from the same GEN files
- Objects produced:
  - the same objects as in the full sim/reco are produced
  - No Raw data format. Digis available for all detectors (but the Tracker) but still created in a special way (at the same time as RecHits)
  - AOD event content IDENTICAL to standard one
- Simulation with Pile-Up easy:
  - The pileup events are superimposed at the generator level
  - They are simulated at the same time as the signal
  - Easier to have simulation in different conditions

Fall 08 FastSim Production

- Release: 2_2_X
- Centre of Mass energy = 10 TeV
- Magnetic field : 3.8 T
- Detector calibration/alignment: IDEAL
- No Pileup
- No Preshower
- Generator Madgraph (only?). Filtering on generator only
- Physics processes: being defined...
- HLT table for $2x10^{32}$. No filtering on HLT.
- Event Content : AODSim

- 2_2_X:
  - Add the latest and greatest Particle Flow improvement which are not backward compatible
  - Fix the L1/HLT (broken in FastSim for the entire 2_1_X series)
    - incidentally, broken also in FullSim since 2_1_8
    - We discovered this on Oct 1st. ~1 month to have a fix.
  - 2_2_X = 2_1_X + PF tags + L1/HLT tags
    - However it seems way more things went in...

Famos So far….

- Planning for the Fall 08 FastSimulation production started long time ago.
  - Lots of work went into a 2_1_X that would be satisfactory for production
  - After the 2_1_10 validation we felt close to be ready for FastSim
- Plans changed to move to 2_2_X and accommodate:
  - Particle Flow latest improvements (data format change) + L1/HLT bug fixes
- However, the amazingly large number of extra changes in 2_2_X (plus the incoming 3_0_x integration):
  - has kept us busy with day-to-day maintenance
  - left no time for real debugging/development of issues already seen in 2_1_10 Validation
  - brought to a situation where we need a validation from scratch of the FastSim in this release (and maybe FullReco as well)
- **Need for a succesful production:**
  - **Some quiet time to work, debug, fix in a stable, closed release.**
  - **Complete Validation of the FastSim, L1/HLT before (pre)production**