Tracking and B-Tagging

Markus Elsing

- Status of new Inner Detector reconstruction
- Overview of b-tagging tools
- Comments on B-tagging, Pixel software and tracking

Status of Track Reconstruction

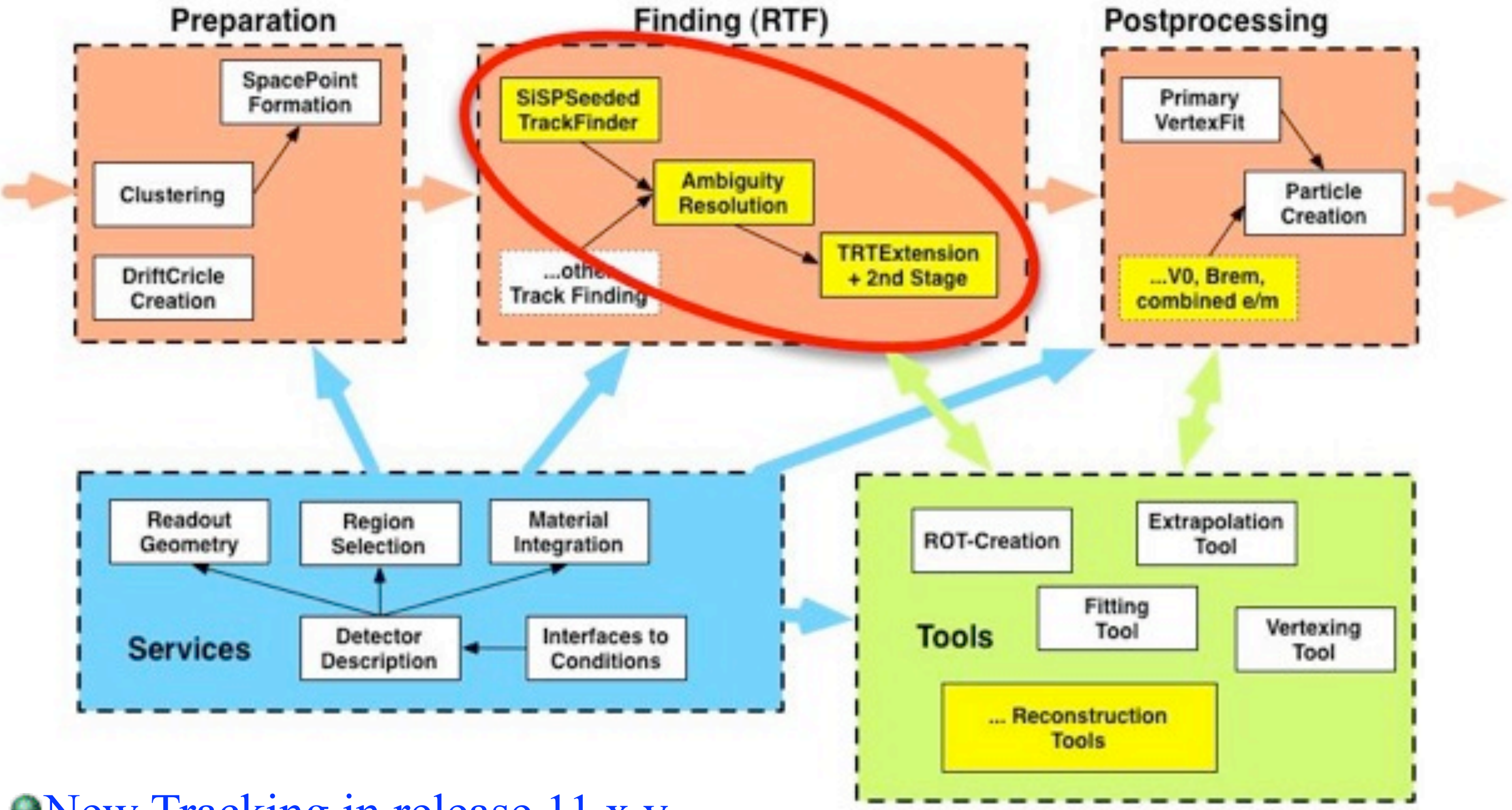
Common Tracking EDM, DD and Tools

- Basis for test beam, cosmics and newTracking reconstruction code
  - Muons joining the game
- Lot of activities for release 11 and afterwards
  - Full tracking geometry in 11.2.0 for first time (Rome initial)
- Migration of standalone packages into new framework progressing
  - Igor has been moving a lot of code from xKalman into Tracking/
  - Waiting for Alan to do so as well (Intersector/iPatFitter first to go)

- Several new techniques are being studied now
  - STEP propagation (propagation in dense material)
  - TrackingGeometry for material corrections
  - Gaussian Sum Filter (electrons)
  - Deterministic Annealing Filter (high lumi TRT reconstruction)

- Recent developments on Vertexing
  - “Classical” Billoir code is available (chi2-fit)
  - New effort on Kalman Filter vertex fitting, robust fitting
  - B-tagging and b-physics still use fortran based codes - migration needed

New Tracking - High Level Design



- New Tracking in release 11.x.y
- Event Filter version and offline version of each Algorithm, but same tools
- Track finding strategy similar to xKalman or iPatRec (inside-out)
- All tools have abstract interfaces, use new EDM, tracking tools, etc.

First Prototype is operational

- Pattern strategy starts from Space Points in Pixels and SCT with an extension into TRT (inside-out tracking)
  - Similar to xKalman and iPatRec

- First prototype consists of 4 algorithms:

SiSPSeededTrackFinder	- track candidate finding in Pixel and SCT
InDetAmbiguitySolver	- select good track candidates, full track fit and resolve ambiguities
TRT_TrackExtension	- extend resolve tracks into TRT
InDetExtensionProcessor	- refit of extensions and replace original

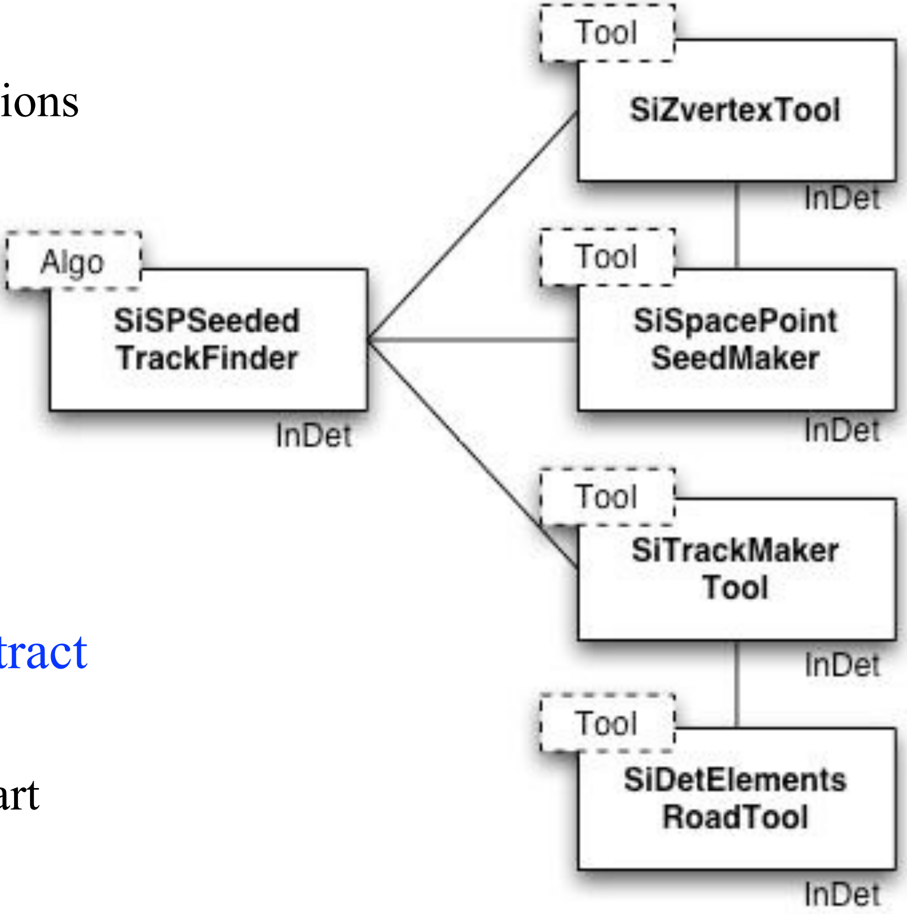
- Prototype covers 3 use-cases
  - Offline, Event Filter and Test Beam

- Basic Functionality in place
  - Many opportunities to contribute still (see later)

Space Point Seeded Track Finder

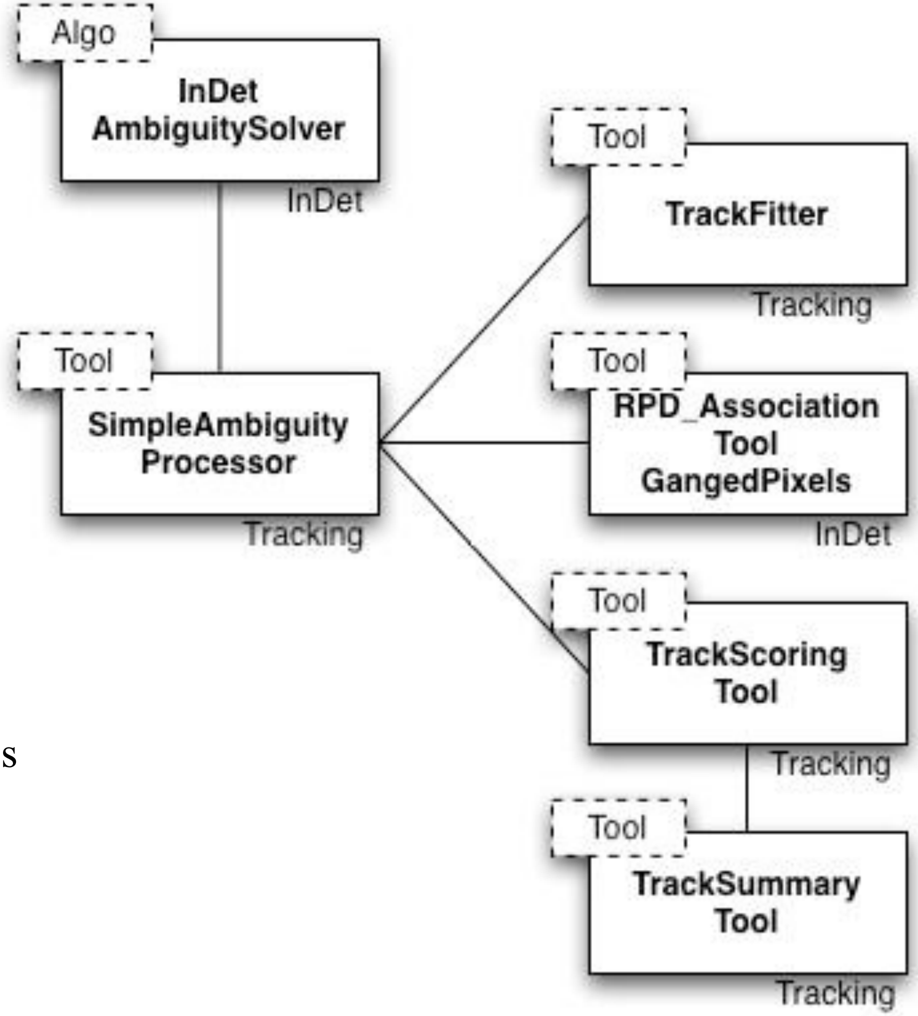
- Track finding in Pixel and SCT
  - Starting from Space Points combinations
  - Baseline: fast Z-vertex scan (optional without)
  - Space Point Seeds pointing back
  - Build up road for each Seed
  - Associate Clusters in all layers
  - Output are TrkTracks

- Implementation using new EDM, abstract interfaces and common tracking tools
  - Region-of-Interest seeded mode is part of interface definition



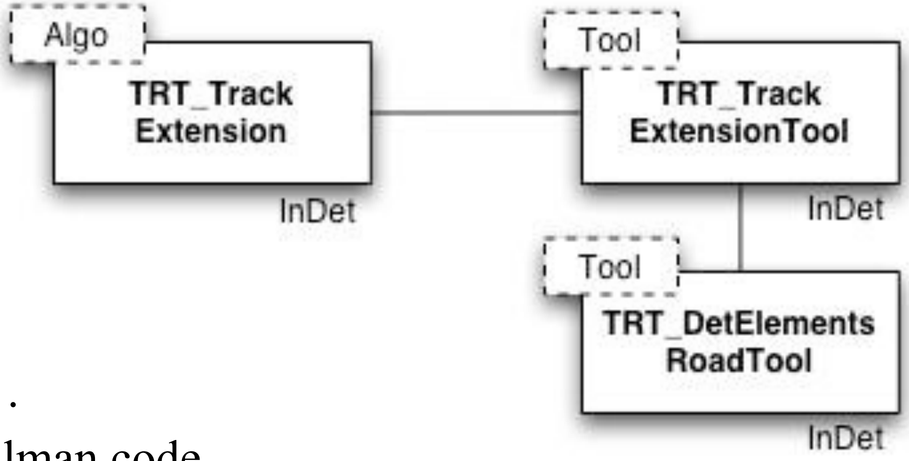
Ambiguity Processor

- Task: resolve track candidates from different input track collections and do final track fit
- "Simple" strategy:
  - Reject double candidates first
  - Score all tracks
    - Number of clusters, holes, χ^2, …
  - Take best, mark PRDs as used
  - Try next best track,
    - if all PRDs are still un-used, take it, mark PRDs as used
    - otherwise drop used PRDs, new score and add new track to candidates
- Takes into account ganged pixels !
- InDetAmbiguitySolver uses InDet and Tracking Tools
  - All have abstract interfaces !!!



TRT Track Extension

- Algorithm to extend Silicon tracks into TRT
  - Runs after ambiguity
  - Build up road in TRT each Si-track
  - Associate Drift Circles in TRT
  - Resolve L-R
  - Output is association list

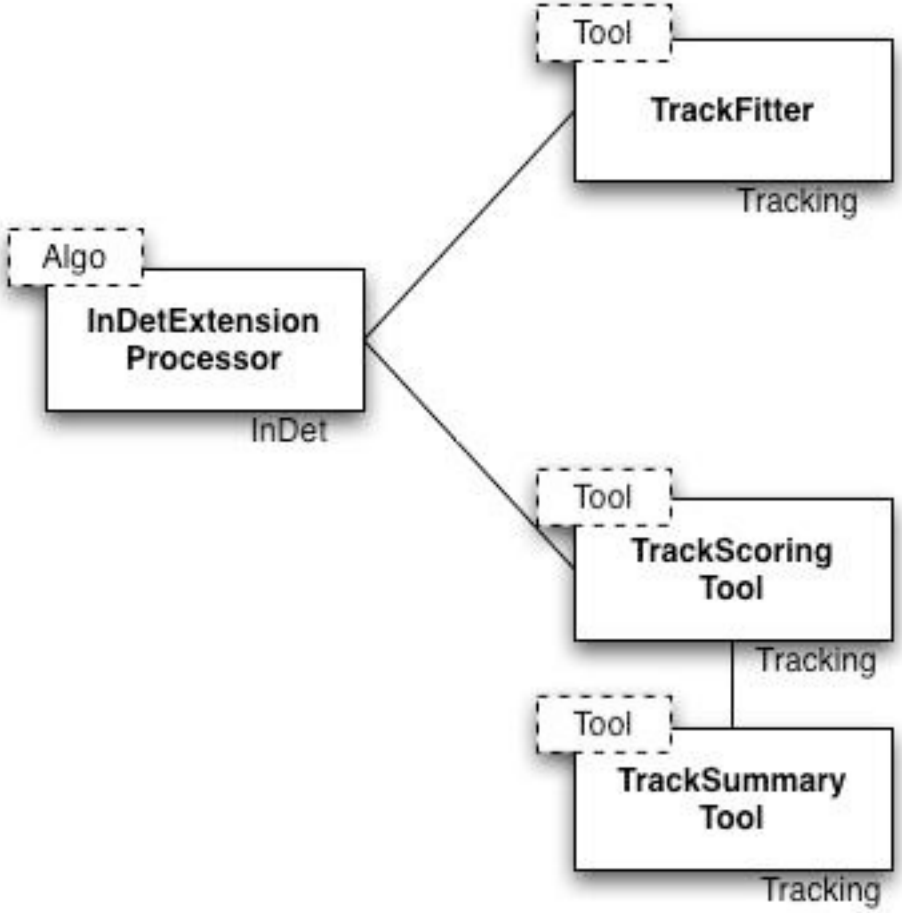


- Same design principles like for the SPSeededTrackFinder
  - Abstract interfaces, EDM, tracking…
  - Igor implemented both based in xKalman code (one finds a lot of tools with "_xk" in cvs)

- Open design principle - see later

Extension Processor

- Strategy:
  - Pattern Recognition provides set of track extensions (into TRT)
  - Refit track + extension
  - Score new and original track
  - If new track is better, replace original

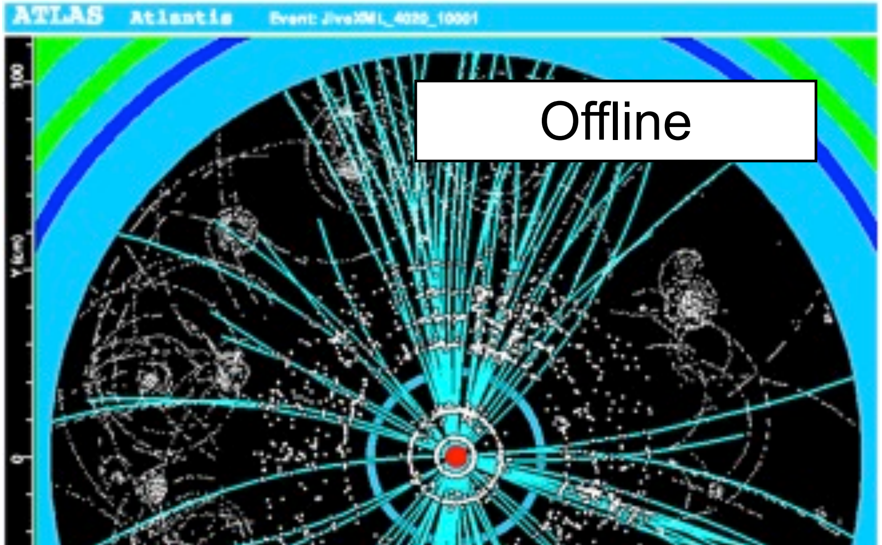


- Uses same scoring function, fitter, … as ambiguity solver

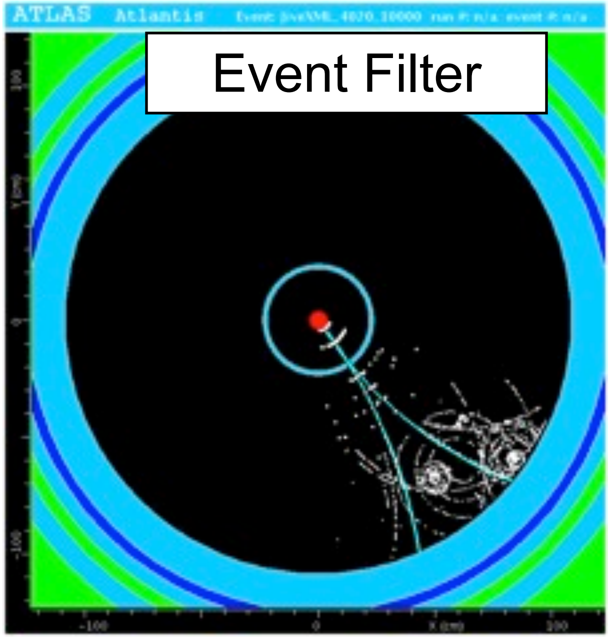
- Comment:
  - Several outliers found in TRT
    - material interactions for electrons…
  - Changed TrkTrack to record those
    - save TR info, input to kink reconstruction in future, …
  - Same problem seen in xKalman and iPatRec
    - Legacy converters have been changed to mark the outliers correctly

Tests of newTracking

- 3 options now in InDetRec_jobOptions.py:
  - xKalman
  - iPatRec (still def.)
  - **newTracking**
- New Tracking is running in offline, CTB and Event Filter

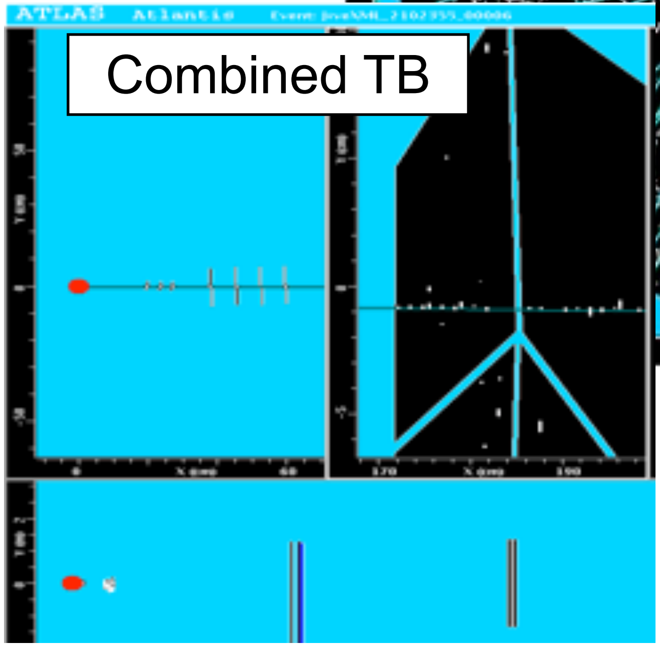


Offline



ATLAS Atlantis Event: JiveXML_4020_10000 run #: n/a event #: n/a
100
Y (cm)
0
-100
X (cm)

Event Filter



Combined TB

Efficiencies, Fakes…

- Using new InDetRecStatistics package
- Total multiplicity per event (P>1 GeV):

	xkalman	iPatRec	newTracking
	63.0	59.7	61.3

- Primary track efficiency / fake rate :

	xkalman	iPatRec	newTracking
Barrel	96% / 4%	96% / 2%	92% / 3%
Transistion	91% / 3%	94% / 2%	91% / 4%
Forward	91% / 3%	94% / 3%	87% / 3%

- Number of hits per track Pixel/SCT/TRT :

	xkalman	iPatRec	newTracking
Barrel	1.9/8.2/26.2	2.0/8.0/29.5	2.0/8.1/31.3
Transistion	1.9/8.1/25.1	2.0/7.9/28.6	2.0/8.1/28.4
Forward	2.0/8.6/17.3	2.0/8.2/19.1	2.0/8.6/18.6

- Work needed on tuning newTracking - still often “best guess”

Timing in recent nightly

- Big effort made to speed up Data Preparation for 11.0.0
  - More than a factor 4 in timing gain compared to 10.0.0
  - Mostly StoreGate, HistoryService, IdentifiableContainer (core framework…)
- Timing of newTracking compared with xKalman/iPatRec ?
  (10 events tth-bb, nightly 11.2.0, lxbuild015)

DataPreparation	~ 599 ms (Pixel 48, SCT 172, TRT 344, SP 35)	
iPatRec	~ 466 ms (excluding truth, stat, legacy-cnv)	
xKalman	~ 608 ms (excluding legacy-cnv)	
SiSPSeededTrackFinder	~ 97 ms	TOTALS: 357 ms (DKF)
InDetAmbiguitySolver	~ 42 ms	
TRT_TrackExtension	~ 45 ms	
InDetExtensionProcessor	~ 173 ms	
Truth	~ 479 ms (PRD 473, SP 2, Trk 4, …)	
PostProcessing	~ 180 ms (VX 12, PC 118, Stat 50)	

- Timing overheads: POOL, EDM (set methods…), Truth (!)
- Overall Inner Detector software is factors faster than LAr (!!!!)
  - We have done our homework, use good timing to add more functionality

Complete Functionality: recently

- Ongoing work on new python configuration scripts
  - Need to make full use of modular reconstruction
  - Hence, replace fitter everywhere in a transparent way

- Statistics package produces summary table at end of job + RTT input ntuple
  - Plus plenty of other activities on validation…

- Shared hits, holes on track search
  - Help from the b-tagging people, both code and validation

- Usage of Beam Spot from ConDB
  - Used in track finding and in primary vertex reconstruction

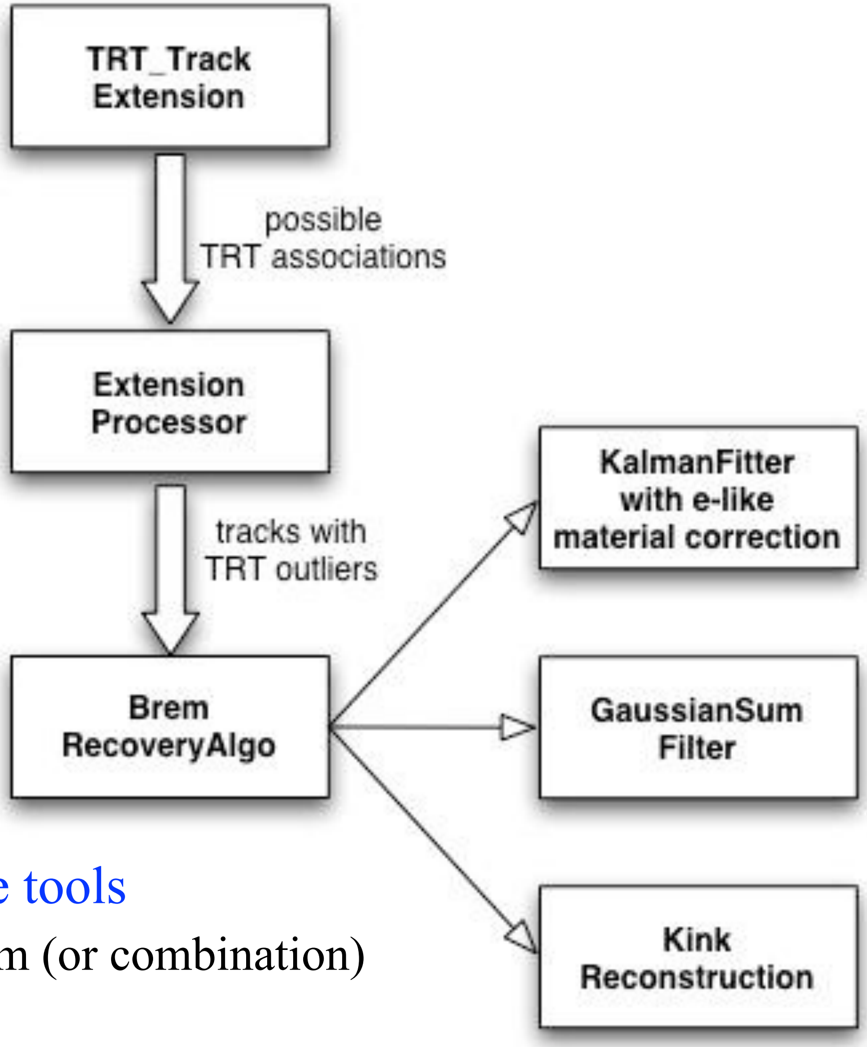
- Changes in KalmanFilter and Extrapolation
  - Speed, storage of outlier, memory management

- Completing Tracking Geometry for Material corrections
  - First full version in 11.2.0, to be validated

Complete Functionality: Brem.-Recovery

- Electron tracks have a lot of outliers in TRT in track fits
  - EDM changed to flag those in ExtensionProcessor (and in xKalman/iPatRec)

- 3 tools available in CVS
  - Track fitter with electron material corr.
  - Gaussian Sum Filter
  - Prototype Kink reconstruction

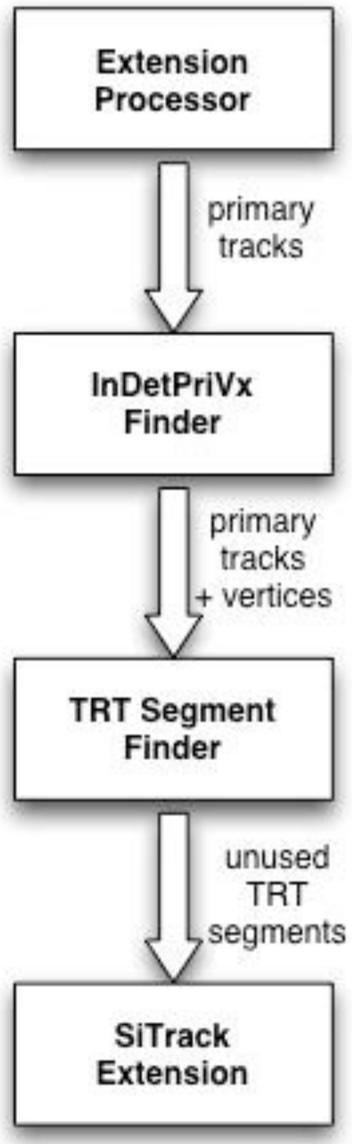


- Need an ID electron algorithm using the tools
  - Should have option to use either of them (or combination)
  - Study what works best and when
  - Opportunity to contribute, strong link to e/gamma

Complete Functionality: TRT seeded rec.

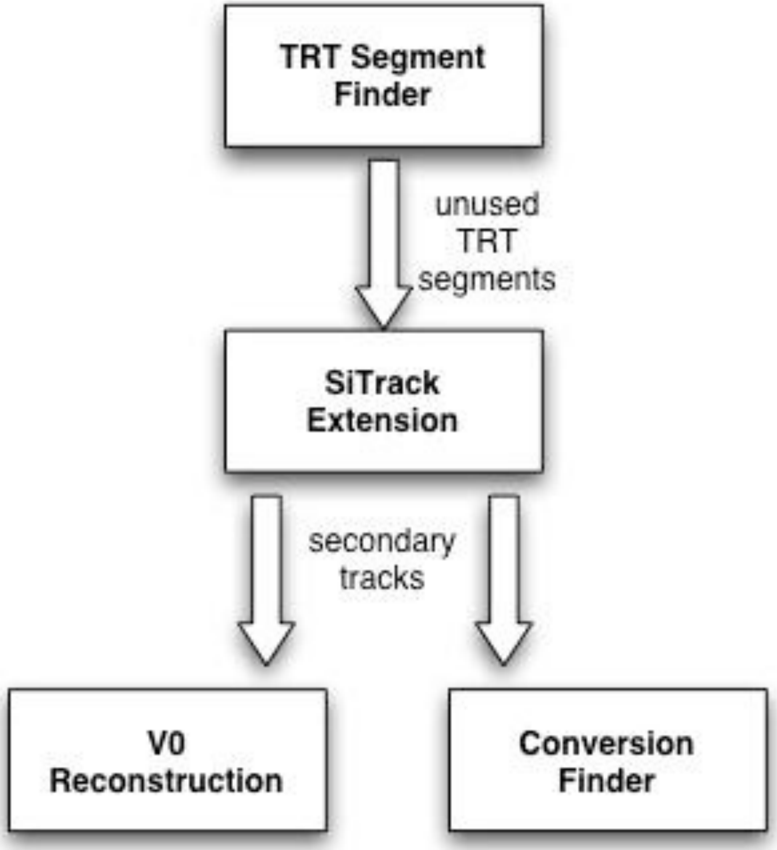
- Several use-cases for TRT seeded reconstruction
  - Dedicated tracking for secondary particles
  - Fallback option for primary track finding
  - Test beam and comics reconstruction
  - TRT segments for calibration/alignment
  - Level-2 TRT track trigger

- Igor is advanced with code
  - First do the secondaries
  - Will need to reshuffle post processing



Complete Functionality: V0, conversions

- Use tracks from TRT seeded tracking
- Develop 2 new packages ongoing
  - V0 reconstruction
  - Conversion finder
- Existing code for both is in CVS
  - Different level of sophistication
  - New packages being worked on
  - Strong link with e/gamma and b-physics
- Complementary:
  - work ongoing for late conversions in TRT



TRT Segment Finder

unused TRT segments

SiTrack Extension

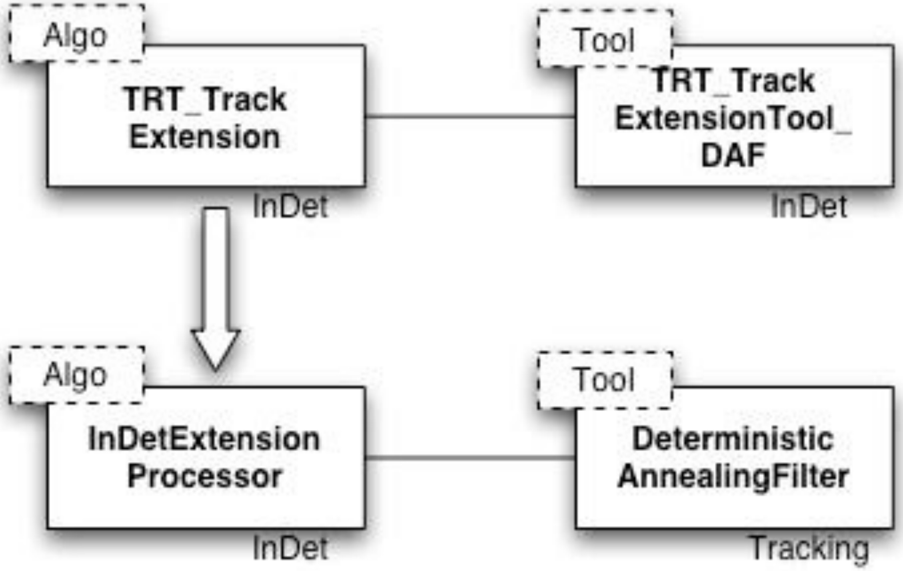
secondary tracks

V0 Reconstruction

Conversion Finder

Complete Functionality: high lumi TRT

- Possible way to improve reconstruction in TRT for high lumi



- Deterministic Annealing Filter
  - High occupancy in TRT
  - DAF freezes out TRT tracks from clouds of points
  - thermodynamic picture of track finding

- DAF available in common tracking area
  - Alternative TRT_TrackExtensionTool in CVS
    - record all DriftCircles near track
  - DAF as fitter in ExtensionProcessor
    - Just a JobOption change

Tracking Milestones and Plans

- Release 11.0.4 - newTracking used for test productions
- Release 11.3.0 - newTracking is default in RecExCommon
- Release 12.0.0:
  - newTracking “inside-out” is production quality for Offline+Event Filter
    - Using fully functional TrackingGeometry
    - newTracking works for Cosmics reconstruction, alongside CTB_Tracking
  - Prototypes: - TRT-seeded tracking for V0, conversions, secondaries
- V0, kink and conversion finding
- package to use electron refitting (GSF/Kalman)
- DAF extension into TRT
  - Intersector and iPatFitter in /Tracking
- Release 13.0.0:
  - newTracking with 2nd pass TRT-seeded track finding is default
  - V0, conversion and electron refitting packages fully operational
  - iPatRec track finding tools in /InnerDetector for testing
- Release 14.0.0:
  - Final decision on track finding strategies for first beam

B-Tagging Overview

B-Tagging Framework on AOD

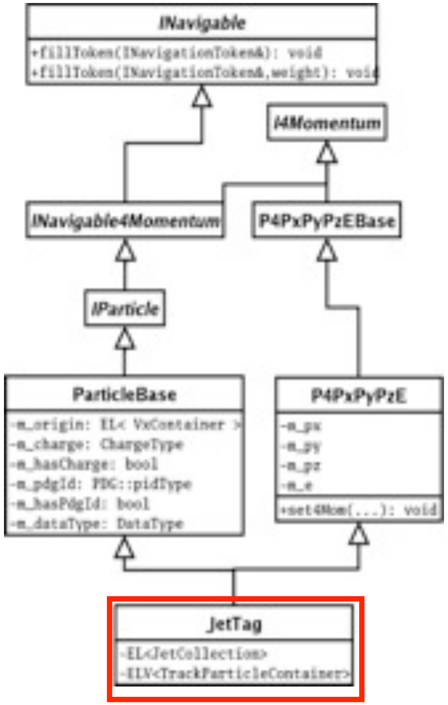
- Goal was to develop a flexible b-tagging framework
  - Easy inclusion of new tagging routines
  - Flexible usage, combination of different taggers
  - B-tagging EDM is part of AOD, poses design requirements

- Framework was starting point for b-tagging developments in ATHENA
  - In absence of any b-tagging in ATHENA Fredrik and Andi started to write some simple taggers to get going on AOD analysis
  - Later migration of Marseille tagging package to ATHENA and inclusion in tagging framework
  - Since then developments progressed in parallel

- See next talk for performance comparisons, results, etc.
  - Just some overview in the following from talks given in Paris

b-tagging EDM

- the b-tagging EDM is way more complex than the one for vertexing!
  - a varying amount of different taggers can be used
  - every tagger produces a likelihood/weight and lots of individual information
  - tag results of different tools have to be "recombinable" without retagging
  - bjet is an "AOD object" and therefore has to behave like one



INavigable
+fillToken(INavigationToken&): void
+fillToken(INavigationToken&,weight): void

I4Momentum

INavigable4Momentum

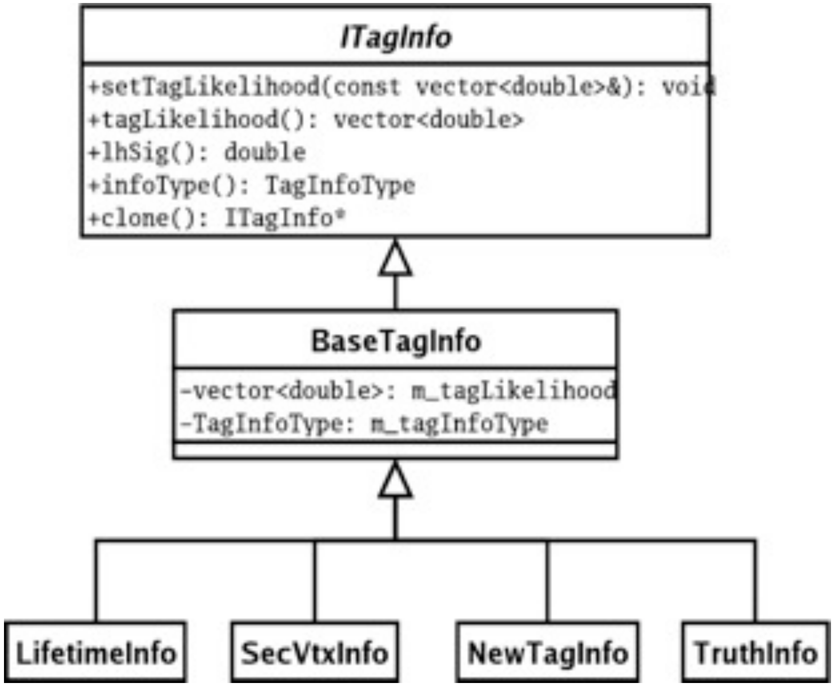
P4PxPyPzEBase

IParticle

ParticleBase
-m_origin: EL< VxContainer >
-m_charge: ChargeType
-m_hasCharge: bool
-m_pdgId: PDG::pidType
-m_hasPdgId: bool
-m_dataType: DataType

P4PxPyPzE
-m_px
-m_py
-m_pz
-m_e
+set4Mom(...): void

JetTag
-EL<JetCollection>
-ELV<TrackParticleContainer>



ITagInfo
+setTagLikelihood(const vector<double>&): void
+tagLikelihood(): vector<double>
+lhSig(): double
+infoType(): TagInfoType
+clone(): ITagInfo*

BaseTagInfo
-vector<double>: m_tagLikelihood
-TagInfoType: m_tagInfoType

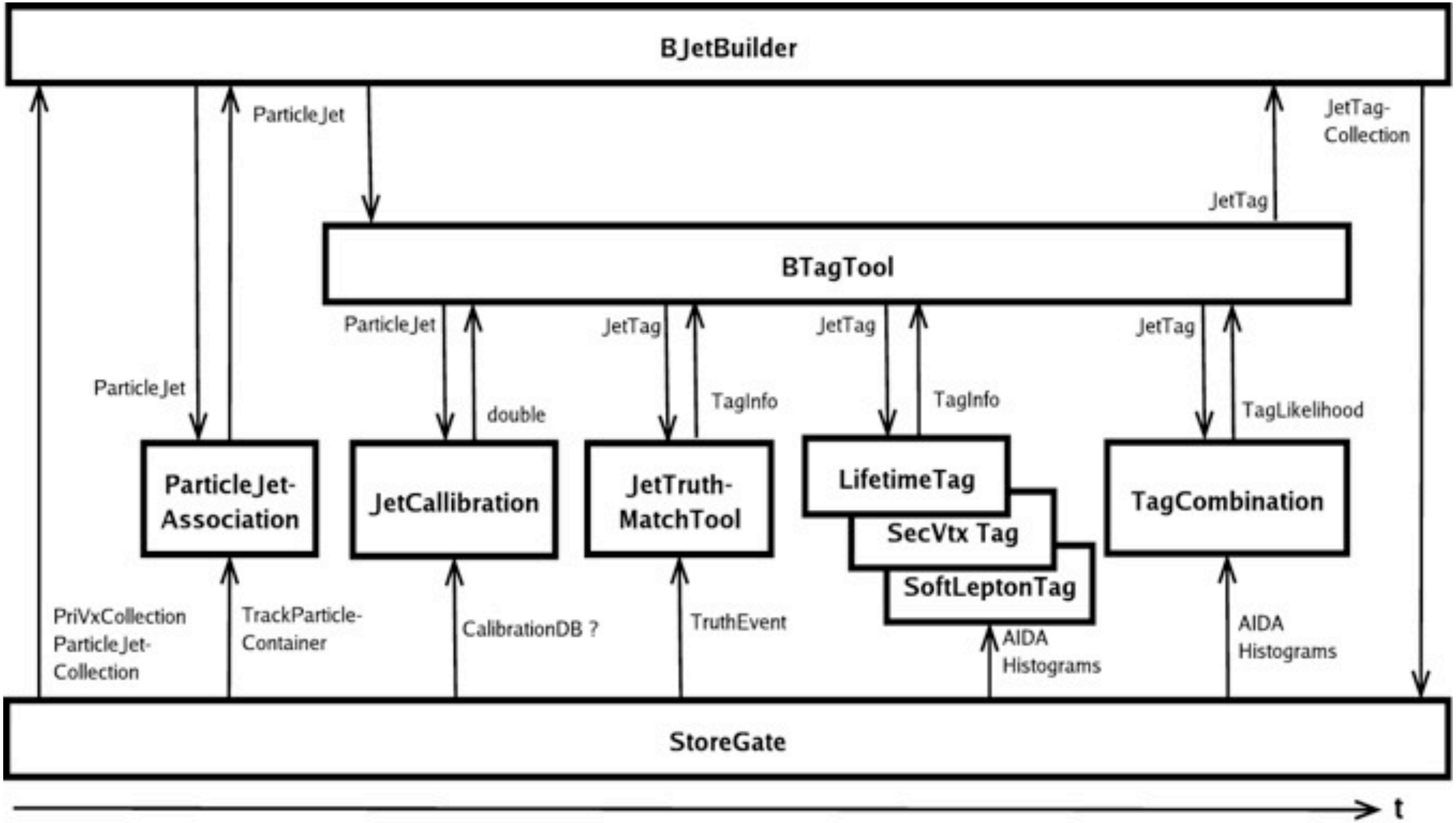
LifetimeInfo

SecVtxInfo

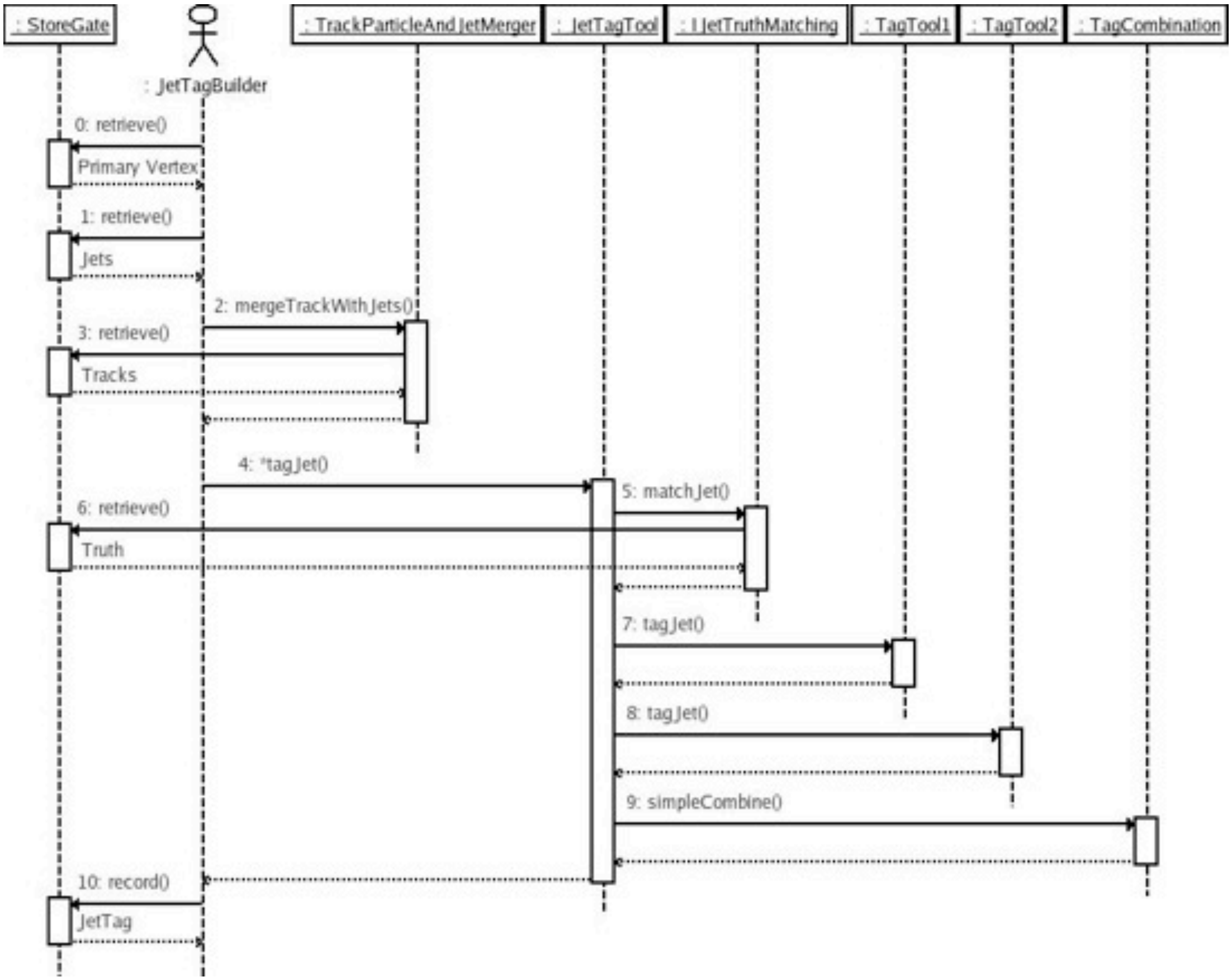
NewTagInfo

TruthInfo

b-tagging flow



JetTagging Sequence Diagram



Software status

A number of redundant taggers are available in AODs :
basically two streams with similar taggers wrt the discriminating variables

	1st stream (AOD and	2nd stream (AOD only)	
IP (long. impact)		Lifetime1D	needs good primary
(trans. impact)	IP2D	Lifetime2D	the most robust
	IP3D	Lifetime3D	combination
Inclusive Secondary Vertex	SV1	SecVtxBU	more demanding for tracking
	SV2	SecVtxTD	

The Soft Electron Tag should be soon available
(only the implementation in AOD is missing)

✓ All taggers are kept for performance studies and cross-checks

⇒ low performance taggers (Lifetime2D/IP2D) are usually rather robust
(easier to understand and commission)

⇒ high performance ones (SV1/SV2) will require more time to control

⇒ taggers identical wrt discriminating variables
(Lifetime2D ~ IP2D, Lifetime3D ~ IP3D)
are kept for cross-checks and do differ in some point
(refined track selection in IPxD,
one 2D vs one 1D pdf for IP3D vs Lifetime3D, …)

✓ For physics analysis, a combination is given :

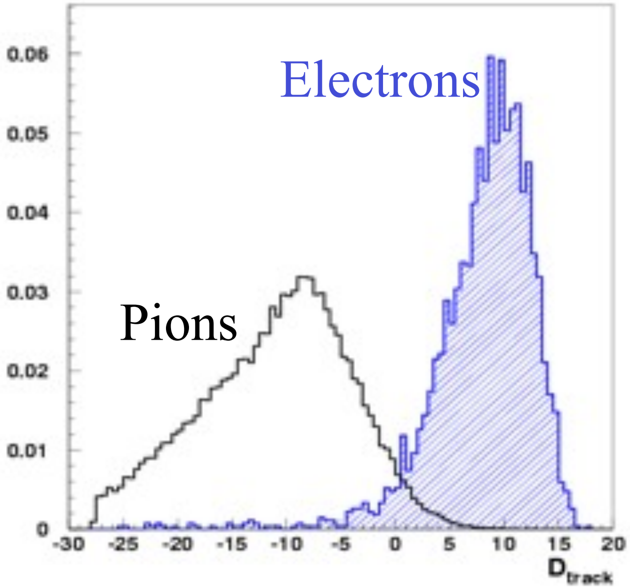
- "1st stream" taggers : (*JetTag)→weight()
corresponding to SV1+IP3D

The most powerful tagger

- "2nd stream" taggers : (*JetTag)→weightForTag("lhSig")
corresponding to Lifetime1D+Lifetime2D(+SecVtxBU)

✓ Soft Electron Tagging :

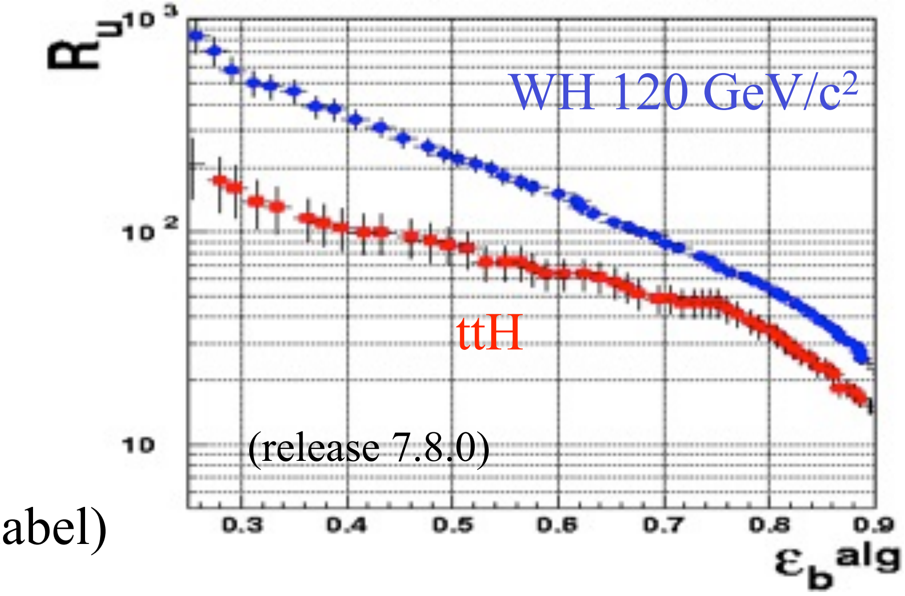
use Soft Electron identification variables to build a probability for each track in a jet
⇒ the track with the highest probability is the "electron candidate"



⇒ light jet rejection vs *algorithm* efficiency :

@ 60% *algorithm* efficiency
(i.e. 0.6*BR(b→eX) ~7.8% b-jet efficiency)

R_u = 151±11 (WH events)
64±11 (ttH events, caveat : mislabel)



Calibration functions, tagging efficiency and light jet rejection determination on data

The most important but least studied topics !

✓ Experience from past or running experiments :

Tevatron ⇒ Space b-tagging cross calibrated from Soft Lepton tagging

* select relatively easily pure b/c from electron/muon tag

* correct for c contamination and for purely hadronic decays with MC

⇒ a Soft Muon Tag is urgently needed !

(should be easier than a Soft Electron Tag)

Use negative impact parameter (à la LEP) :

at zero order, ~ resolution effect, identical for b- and light-jets

⇒ build a tag using a measure of the compatibility of the tracks with the primary vertex

Tagging Algorithm Developments

- Jean-Baptiste already mentioned several aspects…

- Study of calibration for different physics analysis use-cases

- Soft-Lepton tagging - some work ongoing
  - Need to contribute to soft electron/muon identification first (common tracking)
  - Then turn it into a b-tagging algorithm

- Alternative tagging algorithms
  - a la ALEPH/DELPHI using negative IP for auto-calibration
  - SLD inclusive vertex b-tagging, inclusive B vertex chain tagging

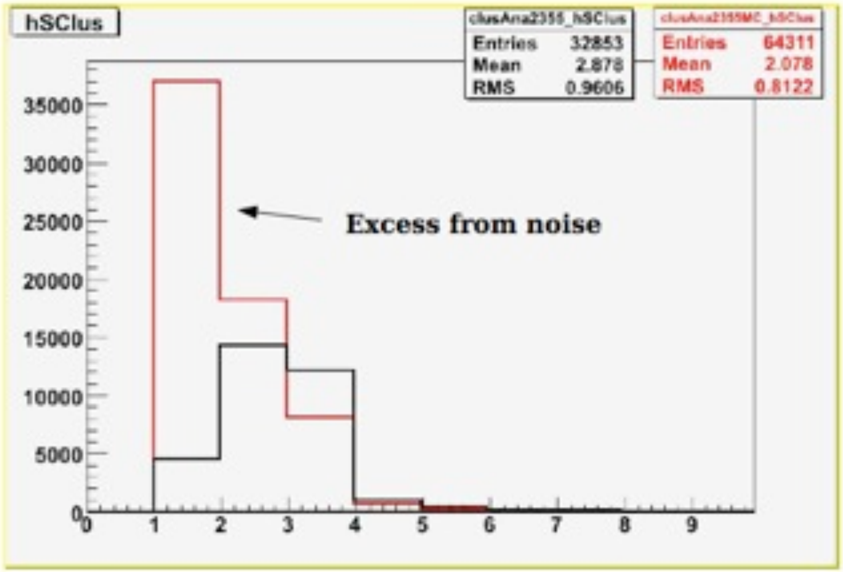
- Trigger b-tagging studies
  - Event Filter supposed to use offline code (seeded?)
  - Currently Inner Detector only covers e/gamma-slice in Event Filter
  - Level-2 code existing
  - Little activity on full slice studies with realistic menus

Pixel Software, Tracking and B-Tagging

Tuning of Clustering

- Time-over-Threshold Clustering enabled since 11.0.0 in CTB and offline
- Cluster errors unchanged, still pitch/sqrt(12)
- ToT clustering studied mostly using Standalone Test Beam data
  - Discussion ongoing how to calibrate individual pixels, if needed
- Cluster errors and position corrections should depend on eta/phi
  - Corrections at time of track fitting missing
- Simulation/Digitization does not reproduce Combined Test Beam data
  - Control over G4 parameters
  - Digitization model, parameters
  - Noise…
- General pattern in Inner Detector software
  - Standalone Test Beam software well tuned
  - Not migrated into ATHENA

Cluster sizes...

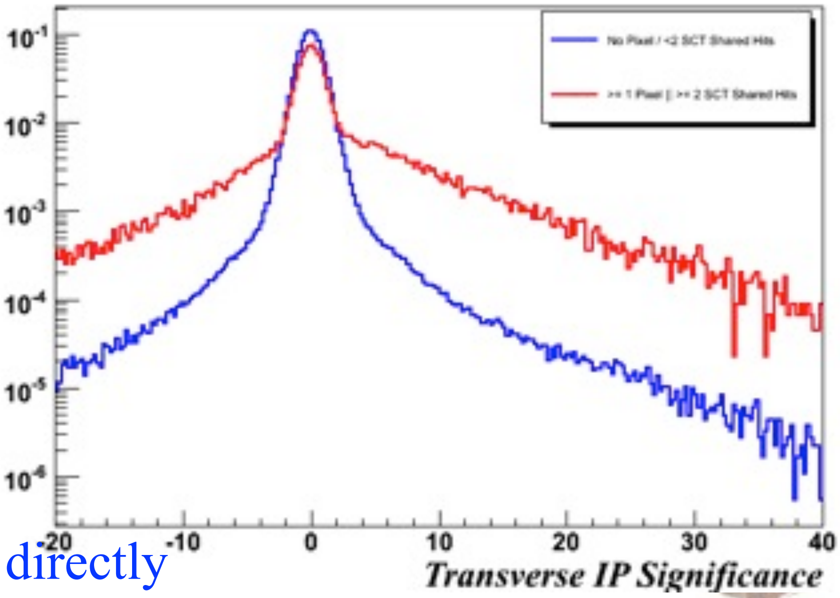


Id software meeting, 13/12/2005, CERN

Shared Hits in Pixels (and SCT)

- Dense tracking environment in jets may lead to large/shared clusters
  - See studies in Physics/Detector TDR on this effect
  - Handling of such situations very different in tracking codes
    - iPatRec - so called “wide” (==shared) and “narror” cluster width used in fit
    - xKalman - allow for shared hit if quality criteria of track passed after fit
    - newTracking - use ambiguity processor to decide on association
  - All seem to give comparable results, optimal solution ?

- B-Tagging now has special treatment of shared hit tracks
  - Significant tails seen in light jets
  - Different track classes
  - 10-20% gain in rejection @ same effi.



- Goal should be:
  Study how to fix tracks in reconstruction directly

Re-clustering of shared Pixel Clusters

- xKalman at time of Physics TDR (Fortran)
  - Try to re-cluster shared pixel clusters to improve
  - Functionality missing in ATHENA

- Basic idea:
  - (large) clusters may match 2 tracks in event
  - Try to split cluster using this knowledge in 2nd stage clustering
  - Reassign split clusters to tracks and refit them
  - Use b-tagging to study effects and tune re-clustering

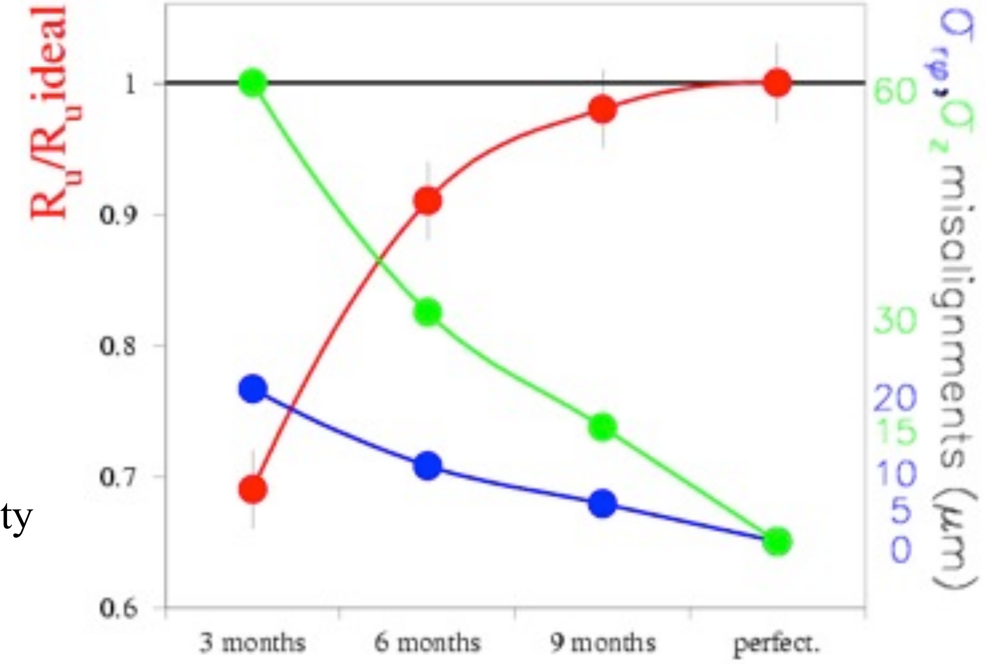
- Side-remark:
  - DAF/MTF might be an option for very high Pt b-jets
  - Some studies (CMS) in literature

Ganged Pixel Ambiguities

- Simulation of so called Ganged Pixel ambiguities has been introduced to software about a year ago
  - Ganged Pixel readout handled in Digitization and in Clustering
  - Clusters are flagged for ganged Pixel hits
  - Pointer to mirror ambiguity recorded in EDM
- NewTracking first package trying to resolve explicitly these cases
  - Relatively frequent in jets, may give rise to impact parameter offsets
- Observations:
  - Ganged ambiguities in B-layer difficult (no interpolation from surrounding layers)
  - Chance to get it wrong in B-layer therefore seems higher… (NewTracking/iPatRec)
- Need to properly study this effect and how to handle it in reconstruction
  - Make use of cluster size (mirror should mostly be 1 pixel cluster) ?
  - Best strategy in reconstruction ? Or record both solutions for the track ?
  - How to handle such tracks in b-tagging code ?

Pixel Misalignment

- Effect of misalignment on b-tagging
  - ttH/tt events, random smearing
- Misalignment is not random !
  - Module is a solid object, track correlations are given by geometry
  - Misalignment is between modules !
- Misalignment has 2 effects
  - Resolution effect, linear with uncertainty
  - Effect on pattern recognition may be non-linear due to instabilities



- Need to introduce misalignment in simlation/reconstruction + repeat study
  - DC3 should have misalignmed Pixel modules - next slide
  - Effect is ~ correct for a given event, but alignment package would easily correct it back
- Proposal: implement “Pixel shaking” scheme
  - New misalignment parameters for each event (a la DELPHI)
  - Misalignment correctly reproduces residual alignment uncertainties on real data
  - Can be implemented within existing GeoModel modifying alignment transformations

Pixel residual misalignment in DC3

Final misalignment

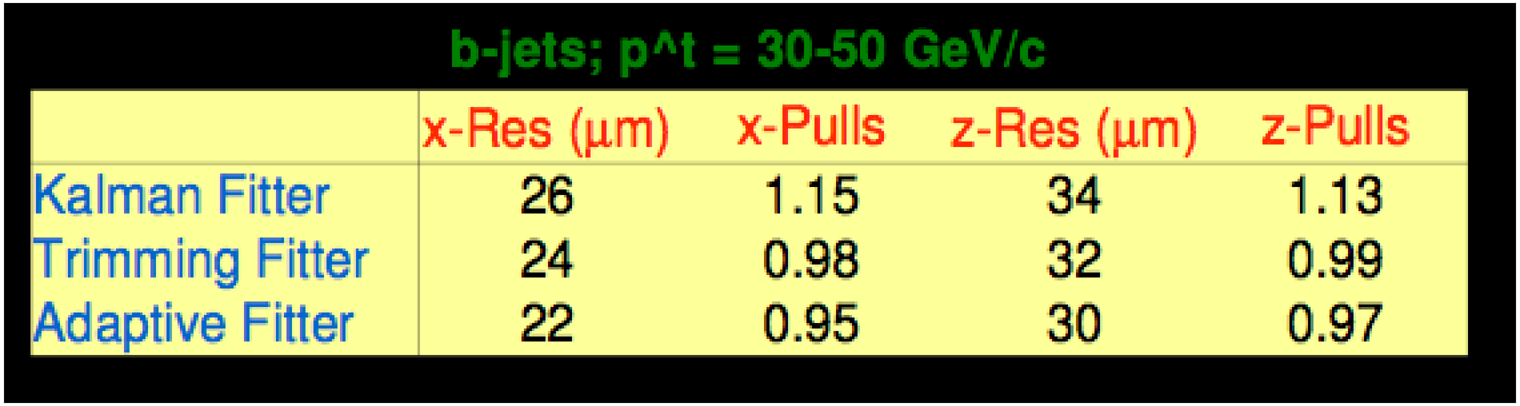
	phi	Z	R	tilt	slope
Modules	5 µm	15 µm	15 µm	0.02 deg	0.01 deg
Barrel/disk	3 µm	10 µm	10 µm	0.01 deg	0.005 deg
All pixel	3 µm	10 µm	10 µm	0.01 deg	0.005 deg

- A little more optimistic than Pixel or ID TDR (7 µm, 20 µm, 20 µm)
- Pessemistic to respect to the requirements for W-mass measurement with 15 MeV (1 µm, 10 µm)

Vertexing in Reconstruction

- Common vertexing use-cases for Reconstruction and b-tagging:
  - Primary vertex finding
  - V0, conversion and hadronic interactions
- Currently primary vertexing is using the Billoir tools
- Some work ongoing on V0, conversions and interactions for new tracking, but far from ready + complete

- Both, the reconstruction algorithms and the tools need further development
  - Lack of complete vertexing suite in ATLAS
  - See CMS studies using e.g. Gaussian Sum or Annealing techniques



b-jets; p^t = 30-50 GeV/c

	x-Res (µm)	x-Pulls	z-Res (µm)	z-Pulls
Kalman Fitter	26	1.15	34	1.13
Trimming Fitter	24	0.98	32	0.99
Adaptive Fitter	22	0.95	30	0.97

Outlook and Discussion