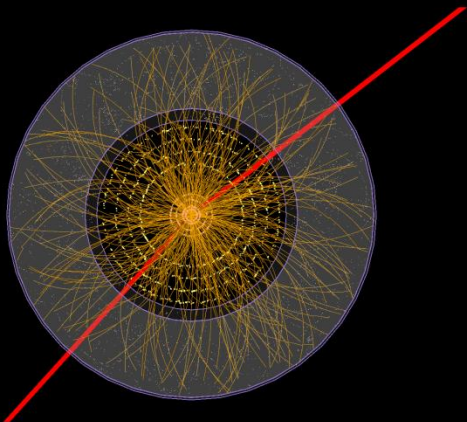


14th Workshop of the tautau Analysis Working Group  
DESY, Hamburg  
November 16, 2015

# ATLAS Tau (Offline) Analysis Framework

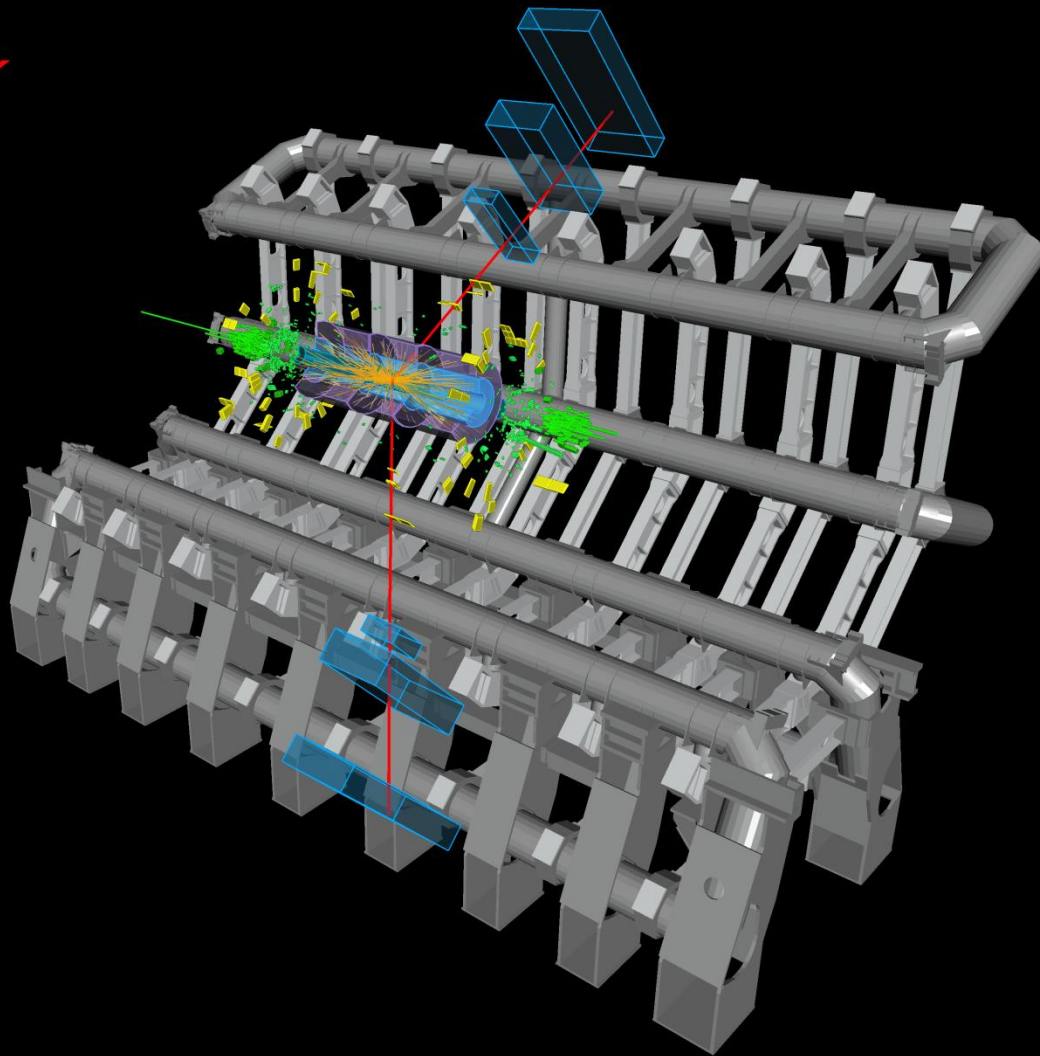
Z. Zinonos



$M_Z = 90.2 \text{ GeV}$



Run: 267638  
Event: 242090708  
2015-06-14 01:01:14 CEST



# Overview

- ❑ Data structure in Run2 – xAOD
- ❑ Data analysis model in Run2 – EDM
- ❑ Analysis frameworks
- ❑ xTauFramework
- ❑ Tau analysis tools



# The xAOD Data Format

## Event Data Model

A collection of classes — interfaces and concrete types — and their relationship which, together, provide a representation of an event detected by ATLAS and eases its manipulation by the reconstruction and analysis developers

## Or in plain words

How “electrons”, “muons”, “jets”, “taus” etc. are represented in computer memory, how they are stored on disk, and how we use them

## xAOD

**Replacement for both  $A_{\text{analysis}}O_{\text{bject}}D_{\text{ata}}$  and  $D_{\text{erived}}P_{\text{hysics}}D_{\text{ata}}$  data for RunII**

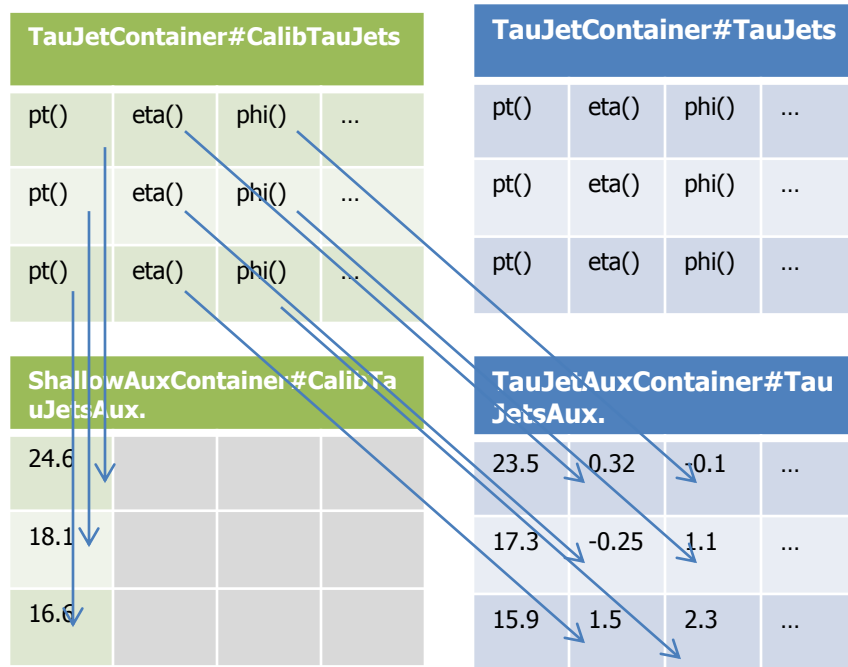
# General xAOD Features

## Double-layer / objects are split in two:

- **Interface objects:** These are objects that provide a usable UI for the type, but don't hold on to any persistent data
- **Payload objects ("auxiliary data store"):** These are relatively dumb objects that just allocate continuous memory for the data, and allow the interface objects to access this data.

## This split means that:

- Data storage technology is independent of the interface that we provide, for instance for *taus*.
- We can partially read information about objects during analysis => memory savings
- Any "regular" code should only ever create auxiliary objects (when creating new xAOD objects), but never manipulate them directly



One `xAOD::ShallowCopyContainer` for every variation  
+ only modified elements are stored  
+ save disk space and require less CPU/memory resources

# How it looks like

Event data is always in a TTree called “CollectionTree”

The interface container (DataVector<xAOD::Jet> in this case) doesn’t hold any payload

All containers of name “Bla” are accompanied by an auxiliary store with name “BlaAux.”

The auxiliary store can hold variables that:

- were already defined at compile time
- were only created at runtime (dynamically)

# ATLAS Data Analysis Model in RunII

## ATHENA analysis

- Athena GAUDI framework
- *Python* + *C++*, compiled with cmt,
- using the framework to loop over events: heavy but powerful
- → detector simulation, reconstruction, 25% of end user analysis
- In RunI producing or using AOD format → RunII: xAOD format
- → Common data format and analysis tool packages for both types of analysis frameworks

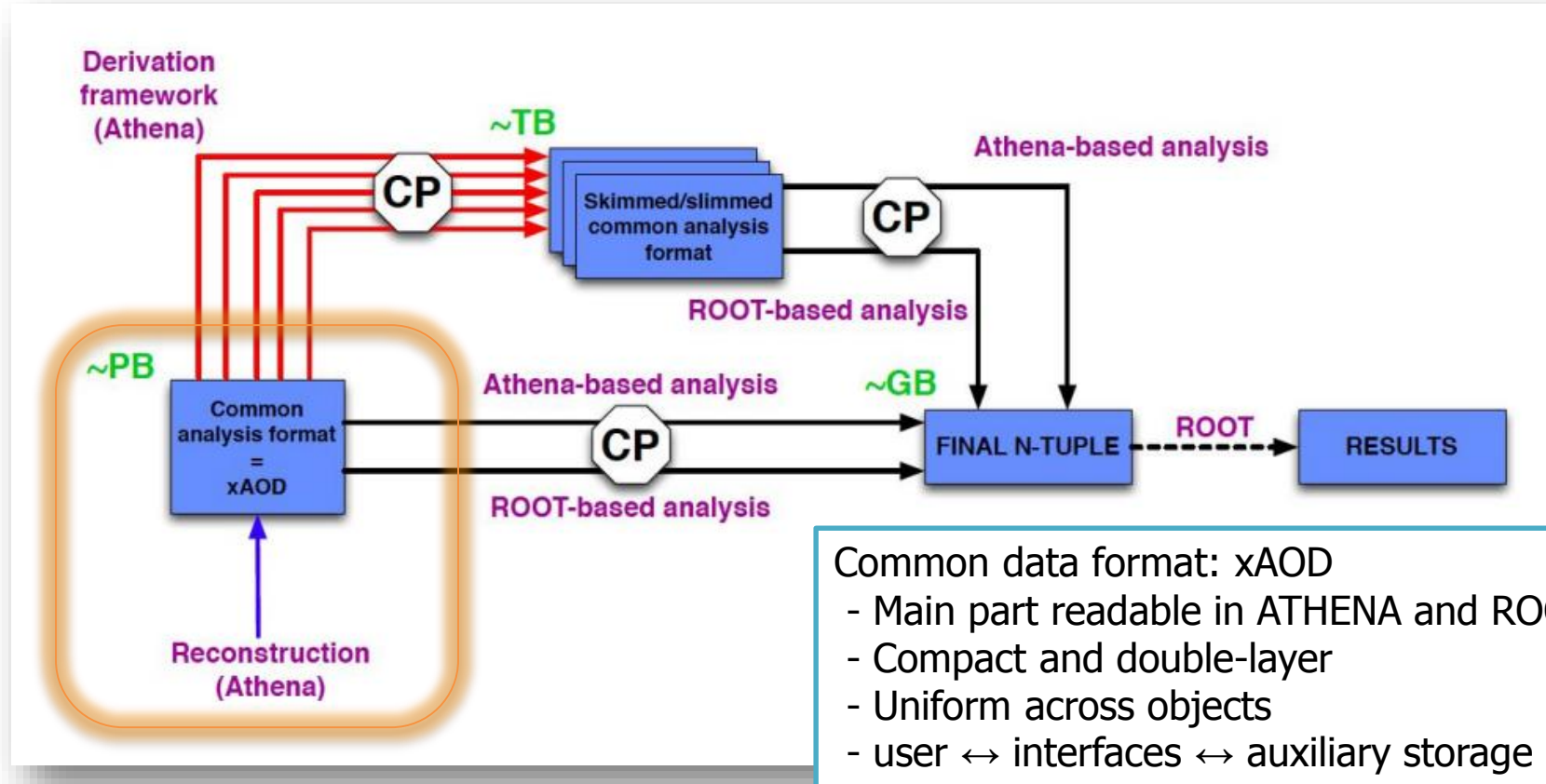
## ROOT analysis

- compilation for example with RootCore
- using e.g. *MakeClass* method or *EventLoop* package to create loop over events:
- lighter, faster, less powerful
- → 75% of end user analyses in ATLAS
- In RunI using ntuple format → RunII: xAOD format !

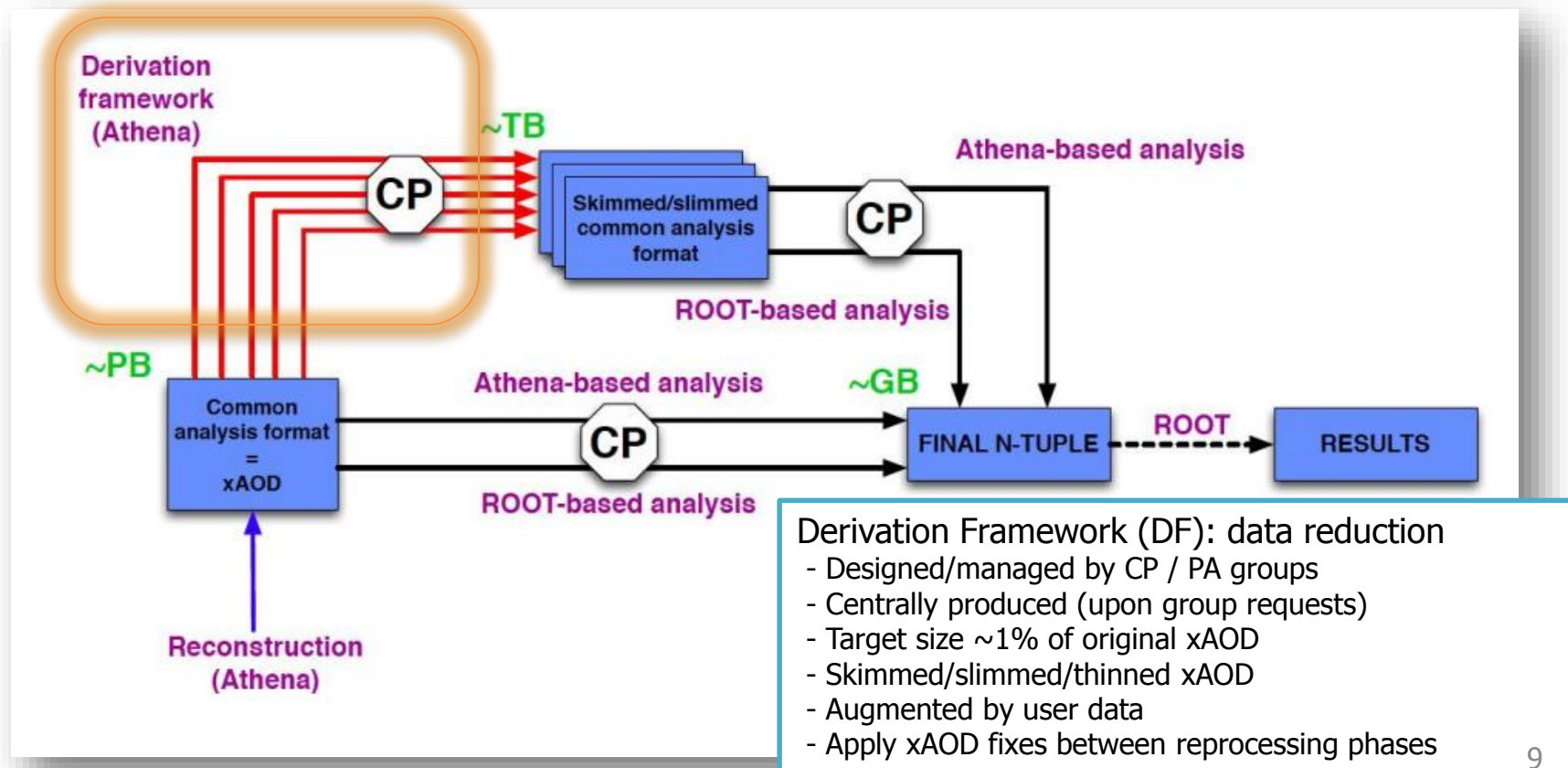
## RootCore

- package that helps developers to build packages that work standalone (outside ATHENA).
- it works both for packages that are ROOT-only and for packages that work with both Athena and ROOT

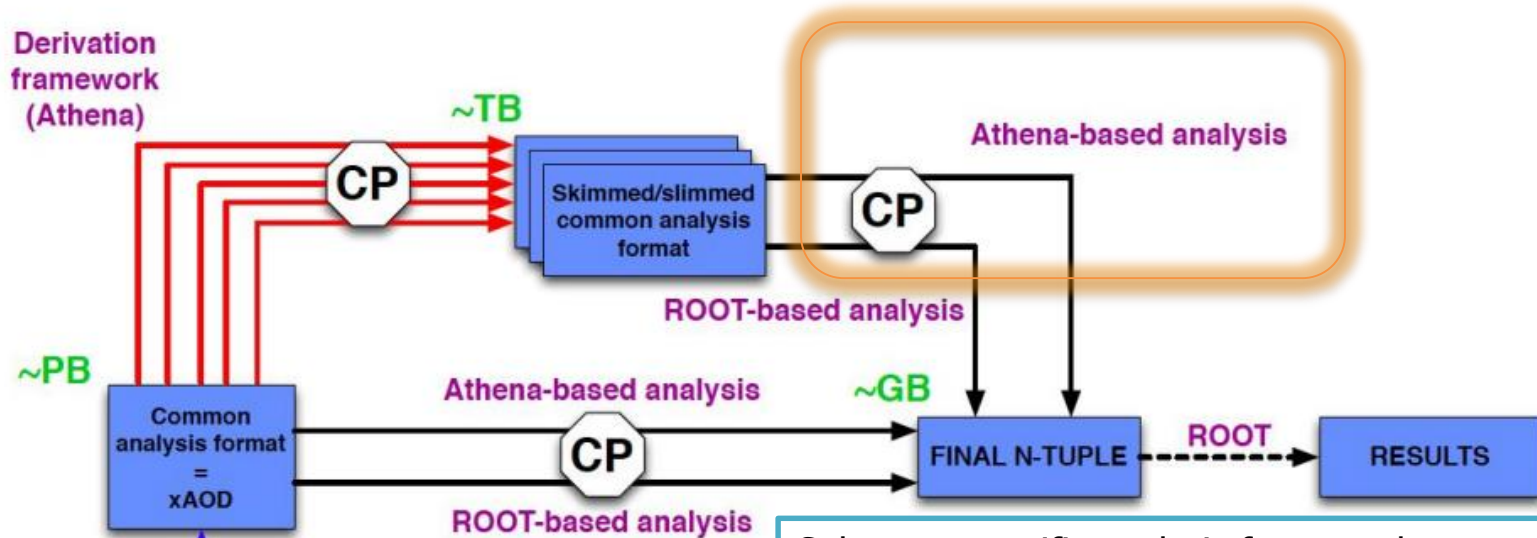
# RunII Analysis Model: common data format



# RunII Analysis Model: common data format



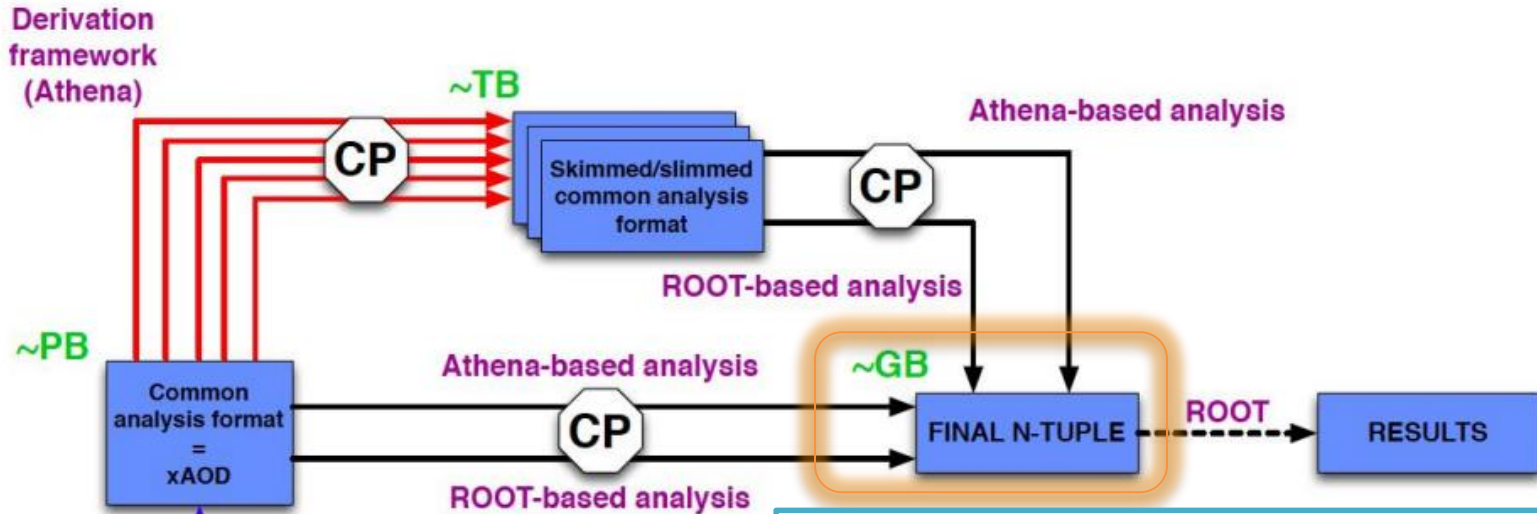
# RunII Analysis Model: common data format



## Subgroup specific analysis frameworks

- ATHENA or ROOT (or PyRoot) based
- Using versioned Analysis Release (weekly basis)
- Dual-use Combined Performance tools:
  - Same tools as in derivation framework
  - Common interface, operate on objects

# RunII Analysis Model: common data format



Final Ntuple:

- Tiny fraction of the original dataset
- Calibrated and corrected data
  - Mini xAOD (  $\rightarrow$  CP tools can be re-applied)
  - Mini ROOT ntuple

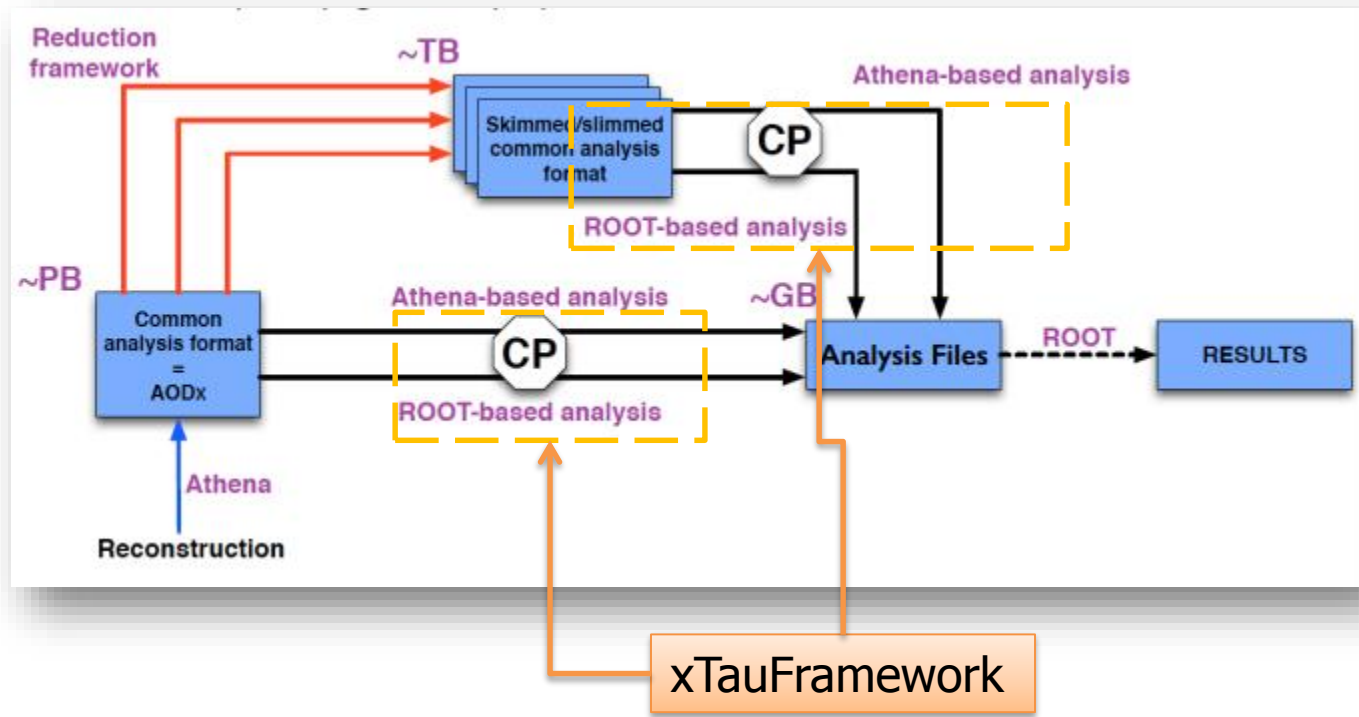
# Analysis Frameworks

- ✓ AnalysisSUSY(aka SUSYTools)
- ✓ AnalysisTop
- ✓ QuickAna
- ✓ HWW xAOD Framework
- ✓ CxAOD Framework (Hbb)
- ✓ xTauFramework (H $\tau\tau$ ,  $\tau$  WG,  $\tau$  trigger)

# xTauFramework

## Blood test

- RootCore-based
- EventLoop structure
- C++11
- Bidirectional : primary or derived xAODs
- Prepares mini ntuples for final plotting analysis!
- Calibrates and corrects objects
- Reconstructs analysis observables
- Applies systematic uncertainties
- Applies CP recommendations



# Main body in a nutshell

<---- persistent -----> <----- transient -----> <---- persistent ----->

input  
xAOD (PB)  
DxAOD (TB)

constructor

setupJob()  
parse job options

node submission  
grid , local, batch

fileExecute()

execute()  
systematics loop  
calibration, corrections, ...

initialize()  
read event, bookkeeping,  
class declarations, CP  
tools configuration ...

changeInput()

histInitialize()  
reparse job options

fileExecute()  
changeInput()

finalize()  
fill weight histos

histFinalize()  
free allocated memory

destructor

## Typical Performance

- HIGG4D2 derivations
- No  $\pi\pi$  mass reco'n
- ~15s wall time spent on memory load and tools initialization
- **~120 Hz/event** processing (64bit)
- 50 GeV disk occ. full 2015 13 TeV data & MC

EventLoop-based

Shallow copies  
- fast operations

VIEW\_ELEMENT  
- sort elements

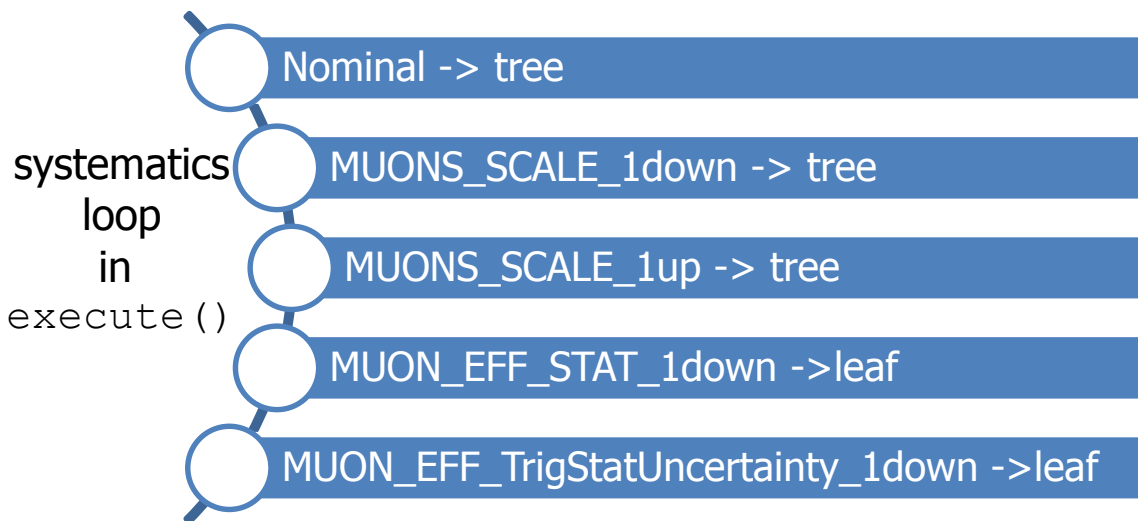
output  
trees  
histograms  
(GB)

# xTauFW Facilities in a nutshell

- Job Options
  - 100% steerable
  - fine tuning of internal operations and output
- Channels
  - use inheritance to define analysis channels
  - common functionalities
  - minimal code implementation
- Variables
  - unified definition of variables
- CP tools
  - templated handle of ~60 CP tools
- Tools
  - big collection of common methods
- Histograms
  - event cutflow, monitoring/control histos, weight histos, cloned per systematic uncertainty
- Trigger
  - decision, matching, scale factors
- MetaData
  - handle of meta data info in data (lumi etc) and MC (generator weights etc)
- Systematics
  - evaluate events considering weight and kinematic systematics

# Systematics

- Register and apply recommended systematics according to CP tool initialization
- Differentiate flavors (EL, JET, TAU, ...) and types (weight, kinematic)



## Systematic uncertainties

### ❑ Kinematic

- evaluate full event and keep separate tree with full dependencies
- container elements resorted

### ❑ Weight

- attach scale factors to the nominal tree

# Lifecycle

- xAOD production
  - MC: ½year
  - Data: usually 2-3 days after storage at T0
- Derivations production
  - Major cache: 1 month
  - Intermediate caches: 1-2 per month
  - MC: 1 week
  - Data: open-ended per major *AtlasDerivation* cache
- Mini Ntuple/xAOD production
  - weekly updates to the Atlas Software Releases
  - 2-3 days production on the Grid
- Plots with systematics: few hours



# TauAnalysisTools


- ❑ Package to provide dual-use standard tools to ease analyses involving taus
- ❑ **TauAnalysisTools** is not meant to actually re-compute energy scales or tau identification

Tool	Application
TauSelectionTool	generic tool to apply a set of requirements on tau candidates
TauSmearingTool	applies tau energy corrections
TauEfficiencyCorrectionsTool	provides identification scale factors and the associated uncertainties
TauTruthMatchingTool	performs matching of taus to the visible truth tau 4-momentum
TauTruthTrackMatchingTool	performs matching of tracks to truth taus and tracks to truth charged particles
TauOverlappingElectronLLHDecorator	decorates reconstructed taus with a likelihood score of matched reconstructed electrons

# Tau reconstruction tools offline

- Add **tauRecTools** and **TauDiscriminant** in the ATLAS Analysis Release (ongoing task)
- Goal: Perform core reconstruction of tau candidates and compute identification
- Usefulness:
  - apply new **tau energy scale**
  - apply new **tau identification**
  - can be done w/ a reprocessing/AOD fixes
  - give analyzers more power in utilizing those tools

# Synopsis

- New ATLAS EDM optimized for Run2
- Centralized frameworks for
  - data reprocessing & reduction (DF)
  - data calibration, correction, flattening (xTauFW)
  - tau recommended treatment in offline analyses (TauAnalysisTools)
  - tau energy & discriminant re-computation (tauRecTools )