PAT: the CMS Physics Analysis Toolkit

W. Adam, V. Adler, B. Hegner, L. Lista, S. Lowette, P. Maksimovic, G. Petrucciani, F. Ronga, R. Tenchini, R. Wolf

Outline

- A common physics analysis toolkit
  - Motivations
  - Requirements
  - Scope
- Design and implementation of the PAT
  - Data Model
  - The tools
  - Using PAT data files

Why an Analysis Toolkit

- **The format** of the bare output of the offline reconstruction is not user friendly:
  - some basic analysis tasks technically tricky (e.g. they require complex book-keeping)
  - not fully usable with simple tools (e.g. TTree::Draw)
- **The physics contents** of the reco. is too inclusive for most analyses, so extra object selection and cleaning is needed.

A CMS-wide toolkit

A unique analysis toolkit for all CMS:

- Avoids duplication of efforts.
- Eases comparison and validation of analysis results
- Provides a well defined starting point for people approaching phsysics analysis in CMS

Essential requirements

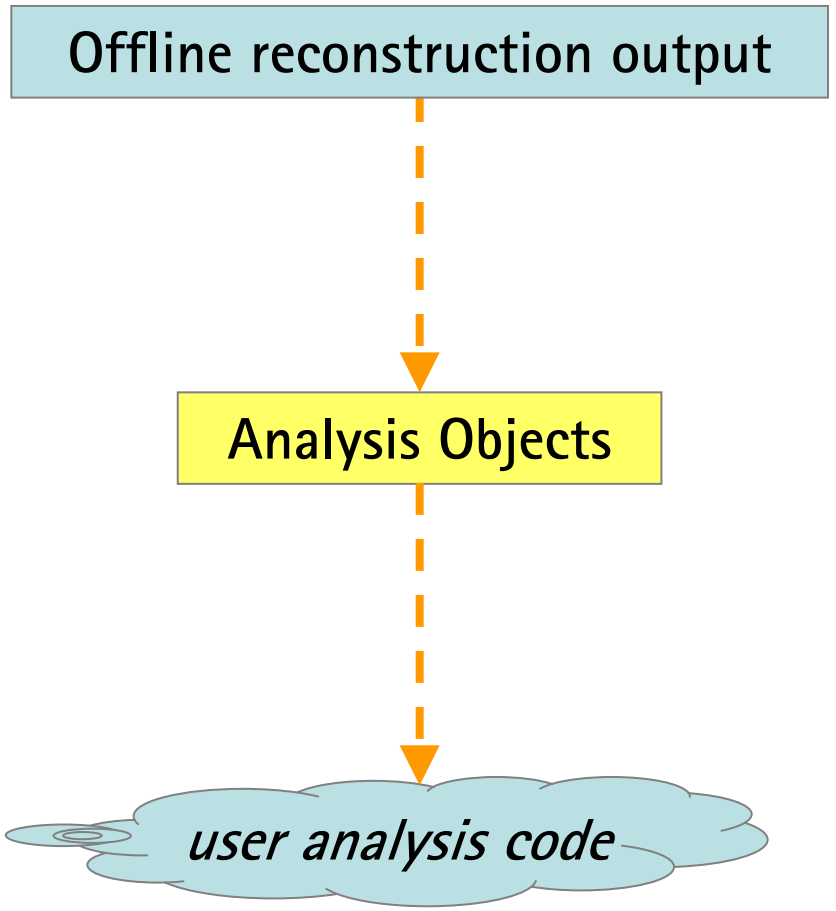
The three main requirements for the toolkit

- It must be easy to use by unexperienced people
- It must be flexible enough to cover the wide range of physics analyses in CMS
- It must not constrain what experienced users can achieve, with a little more effort.

Scope of the toolkit

The Physics Analysis Toolkit:

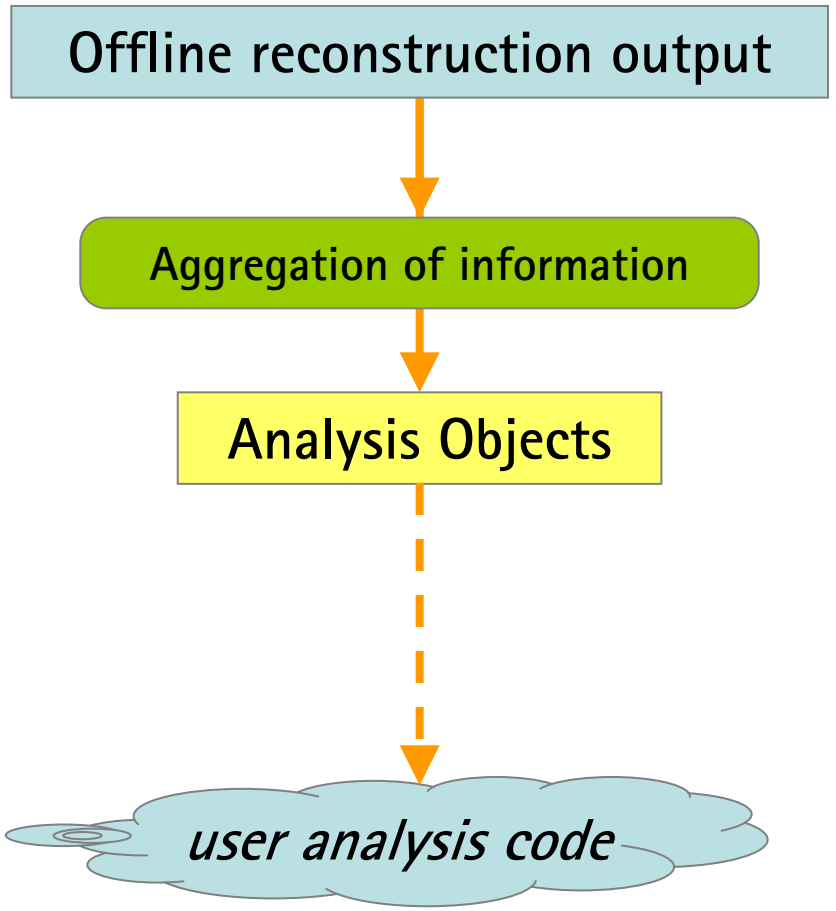
1. Defines "Analysis Objects" that are more usable than the ones from offline reco.



Scope of the toolkit

The Physics Analysis Toolkit:

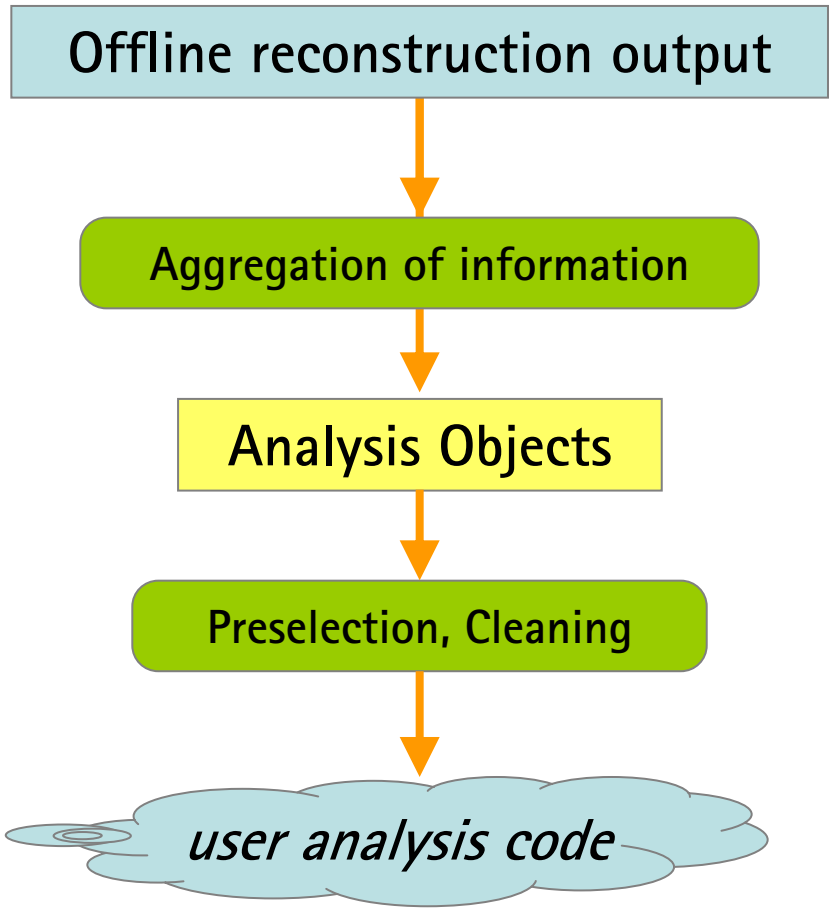
1. Defines "Analysis Objects" that are more usable than the ones from offline reco.
2. Provides tools to aggregate the output of reco. into these analysis objects



Scope of the toolkit

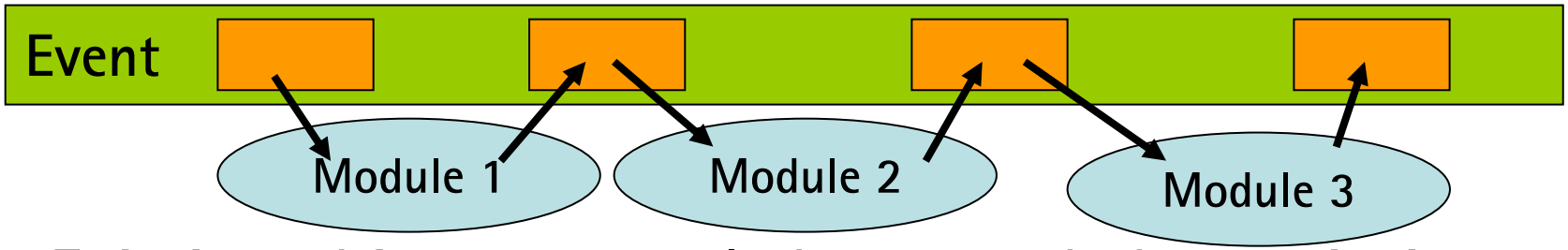
The Physics Analysis Toolkit:

1. Defines "Analysis Objects" that are more usable than the ones from offline reco.
2. Provides tools to aggregate the output of reco. into these analysis objects
3. Provides tools to perform cleaning and preselection of these analysis objects

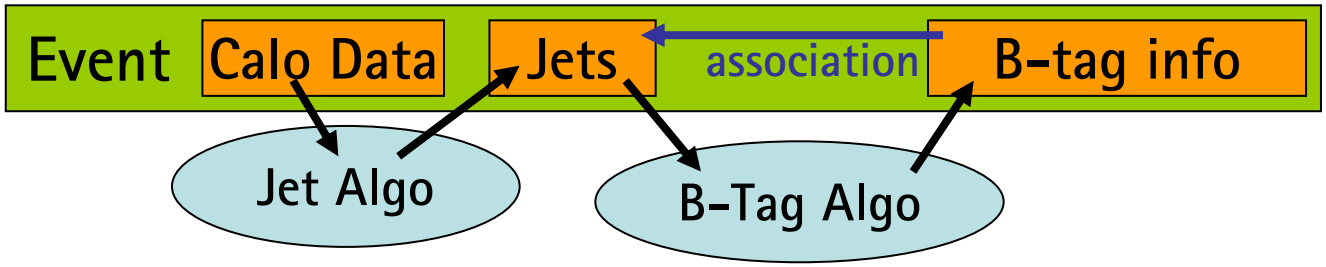


CMS Event Data Model (A very naïve picture)

- Data processing is done by a sequence of modules, which read and write to an "Event" data container



- Existing objects can only be extended associating external information to them.



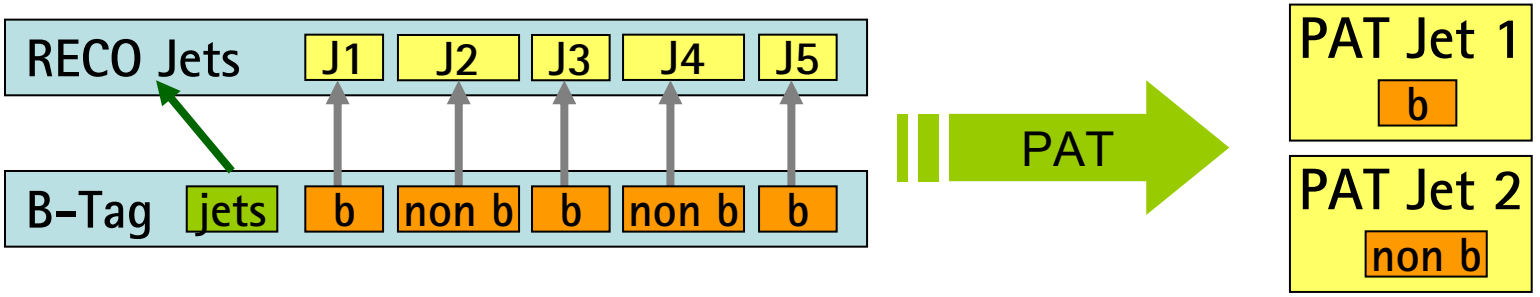
PAT Analysis Objects

The physics object classes defined by the PAT:

- Inherit from those used by offline reconstruction, to allow common tools to work on both.
- Provide a single entry point to aggregated info, to improve usability and make them standalone.
- Allow great flexibility in the choice of analysis information to store inside them.

Aggregation of information

1. Information that is externally associated to the objects in the offline data model can be imported



2. Referenced constituents can be embedded in a way that is transparent to code accessing them.

Flexibility

- For most high level tasks, a large and always evolving set of algorithms is available.
  The analysis objects can store any number of these algorithm results, accessible by algorithm name.
- To cover unforeseen or to analysis specific cases, support for generic user variables is provided.
  The implementation even allows to store any C++ object, in a fully typesafe way.
- The performance cost of these features and of the possible implementations was taken into account.

Example code

```
// Read b-tagging discriminator for a jet
float bdisc = jet.bDiscriminator("combinedSecondaryVertex")

// Plot comparison between reco and simulation truth
information on MC samples
reco::GenParticleRef mcmu = muon.genParticleRef();
if (mcmu.isNonnull()) {
    deltaPtHisto->Fill(muon.pt() – mc->pt());
}

// Access generic user data
float noiseEnergy = jet.userFloat("etInNoisyTowers")
const XYZVector * dir = jet.userData<XYZVector>("noisyTowerDir")
```

The PAT tools

- **Same modular design as the rest of CMS software:** workflow can be re-arranged to suit everyone's needs.
- **Everything is controlled by python configuration files:** safe, easy to compare, traceable from the output file.
- All intermediate steps of the processing can be persisted to disk for sharing, inspection or debugging
- To provide much flexibility without having the users change the C++ code of PAT, an expression parser based on ROOT Reflex is used to allow specifying generic cuts, e.g.

  (pt > 10) & (abs(track.d0) < 0.1) & electronID('tight')

Configurability

```
# MC matching for photons
mcPdgId     = cms.vint32(22),    # one or more PDG IDs to consider
mcStatus    = cms.vint32(1),     # one or more PYTHIA status codes
maxDeltaR = cms.double(0.2),   # deltaR for the match
maxDPtRel = cms.double(1.0),   # deltaPt/Pt for the match

# Ambiguity resolution between reco electrons and photons
checkOverlaps = cms.PSet(
    electrons = cms.PSet(
        src = cms.InputTag("cleanLayer1Electrons"),
        algorithm = cms.string("bySuperClusterSeed"),
        requireNoOverlaps = cms.bool(True),
```

Easier Configuration

To avoid users getting lost in too many configurables:

- A reference configuration is provided, with all basic features configured according to the defaults from the physics object experts.
- Additional example config files are provided for the most used non trivial changes to the configuration.
- Python helpers are provided to perform tasks that would otherwise require to change a lot of different parameters (often in a correlated way)

High-level configuration

```
# Switch off trigger matching (e.g. on MC with no trigger info)
from PhysicsTools.PatAlgos.tools.trigTools import switchTriggerOff
switchTriggerOff(process)

# Switching to SISCone R=0.5 calorimeter jets
from PhysicsTools.PatAlgos.tools.jetTools import *
switchJetCollection(process,                    # the configuration object
        cms.InputTag('sisCone5CaloJets'),   # the name of the new jets
        doBTagging=True,                # Run b-tagging
        jetCorrLabel=('SC5','Calo'),  # algo and type for JES corrections
        doType1MET=True,                # recompute Type1 MET with these
        genJetCollection=cms.InputTag("sisCone5GenJets")) # MC Jets
```

Accessing PAT data

Different supported ways of using PAT data files:

- Batch processing, just like any other CMS data file.
- Interactive analysis from ROOT (with CMS libraries), either with macros or quick inspection a la ntuple
  Events->Draw("electron.pt()","electron.trackIso() < 3")
- Visualization using the CMS Fireworks software

Conclusions

- About one year ago the development of a single CMS-wide Physics Analysis Toolkit was started. Now it's being used by a large fraction of people doing analysis on simulated data.
- The tookit defines a set of tools to perform common analysis tasks, and a data format for high level physics objects that combines usability and flexibility with good performances.
- The toolkit is suitable both for beginners and for experts:
  - It allows beginners to perform easily the standard tasks,
  - Experts can exploit its flexibility to perform advanced tasks without being constrained.