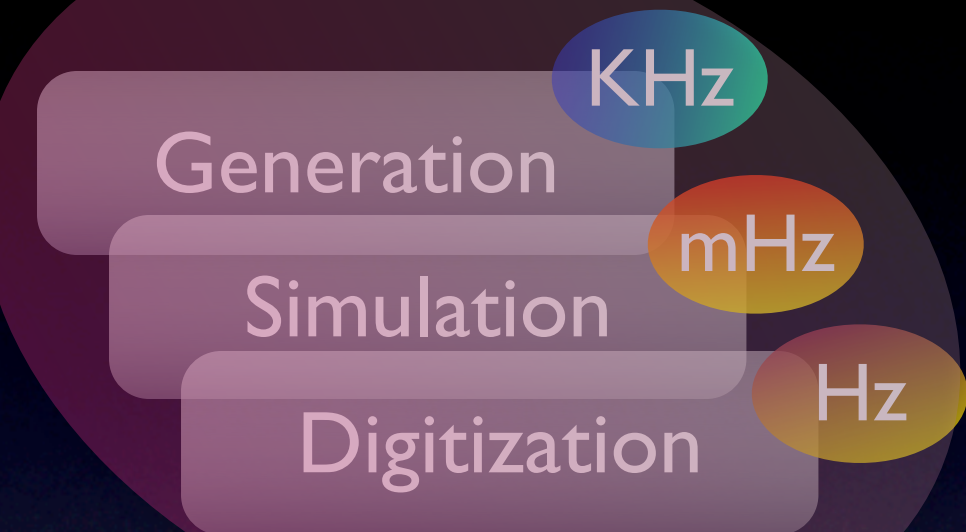


ATLAS Analysis Model

Amir Farbin
University of Texas at Arlington

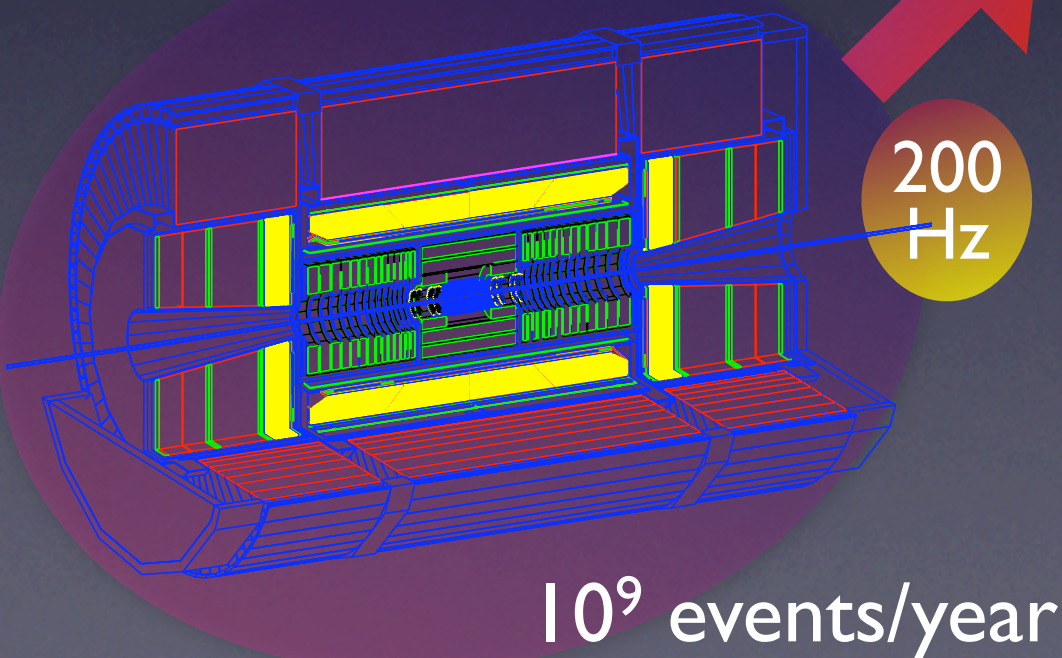
HEP Computing

Full Simulation



Will only simulate
20% of data

High-level Trigger



Data
Store

Reconstruction

Data
Base

Fast Simulation



Algorithmic
Analysis

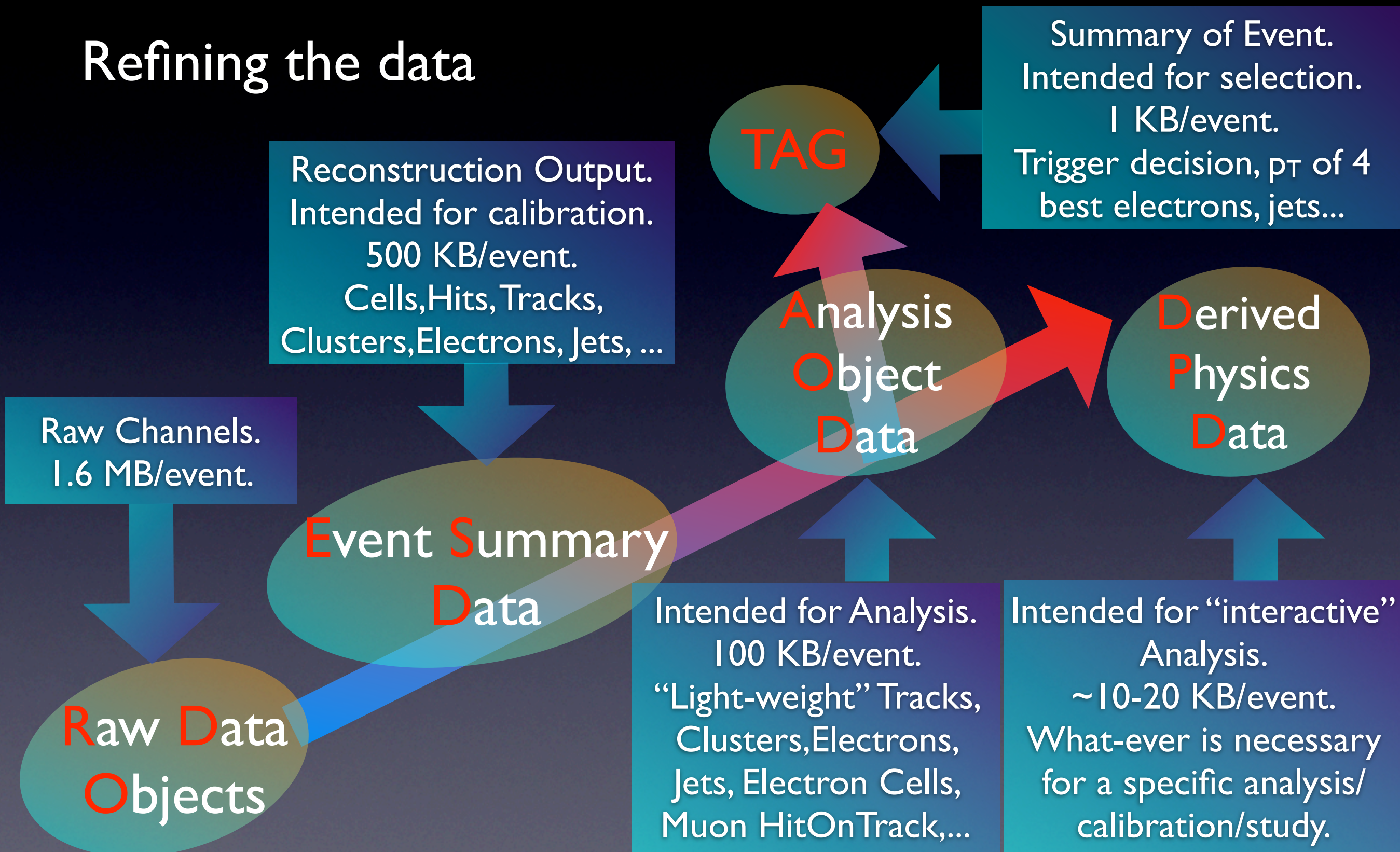
Interactive
Analysis

Statistical
Analysis

Data Analysis &
Calibration

The Event Data Model

Refining the data



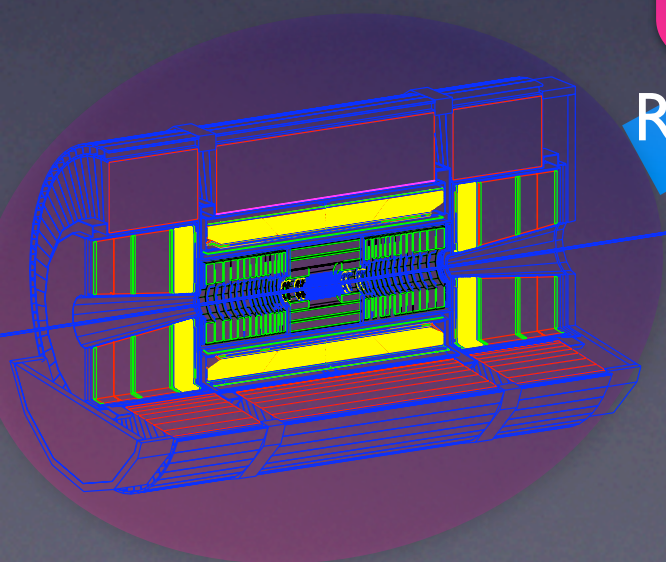
The Computing Model

- Resources Spread Around the GRID

- Derive 1st pass calibrations within 24 hours.
- Reconstruct rest of the data keeping up with data taking.

- Reprocessing of full data with improved calibrations 2 months after data taking.
- Managed Tape Access: RAW, ESD
- Disk Access: AOD, fraction of ESD

- Interactive Analysis
- Plots, Fits, Toy MC, Studies, ...



Tier 0

RAW

CERN
Analysis
Facility

RAW/
AOD/
ESD

Tier I

10 Sites Worldwide

AOD

Tier 2

30 Sites Worldwide

DPD

Tier 3

- Production of simulated events.
- User Analysis: 12 CPU/Analyzer
- Disk Store: AOD

- Primary purpose: calibrations
- Small subset of collaboration will have access to full ESD.
- Limited Access to RAW Data.

Framework Elements

- Athena is an extended version of LHCb's Gaudi framework used for high-level trigger, simulation/reconstruction, and analysis.
- Elements:
 - *Algorithms*- one execute per event, managed by framework.
 - *Tools*- multiple executes per event.
 - Event Data
 - Services
 - StoreGate- Transient Data Store- Mechanism for communication between Algorithms
 - Tool Service- Tool Factory
 - Interval of validity
 - Histogram Service
 - POOL- Persistency

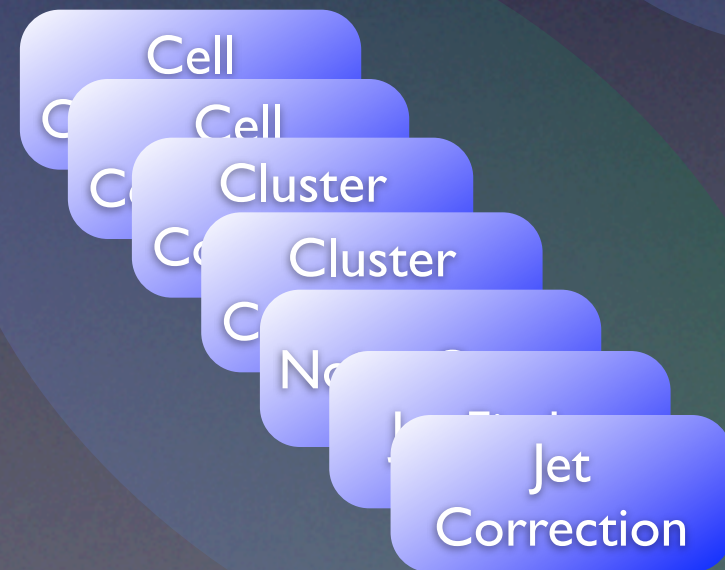
Algorithms:
Per-event
Operations



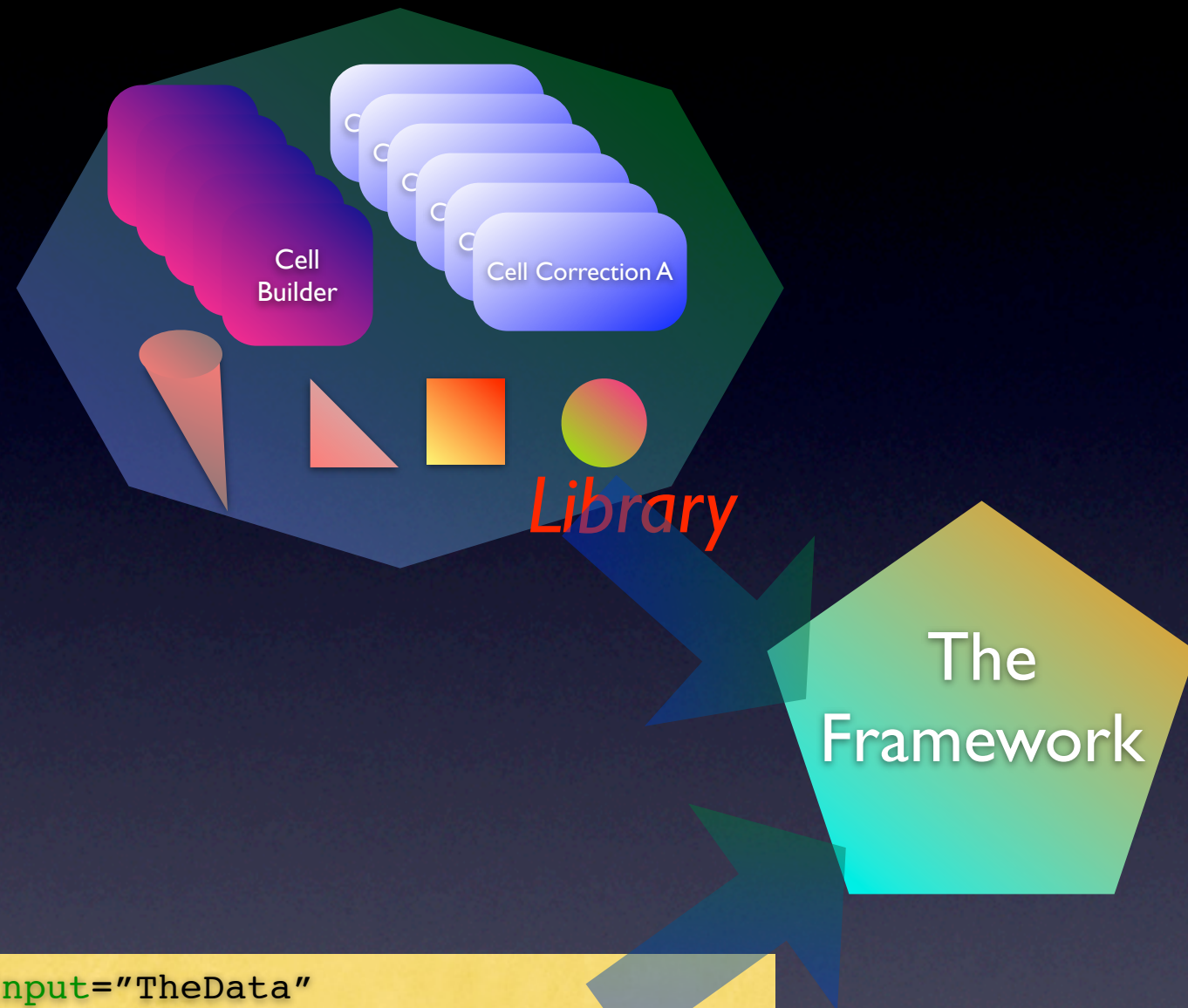
Event Data



Tools:
Per-object
Operations



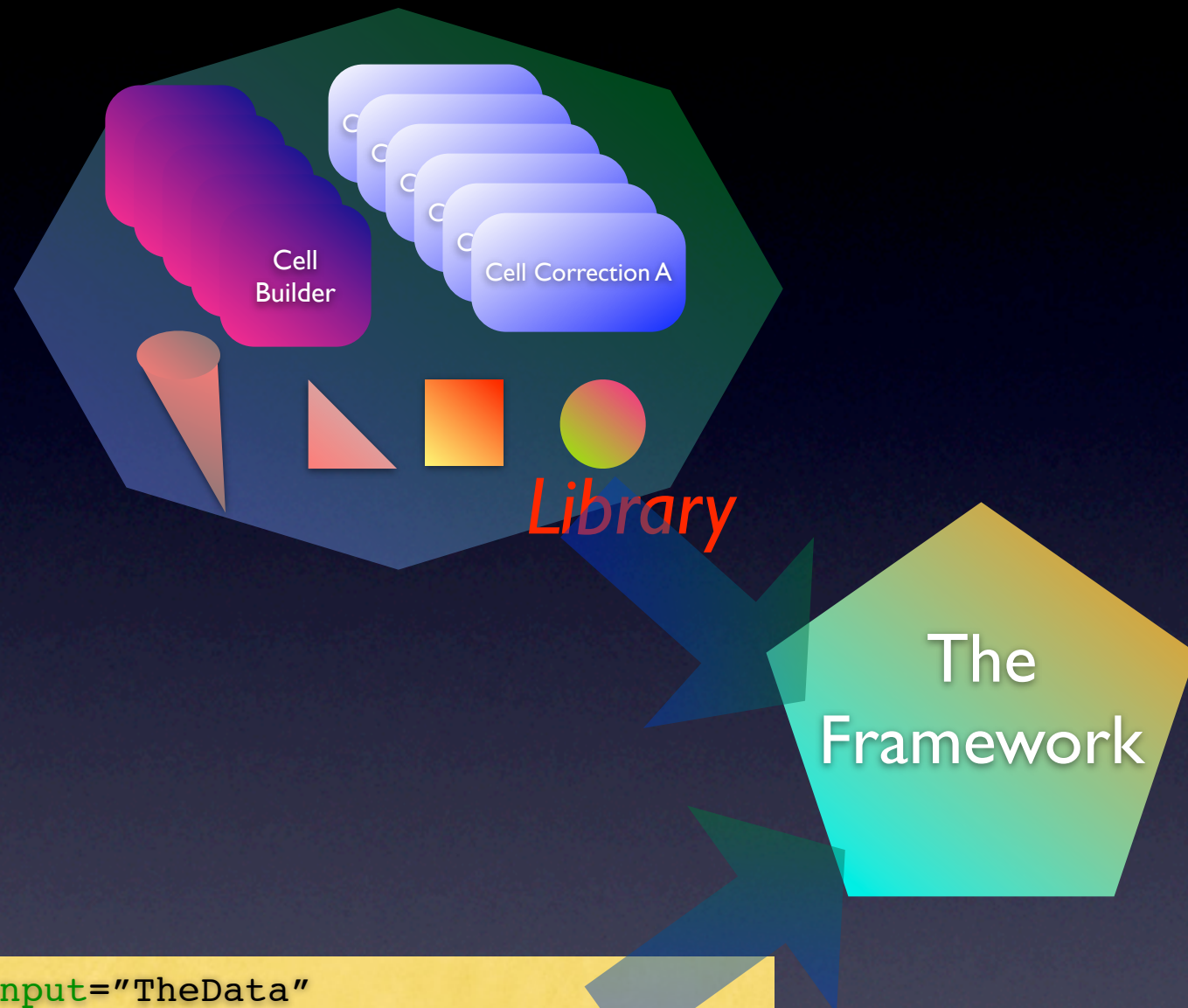
- The reconstruction application is a specific configuration of a library of framework elements.



```
Input="TheData"  
Algorithms+=CellBuilder  
(In="LArgChannels",Out="Cells1")  
Algorithms+=CellCalibrator  
(In="Cells1",Out="Cells2")  
CellCalibrator+=CellCorrectionA()  
CellCalibrator+=CellCorrectionB()  
Algorithms+=ClusterBuilder  
(In="Cells2",Out="Clusters1",MinEnergy=10*GeV)  
....
```

A Configuration

- The reconstruction application is a specific configuration of a library of framework elements.



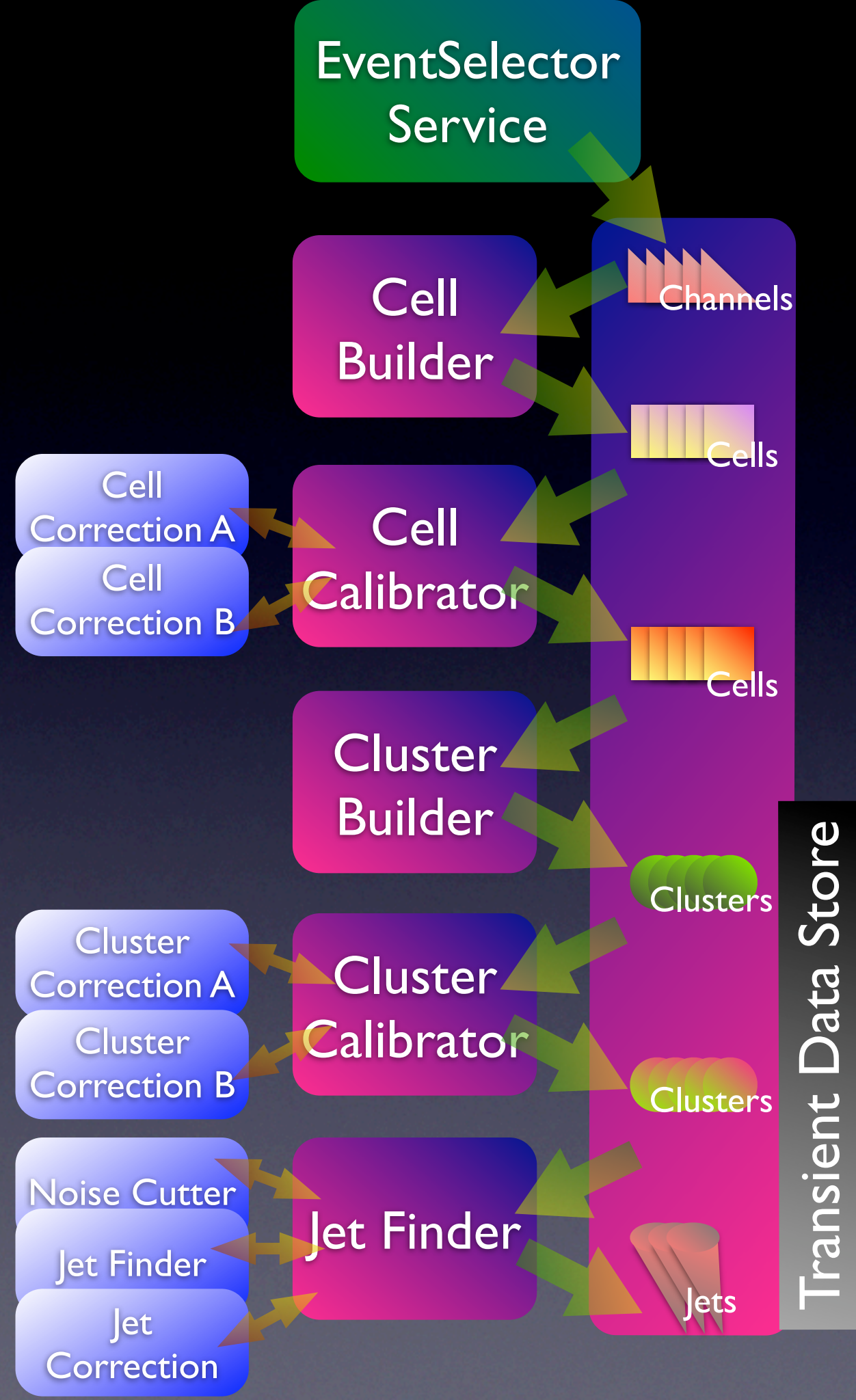
Library

The Framework

```

Input="TheData"
Algorithms+=CellBuilder
(In="LArgChannels",Out="Cells1")
Algorithms+=CellCalibrator
(In="Cells1",Out="Cells2")
CellCalibrator+=CellCorrectionA()
CellCalibrator+=CellCorrectionB()
Algorithms+=ClusterBuilder
(In="Cells2",Out="Clusters1",MinEnergy=10*GeV)
....
  
```

A Configuration



Lessons from Other Experiments I

Observations from:
BaBar, CDF, D0, H1
*ATLAS Analysis Model
Workshop (Oct 2006)*

- *Observation:* Speed is the most important factor in the Analysis Model adopted by users...no matter what the management says or sw-developers provide.
- When it is impractical to repeatedly iterate analyses on AOD, users dump large ntuples which mostly copy AOD contents... and perform analysis outside the software framework.
- Solution:
 - Optimize AOD access speed to can close to the ROOT limit (10MB/s).
(Transient/Persistent Separation)
 - Allow direct access to AOD in ROOT.

Lessons from Other Experiments II

Observations from:
BaBar, CDF, D0, H1
*ATLAS Analysis Model
Workshop (Oct 2006)*

- *Observation:* Tasks naively thought to be addressed by “ESD”-based analysis or reprocessing (eg: calibration, alignment, track-fit, re-clustering) are routinely performed in the highest level of analysis.
 - ➡ As experiments evolve:
 - “ESD” bloated and too difficult to access \Rightarrow dropped
 - “AOD” is gradually augmented with some “ESD” quantities (eg: hits in roads/cells) to provide greater functionality at analysis time.
- Solution:
 - Make sure reconstruction and calibration can be applied to AOD objects.
 - Make it easy to adjust the content of the AOD.
 - Add sufficient information to the AOD permit foreseen analysis tasks. Lots of recent iterations on AOD content in the context of analysis model.

What is Analysis?

- *Re-reconstruction/re-calibration*- often necessary.
- *Algorithmic Analysis*: Data Manipulations ESD→AOD→DPD→DPD

Tier 1/2

- *Skimming*- Keep interesting events
- *Thinning*- Keep interesting objects in events
- *Slimming*- Keep interesting info in objects
- *Reduction*- Build higher-level data which encapsulates results of algorithms
- Basic principle: Smaller data → more portable & faster read

Tier 3

- *Interactive Analysis*: Making plots/performing studies on highly reduced data.
- *Statistical Analysis*: Perform fits, produce toy Monte Carlos, calculate significance.

Stages in Analysis

- Use TAG to quickly select subset of events which are interesting for analysis.
- Starting from the AOD

Physics Group

- Stage 0: Re-reconstruction, re-calibration, selection (AOD)
 - Redo some clustering/track fitting, calculate shower shapes, apply corrections, etc...
 - Typical: 250 ms/event, In: 75% AOD, out 50% AOD
- Stage 1: Selection/Overlap removal/complicated analysis (AOD/DPD)

Analysis Group

- Select electrons/photons → find jets on remaining clusters → b-tag → calculate MET
 - Perform observable calculation, combinatorics + kinematic fitting, ...
 - Typical: 20 ms/event, In: 25% AOD, Out: 10% AOD

Personal

- Stage 2: Interactive analysis (AOD/DPD)
 - Final selections, plots, studies.
 - Prototype earlier steps!
 - Typical: 0 ms/event, In: 1% AOD, Out: 0
- Stage 3: Statistical Analysis

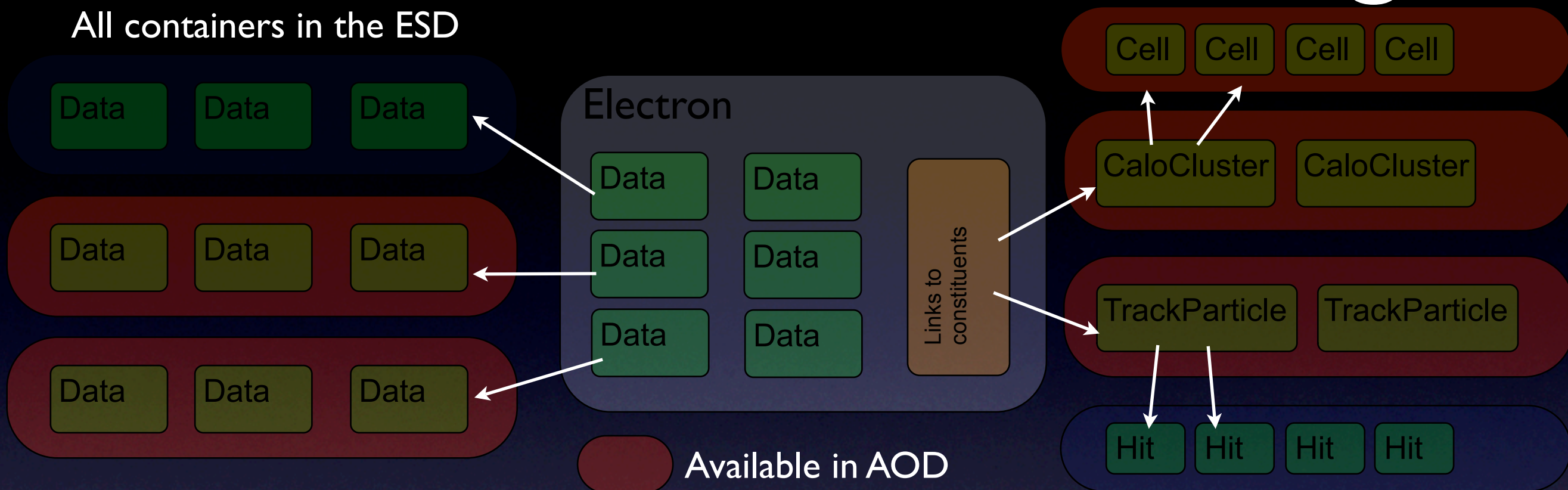
Stages vs Resources

- ATLAS will record 200 Hz of data, regardless of luminosity → 10^9 event/year.
- CM Assumption 700 Analyzers: 12 tier 2 CPU/person for analysis at any give time.
- Not unusual for some analysis to start with 50% of the data.
- Assuming perfect software/hardware (10 MB/s read in = ROOT limit).

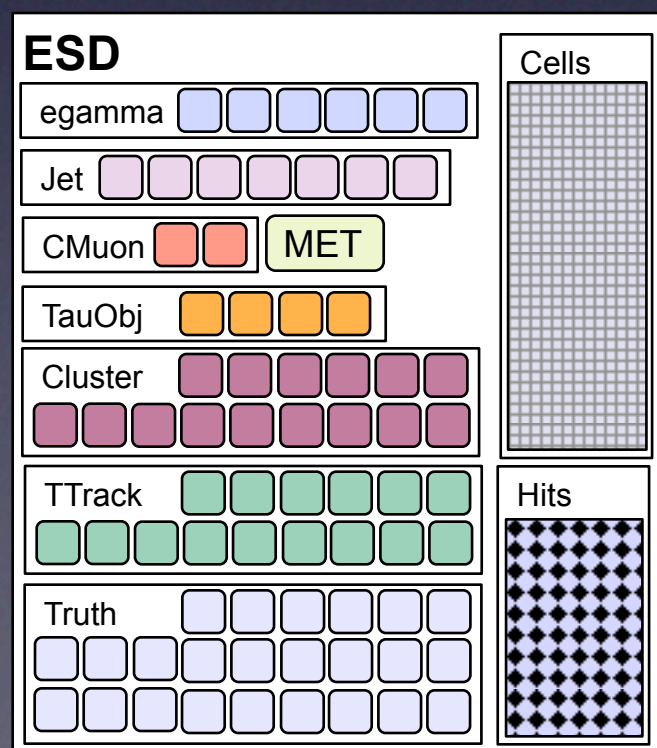
		Laptop 1 Cores	Tier 3 25 Cores	Tier 2 10 Persons 100 Cores	Tier 2 100 Persons 1000 Cores	
Step 0	1 Hour	0.0001%	0.0035%	0.0140%	0.1398%	Working group on Tier 2
	Overnight	0.0017%	0.0419%	0.1678%	1.6777%	
	1 Week	0.0235%	0.5872%	2.3487%	23.4874%	
	1 Month	0.1007%	2.5165%	10.0660%	All	
Step 1						Analysis group on Tier 2
	1 Hour	0.0016%	0.0400%	0.1600%	1.6000%	
	Overnight	0.0192%	0.4800%	1.9200%	19.2000%	
	1 Week	0.2688%	6.7200%	26.8800%	All	
Step 2	1 Month	1.1520%	28.8000%	All	All	Single Analyzer on Tier 3
	1 Hour	0.3600%	9.0000%	36.0000%	All	
	Overnight	4.3200%	All	All	All	
	1 Week	60.4800%	All	All	All	
	1 Month	All	All	All	All	

Event Data Model Design

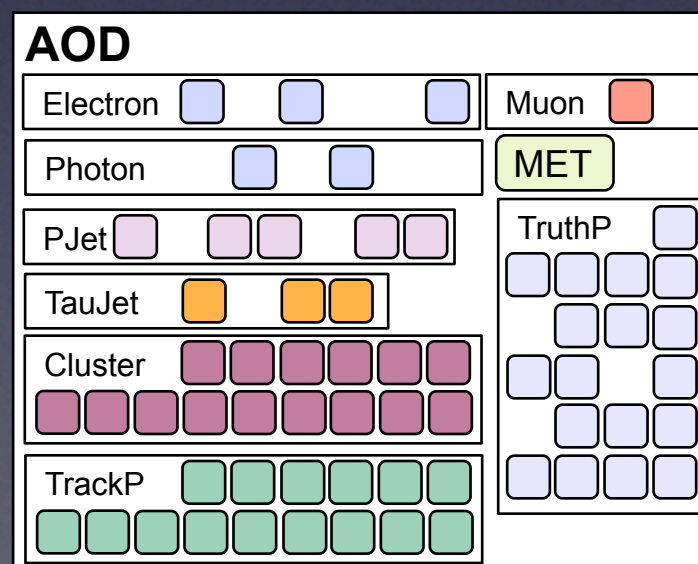
All containers in the ESD



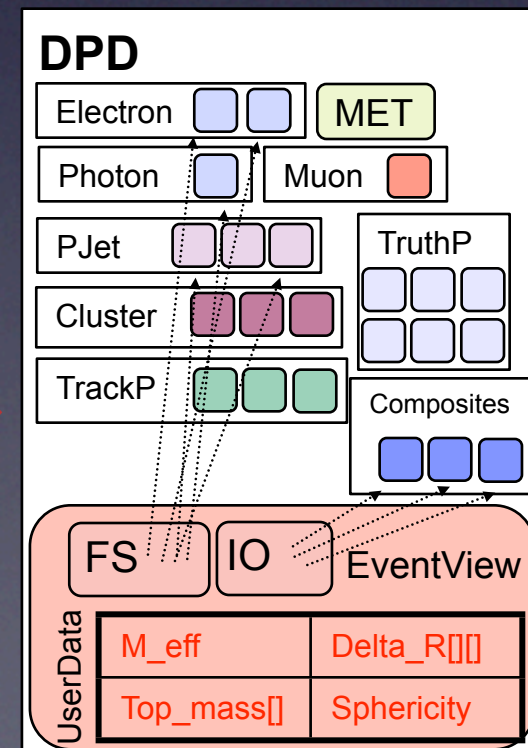
Benefits: 1. Move data between AOD/ESD w/o schema change. 2. Read on Demand



AOD
Building



DPD
Building



Why?

Stage I: DPD Building

- The AOD data-set is too big to store locally for interactive access (eg Tier 3/laptop) or to run on with one Core.
- AOD is general purpose, containing more information than necessary for any given analysis...

How?

- So analysts should skim, thin, slim, & reduce the data to a manageable size for interactive analysis.
- 2 Aspects:
 - Basic framework support for such operations
 - *Skimming*: Easy... write out subset of what you read in. Gaudi Filters.
 - *Slimming*: write out subset of input containers. POOL output list.
 - *Thinning*: write out subset of object inside containers. Thinning service.
 - *Reduction*: User annotations. Add EventView/UserData. This hasn't been fully worked out.
 - Provide tools which encapsulate the physics decisions behind these operations... eg particle selection.

Collaborative Analysis

- Problem: how do you get 2000 physicists to
 - perform analysis in consistent ways
 - easily share & compare their work
- Same problem as reconstruction.
 - The reconstruction software is simultaneously developed by 100's of people over many years.
 - A common set of framework elements form the basic language of event processing.
 - Application is created at runtime.
- Solution: Apply the same framework design to analysis → EventView Framework (see first poster session)...

The EventView

- Holds the “state” of an analysis.
- Objects in the AOD + Labels.
- Objects created in the coarse of analysis + Labels.
- **EventView is a generic analysis data object which is meant to represent the state of an analysis... sort of like an advanced ntuple which uses Athena framework elements.**
- Can be written/read from file and shared (even with a theorist!)
- Convention: each EventView holds *one* interpretation of an event... very natural book keeping tool.

EventView

Final State Particles



Inferred Objects

UserData

“Sphericity”:0.22

“Missing_Et”:41.2

“Top_Mass”:172.6

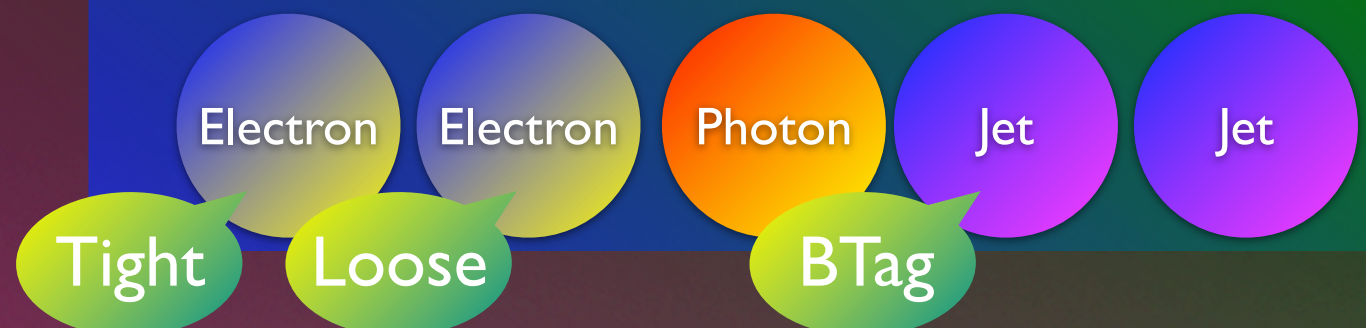
“Lep_Bjet_Th”:0.44

The EventView

- Holds the “state” of an analysis.
- Objects in the AOD + Labels.
- Objects created in the course of analysis + Labels.
- UserData: Anything other data generated during analysis.
- Can be written/read from file and shared (even with a theorist!)
- Convention: each EventView holds *one* interpretation of an event... very natural book keeping tool.

EventView

Final State Particles



Inferred Objects



UserData

“Sphericity”:0.22	“Top_Mass”:172.6
“Missing_Et”:41.2	“Lep_Bjet_Th”:0.44

EventView Framework

- Analysis is a series of EventView Tools executed in a particular order.

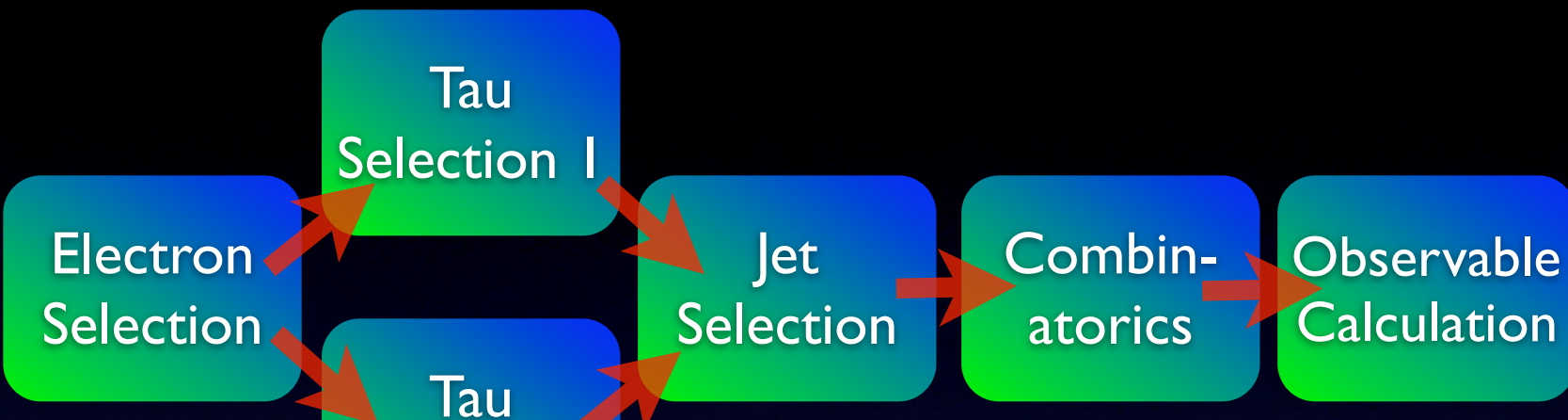
- Framework generates multiple Views of an event representing

- EventView Framework allows breaking analysis into modular pieces so different parts can be developed by different people...

- Controls the flow of analysis algorithms and data

- Different input (eg: generator, full reconstruction, fast simulation)

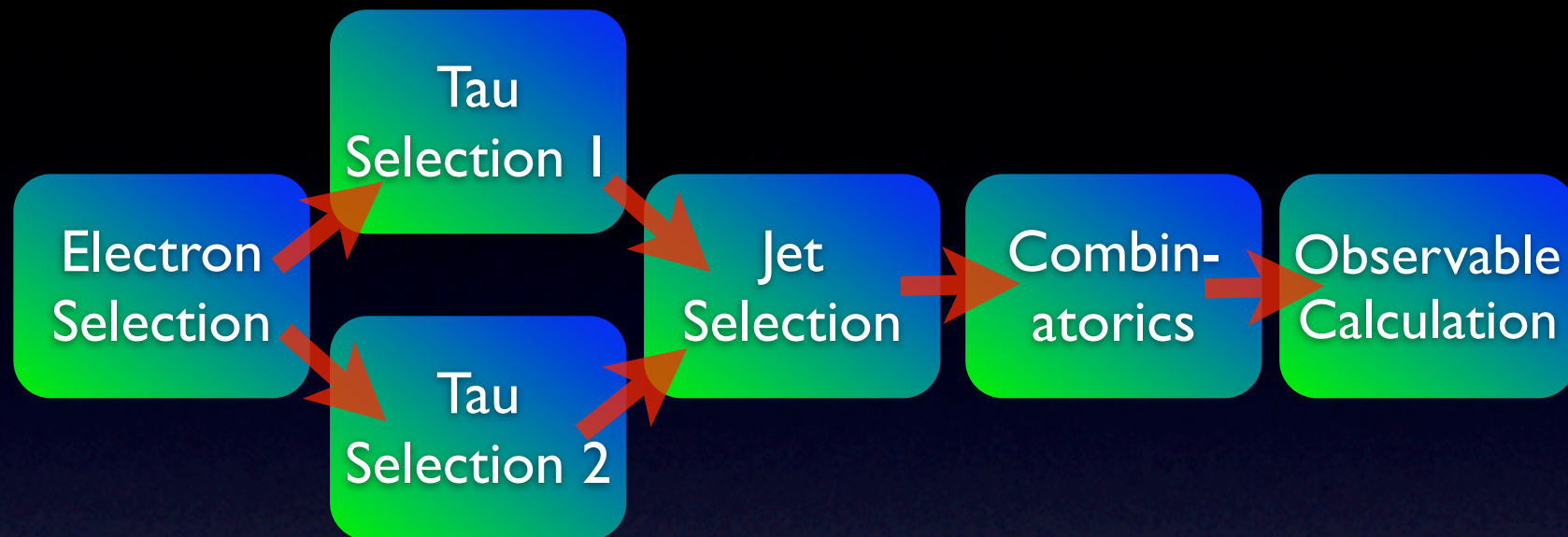
Everything consistent within one EventView \Rightarrow Framework handles bookkeeping.



Data Flow

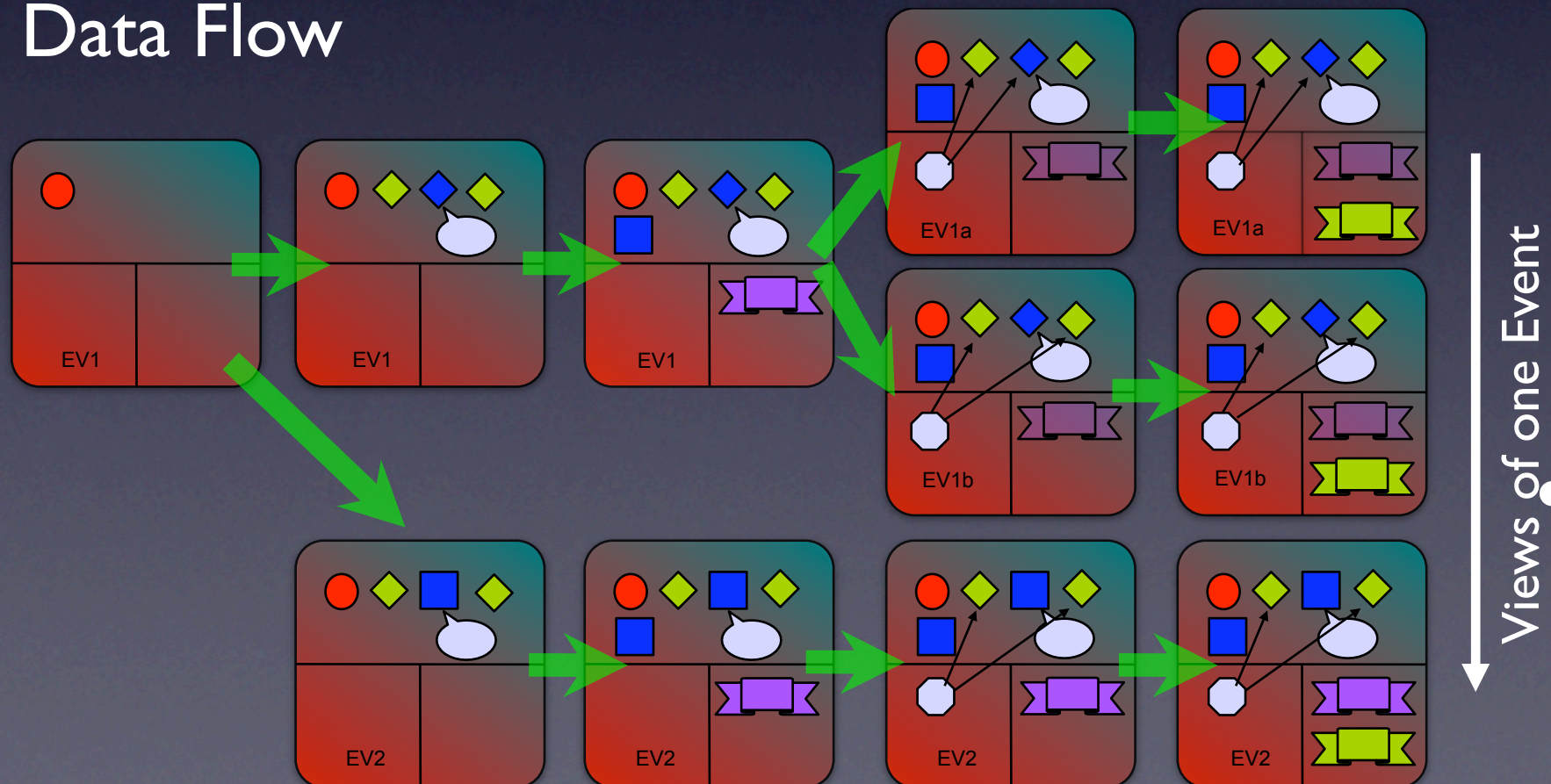


EventView Framework



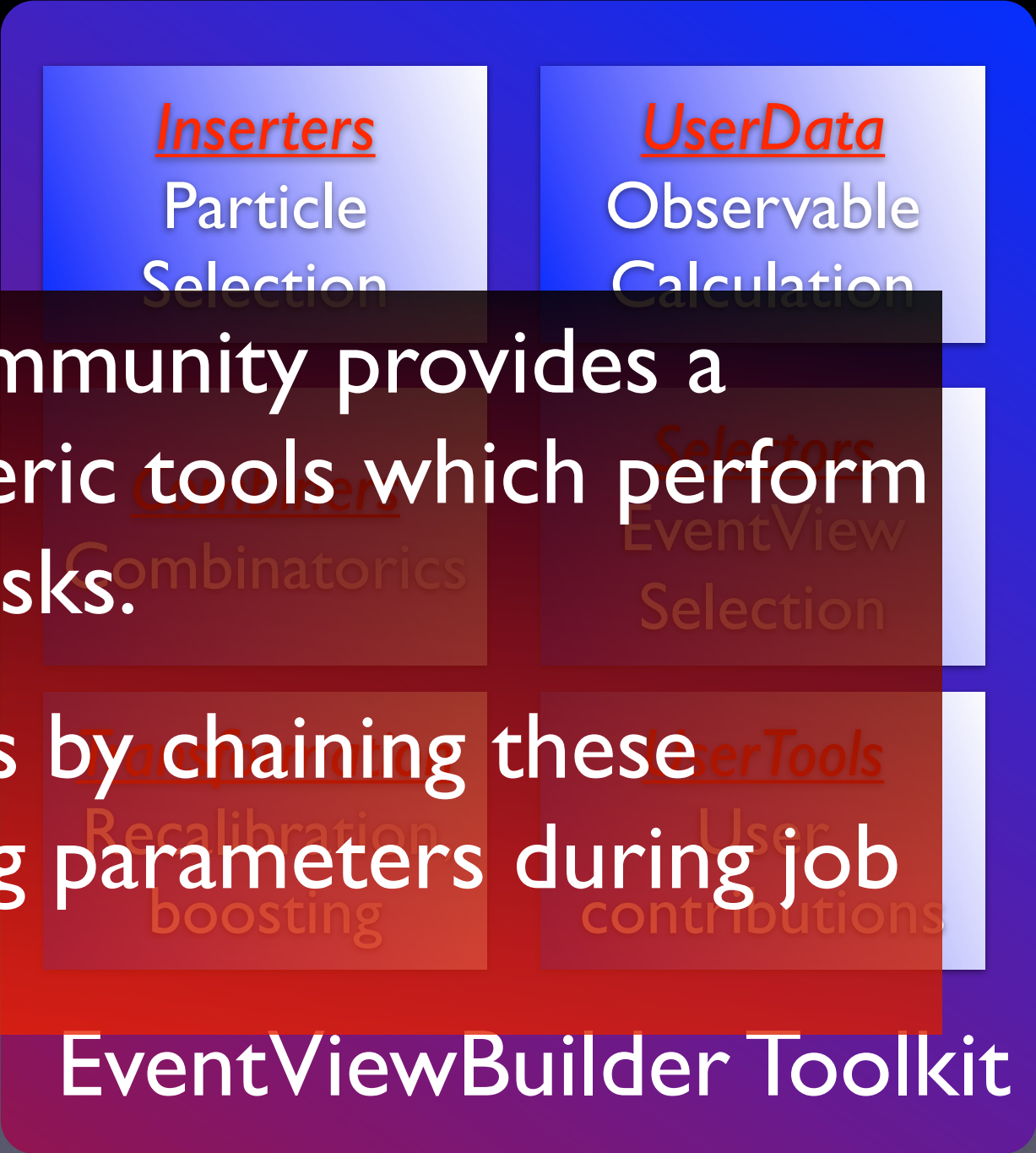
Analysis Flow

Data Flow



- Analysis is a series of EventView Tools executed in a particular order.
 - Framework generates multiple Views of an event representing
 - Different analysis paths
 - Different combinatorics choices
 - Different input (eg: generator, full reconstruction, fast simulation)
- Everything consistent within one EventView \Rightarrow Framework handles bookkeeping.

EventView Toolkit

- 100's of generalized tools which can be configured to perform specific tasks.
 - Tools instantiated/configured at runtime in python... users can perform complicated analyses w/o any C++.
 - Provide the language for basic analysis concepts: "inserters", "loopers", "associators", "calculators", "combiners", "transformers".
 - Tools explicitly designed to be extended by users (when necessary).
 - Complicated Athena stuff in base classes.
 - Users only need to implement "the physics".
 - Users now routinely contribute new tools.
- 
- The diagram illustrates the EventViewBuilder Toolkit architecture. It features a large blue rounded rectangle containing a grid of tool categories. The categories are arranged in four rows and two columns:
- Row 1: Inserters Particle Selection (left), UserData Observable Calculation (right)
 - Row 2: Combiners Combinatorics (left), Selectors EventView Selection (right)
 - Row 3: Recalibration Recalibration (left), UserTools User contributions (right)
- At the bottom of the blue rectangle, the text "EventViewBuilder Toolkit" is displayed in white. A large red semi-transparent rectangle is overlaid on the left side of the diagram, containing the text: "The EventView Community provides a large library of generic tools which perform common analysis tasks. Users build analyses by chaining these together and setting parameters during job configuration."

EventView Toolkit

- 100's of generalized tools which can be configured to perform specific tasks.
- Tools instantiated/configured at runtime in python... users can perform complicated analyses w/o any C++.
- Provide the language for basic analysis concepts: “inserter”, “looper”, “associator”, “calculator”, “combiner”, “transformer”.
- Tools explicitly designed to be extended by users (when necessary).
 - Complicated Athena stuff in base classes.
 - Users only need to implement “the physics”.
 - Users now routinely contribute new tools.

Inserters

Particle
Selection

UserData

Observable
Calculation

Combiners

Combinatorics

Selectors

EventView
Selection

Transformation

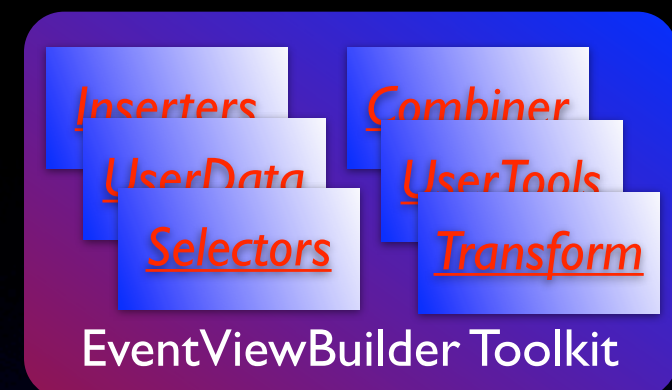
Recalibration,
boosting

UserTools

User
contributions

EventViewBuilder Toolkit

“View” Packages



SUSYView

TopView

EventView
Performance

HiggsToGG
View

- Physics community have built various high-level analysis packages which perform some of the steps of specific analysis.

- Several have been adopted as the DPD-building mechanism for Physics Analysis groups (most successful: Top Working Group).

- EventView Framework provides standardized mechanisms for building custom DPDs.
- EventView and software packages have a much faster development cycle than releases or patches! So the EV team provides/distribute pacman caches.

- Analysis packages are mostly configurations of standard tools... minimal new C++.
- HighPtView: Generic Analysis package running in production \Rightarrow Standard:

- Particle selections
- Truth/Trigger Match
- Output
- \Rightarrow Serves as benchmark/starting point for

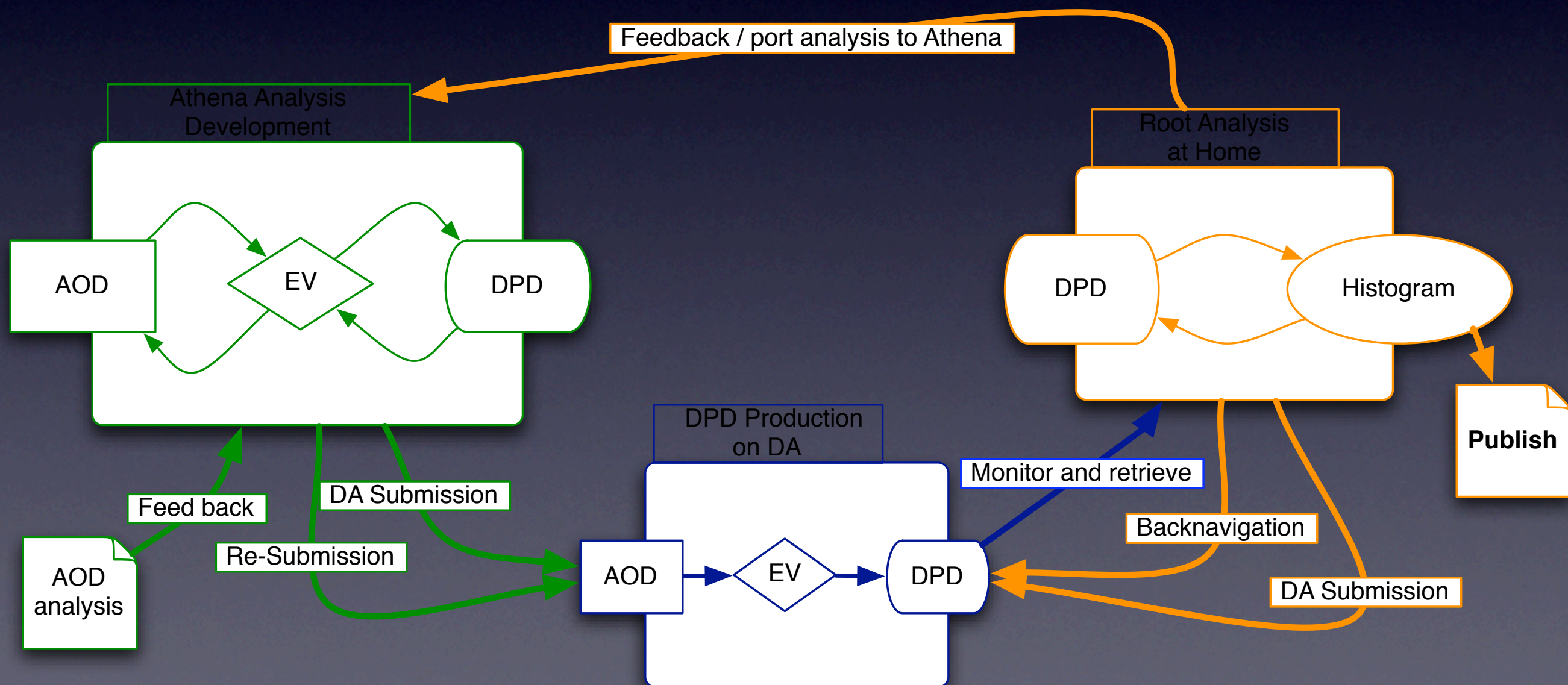
- And Performance packages: ElectornPhotonView, JetView, MuonView

“View” Packages



- EventView Framework provides standardized mechanisms for building custom DPDs.
- EventView and software packages have a much faster development cycle than releases or patches! So the EV team provides/distribute pacman caches.
- Analysis packages are mostly configurations of standard tools... minimal new C++.
- HighPtView: Generic Analysis package running in production \Rightarrow Standard:
 - Particle selections
 - Truth/Trigger Match
 - Output \Rightarrow Serves as benchmark/starting point for analyses
- Many physics groups customizing HighPtView for specific analyses \Rightarrow SUSYView, TopView, ...
- And Performance packages: ElectornPhotonView, JetView, MuonView

Analysis Work Flow



Stage 2: Interactive Analysis

- Format of the DPD
 - Use athena convertors to read EDM objects into ROOT... so the DPD format is the same as AOD/ESD.
 - Allow saving additional non-standard info... eg EventView/UserData.
- Dataset management
 - N datasets (eg data, signal MC, bkg1 MC, bkg2 MC, ...)
 - M_i files in each... different cross-section, preselection (trigger?) efficiency, ...
- Interactive Plotting (eg TTree::Draw).
 - Limited. Usefulness depends on DPD format. Ex: With EventView DPD you can make efficiency, resolution, scale plots because results of matching is stored in DPD.
 - But inefficient for making lots of plots from the same dataset because each plot requires its own loop over data.
- Batch Analysis (eg TTree::MakeClass → Loop())
 - As sophisticated as your input DPD allows. Compile for speed.
 - Simultaneously generate multiple histograms, ntuples, etc...
- Finalizing plots, making tables, etc

Interactive Analysis Frameworks

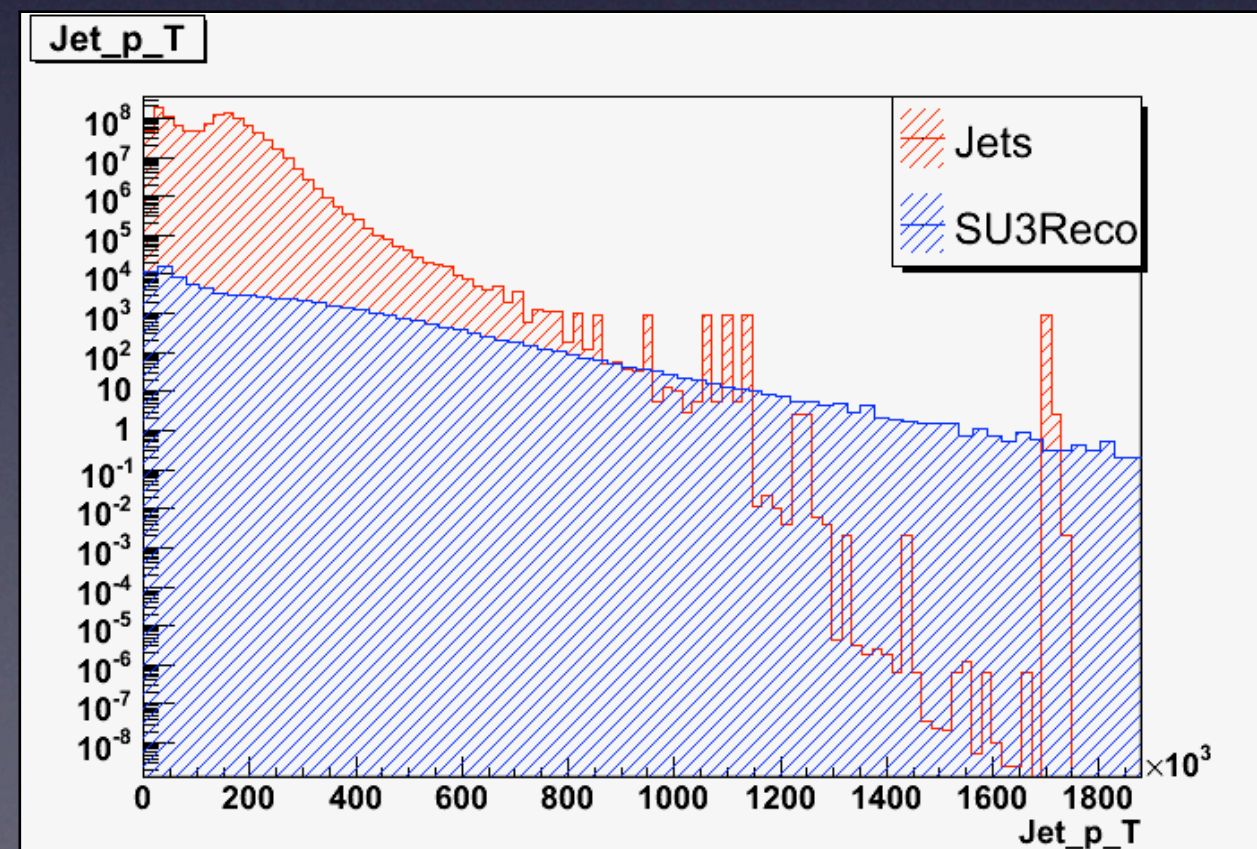
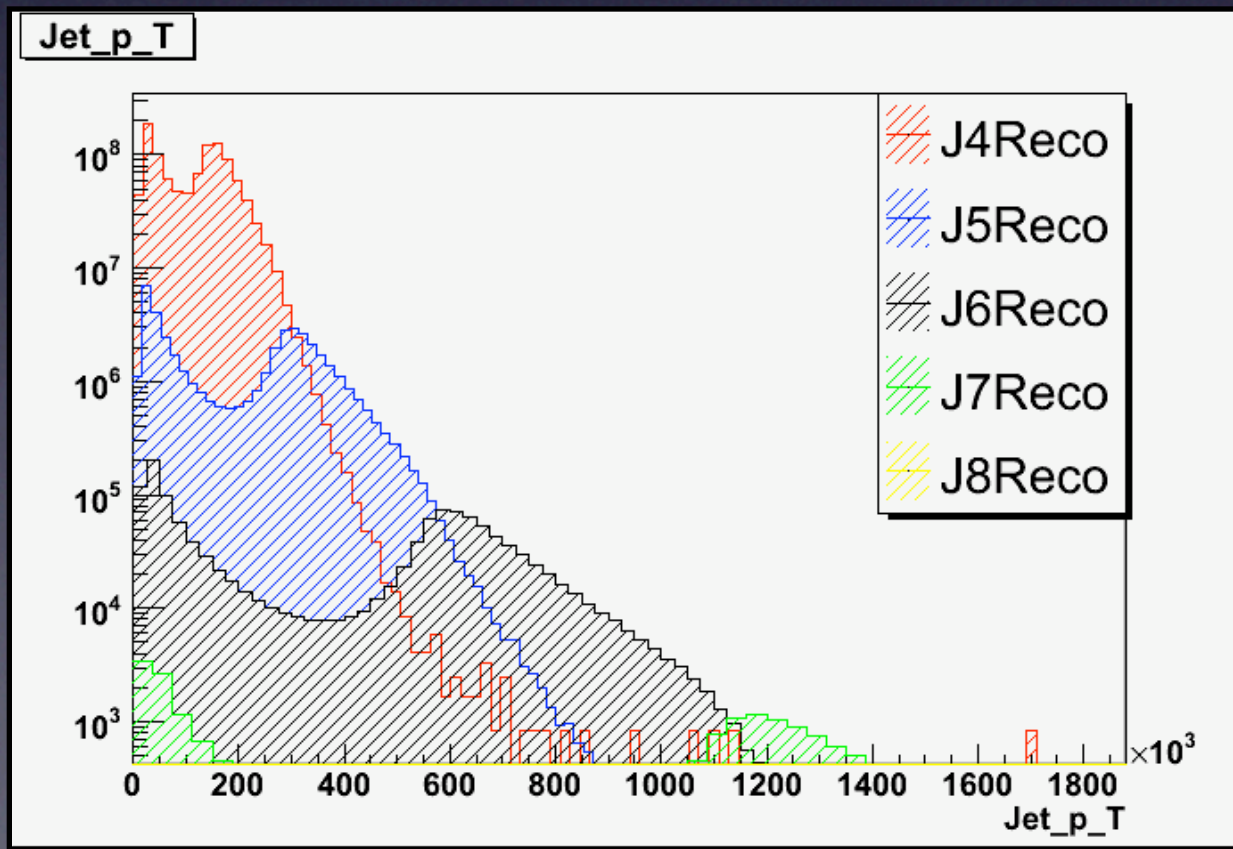
- ROOT/PyRoot frameworks emerge as analyses become more sophisticated than what is manageable in a macro.

SPyRoot

```
import SampleHandler
Data=SampleHandler.SampleGroup()
BaseDirectory="/space2/jboyd/anal/data/v7/"
SampleNames= [ ["J4",3.08E+005], ["J5",12470], ["J6",360.4], ["J7",5.707], ["J8",
0.24], ["SU3",19.3] ]
for S in SampleNames:
    Data.AddDirectory(S[0]+"Reco",BaseDirectory+S[0],"EV0","Reco",S[1])
```

```
- Data.Compare
  (["J4Reco","J5Reco","J6Reco","J7Reco",
   "J8Reco"],"Jet_p_T")
```

```
- Data.AddCombinedSample('Jets',Data,
  ['J4Reco','J5Reco','J6Reco','J7Reco','J8R
  eco'])
- Data.Compare
  (['Jets','SU3Reco'],'Jet_p_T')
```



Batch Analysis

```
TheAnalysis=TTreeAlgorithmLooper("TestAnalysis")
```

```
TheAnalysis.AddAlgorithm(VarHistAlgorithm("JetN_hist","JetN_hist","JetN_hist","T.jetN",  
20,0,20,["jetN"]))
```

```
TheAnalysis.AddAlgorithm(SimpleVarCutAlgorithm("4JetsCut","T.jetN>3",["jetN"]))
```

```
TheAnalysis.AddAlgorithm(VarHistAlgorithm  
("MET_hist","MET_hist","MET_hist","T.MissingEt",100,0,1000000,["MissingEt"]))
```

```
TheAnalysis.AddAlgorithm(SimpleVarCutAlgorithm("METCut","T.MissingEt>100000.",  
["MissingEt"]))
```

```
TheAnalysis.AddAlgorithm(TransverseMassAlg("M_T"))
```

```
TheAnalysis.AddAlgorithm(WriterAlgorithm(["M_T","Jet_N",...])
```

```
import RunHandler
```

```
RH=RunHandler.RunHandler(["SU4Reco","J1Reco"],TheAnalysis,"myRH")  
RH.Loop()
```

```
import pickleResults
```

```
pickleResults.save(RH.Results,"myAnalysis_")
```

```
res.GetCutTable(Samples=["SU3Reco"],Lumi=1000.0)
```

Sample: **SU3Reco**

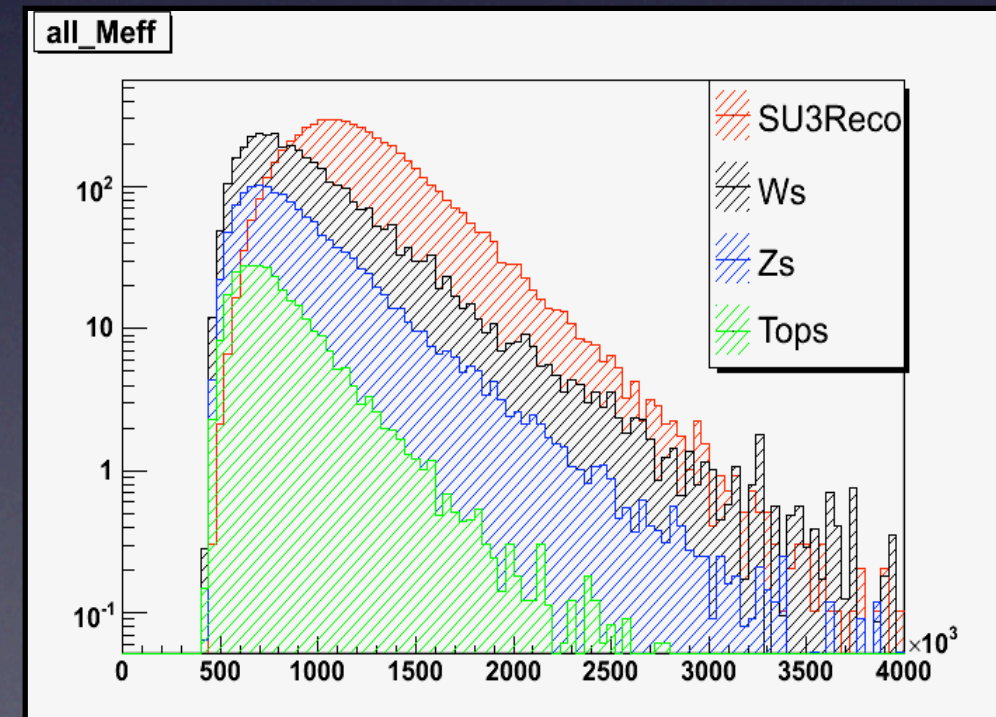
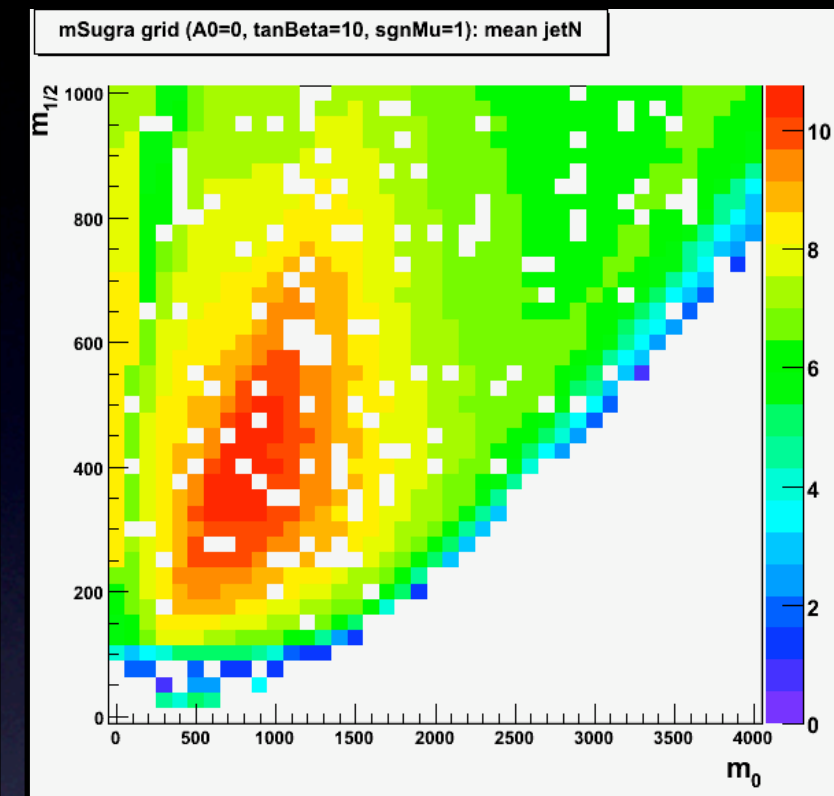
Cut: 4JetsCut -> Eff: 0.69 err: 0.001

Cut: METCut -> Eff: 0.88 err: 0.0009

Cut: JetCutAlgo -> Eff: 0.47 err: 0.001

Final Cut Effic: 0.29 err: 0.001

exp. evts. (after cuts): 5654.1 err: 5.9



Summary

- ATLAS Analysis Model focuses on ensuring framework, event data model, analysis tools, and persistency technologies allow analyzers to:
 - Re-reconstruct and re-calibrate objects on AOD while still remaining within the space budget.
 - Unify reconstruction and analysis objects.
 - Carefully tune AOD contents.
 - Build custom Derived Physics Data.
 - Identify basic operations: skimming, thinning, slimming, reducing
 - Provide framework support for these operations.
 - Provide a high-level framework for collaborative development of analysis packages based on common tools.
 - Efficiently analyze DPDs on local resources.
 - Make framework objects directly readable in ROOT.