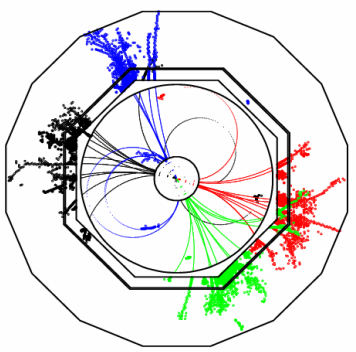


Working with Pandora PFA

John Marshall,
University of Cambridge

ILD Workshop, DESY, July 5 2010

Pandora Structure

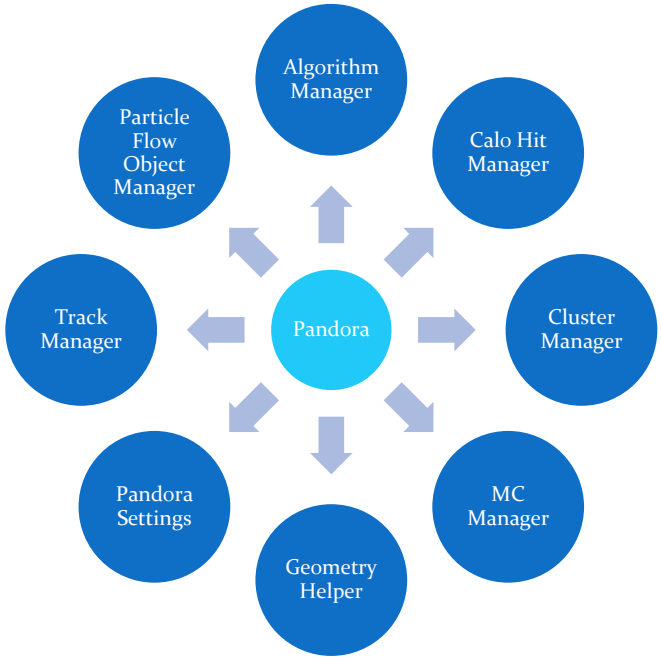
User Application:

- Specify Geometry
- Create Calo Hits
- Create Tracks
- Create MC Particles
- Register User Content

Get Particle Flow Objects

Pandora API

Pandora Framework, treat as "black box":



Pandora Content API

Pandora Algorithms:

- Clustering Algorithm
- Topological Association Algorithms
- Statistical Reclustering Algorithm
- Photon Recovery Algorithm
- Fragment Removal Algorithms
- Track-cluster Association Algorithms
- PFO Construction Algorithm

Writing a Pandora Application

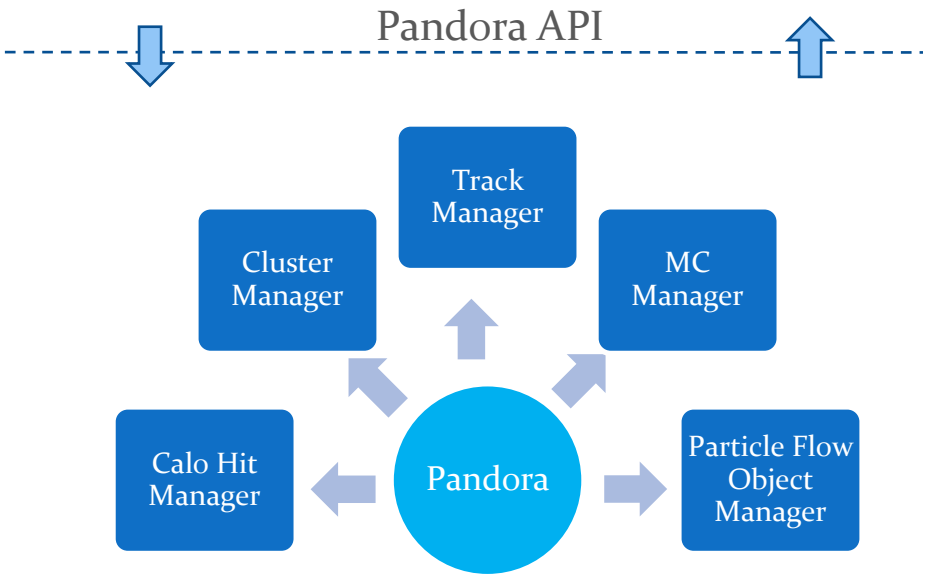
- To run Pandora, a user needs to write a small application in their chosen software framework.
- This application uses the PandoraAPI to supply Pandora with details of the detector geometry and of the calo hits and tracks in each event.
- Pandora then builds its own simple objects.

- Construction of these objects is easy; the user makes a Parameters class, fills the member variables and then calls the API Create function.
- Example member variables for a track:
  d0, z0, track state at start, track state at ECal, etc.
- All member variables must be specified, or an exception will be thrown when Create is called.

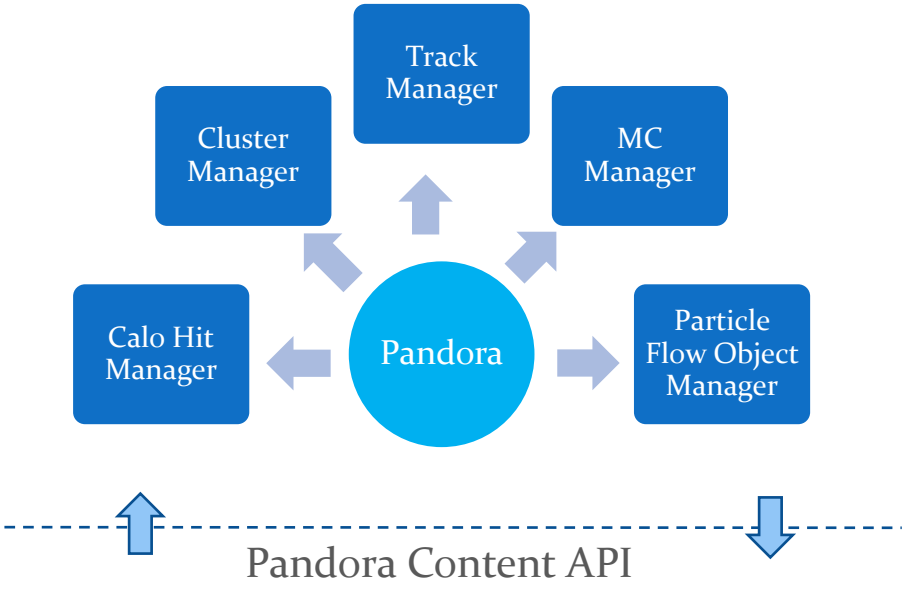
- The user can provide this information in any order, then call the API ProcessEvent function.
- Finally, user calls the API GetParticleFlowObjects function.

User Application, e.g. MarlinPandora

```
PandoraApi::Track::Parameters parameters;
parameters.m_d0 = ...;
 ...
PandoraApi::Track::Create(pandora, parameters);
```



Writing a Pandora Algorithm



e.g. Clustering Algorithm

```
PandoraContentApi::GetCurrentTrackList(...);
PandoraContentApi::GetCurrentOrderedCaloHitList(...);
 ...
PandoraContentApi::Cluster::Create(...);
```

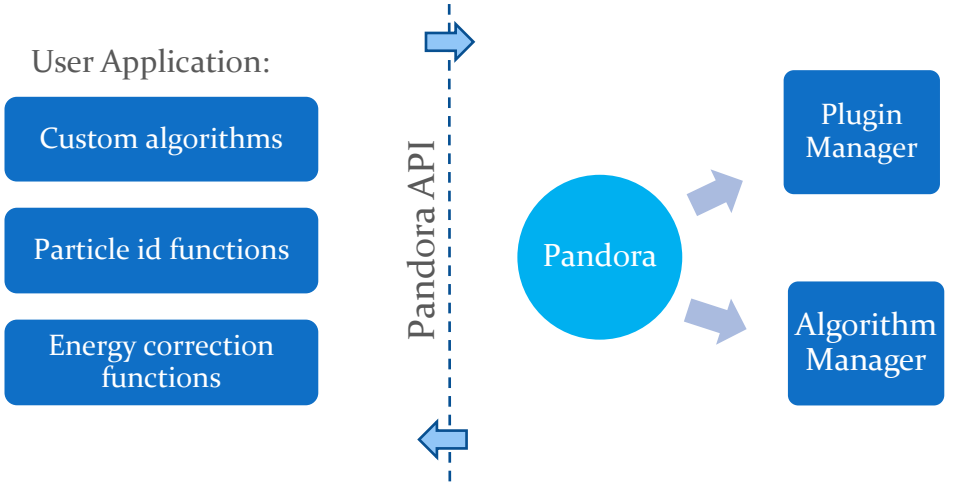
- Pandora Managers are designed to store named lists of their respective objects.
- These objects can be accessed by the Pandora Algorithms, which perform the reconstruction.
- The algorithms interact with the Managers in a controlled way, via the PandoraContentAPI, and the Managers perform the memory management.

- The algorithms should therefore contain exclusively physics-driven code, alongside the following typical usages of the PandoraContentAPI:
  - Create new clusters and particle flow objects
  - Modify clusters (adding hits, merging, deleting)
  - Access the current lists of Pandora objects
  - Save new lists of clusters, calo hits or tracks
  - Run a daughter algorithm, etc…

- Static helper functions are provided to perform tasks that are useful to multiple algorithms, and the Pandora algorithms are configured via xml and can be swapped in/out without recompiling.

Pandora Plugins

- PandoraPFANew has been designed to make it easy for people to get involved and try out new ideas. Users can therefore register and use their own content, including:
  - Particle identification helper functions,
  - Hadronic and electromagnetic energy correction helper functions,
  - Particle flow algorithms, allowing for a completely different reconstruction.



- Pandora API is used to pass addresses of helper functions and algorithm "factories" to Pandora managers.
- This gives Pandora the ability to call the static helper functions and to create and run instances of the user algorithm.

- Existing Pandora algorithms and functions have been written to ensure they have no dependencies and that they are (subject to changing steering parameters) largely detector independent, but this need not apply to custom content.
- This provides an opportunity to bring together many independent analyses within Pandora.

Pandora Plugins

Particle Identification Functions

1. Implement particle id function:
```
static bool MyClass::MyParticleIdFunction(const pandora::Cluster *const pCluster);
```
2. Register function with Pandora under a specific name:
```
PandoraApi::RegisterParticleIdFunction(pandora, "MyFunctionName", &MyClass::MyParticleIdFunction);
```
3. In PandoraSettings xml file, assign named function to one of a number of particle id 'slots':
```
PhotonFast, PhotonFull, ElectronFast, ElectronFull, MuonFast, MuonFull, ...
```
4. Algorithms call the helper functions, e.g. `ParticleIdHelper::IsMuonFast(pCluster);`

Currently available: Fast photon and electron id, reproducing old Pandora performance. New fast muon id function.

Energy Correction Functions

1. Implement energy correction function:
```
static void MyClass::MyEnergyCorrection(const pandora::Cluster *const pCluster, float &correctedEnergy);
```
2. Register function with Pandora under a specific name:
```
PandoraApi::RegisterEnergyCorrectionFunction(pandora, "MyFunctionName", HADRONIC/ELECTROMAGNETIC,
        &MyClass::MyEnergyCorrection);
```
3. In PandoraSettings xml file, specify ordered list of electromagnetic and hadronic correction functions.
4. Algorithms query Pandora Clusters for `CorrectedHadronicEnergy()` or `CorrectedElectromagneticEnergy()`.

Currently available: functions to remove effects of calo hits with anomalously high energies, coil energy loss correction.

Pandora Plugins

- Creating a custom algorithm allows access to the full range of functions in the PandoraContentAPI. This allows implementation of almost any conceivable particle flow algorithm.
- Can write new algorithms to complement existing Pandora reconstruction, or can write simple 'wrapper' algorithms to bring results of other packages right to heart of Pandora reconstruction.
- For example, can very simply (~50 lines of code) use output from e.g. GARLIC to replace Pandora photon-clustering stage:

Particle Flow Algorithms

1. Copy and rename Pandora template algorithm class, then register new algorithm with Pandora:
   ```
   PandoraApi::RegisterAlgorithmFactory(pandora, "MyAlgorithmName", new MyAlgorithm::Factory);
   ```
2. Add algorithm to the PandoraSettings.xml file; it will then be called automatically for each event.
3. Algorithm reads existing photon cluster collection, then uses unique identifiers in Pandora CaloHits to simply recreate the photon clusters within the Pandora framework.
4. Algorithm uses simple API call to save new photon clusters in a named cluster list, and to remove them from subsequent reconstruction.
5. ClusterPreparation algorithm configured to add photon clusters back into reconstruction when desired; they are probably best added at Pfo construction stage.
6. Should observe an immediate improvement in jet energy resolution.

Algorithm Tuning

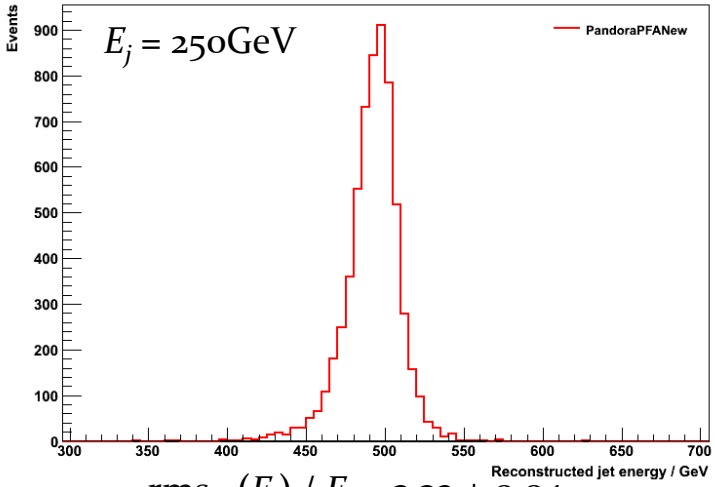
- The Pandora framework is designed to make algorithm tuning painless. Algorithms can be swapped in/out without recompiling and there are no hard-coded numbers; all parameters are configurable.
- The parameters are divided between the user application and the PandoraPFANew library:
  - User application parameters are those required to isolate PandoraPFANew from specific detector and software framework details. These include calibration constants and specific track quality cuts, etc.
  - PandoraSettings.xml file specifies which algorithms are to be used each event and configures these algorithms. It can override any of the 800 default parameter values and is independent of user application.
- The majority of parameters in Pandora take default values, compiled into the code. Most parameters are member variables of the algorithms and are commented in the algorithm header files:
  - `float m_tanConeAngleECal; ///< ECal tan cone angle used to calculate cone approach distance`
- Each algorithm has a ReadSettings method, called at startup. In this method, parameters are assigned their default values and the relevant section of the xml file is scanned to see if values have been overridden:

```
m_tanConeAngleECal = 0.3f;
PANDORA_RETURN_RESULT_IF_AND_IF(STATUS_CODE_SUCCESS, STATUS_CODE_NOT_FOUND, !=,
  XmlHelper::ReadValue(xmlHandle, "TanConeAngleECal", m_tanConeAngleECal));
```

- The xml helper will look for the key shown below. The original value will be unchanged if this key isn't present:
  - `<TanConeAngleECal>0.2</TanConeAngleECal>`
- In addition to algorithm settings, there are also static members for Helper classes, and a few "global" settings which are members of the PandoraSettings singleton.

Algorithm Tuning

- Algorithms specified within the <pandora></pandora> tags in the PandoraSettings.xml file will be called for each event. The algorithms are called in the order they are specified:
  - <algorithm type = "Clustering"/>
- For simple algorithms, using default parameter values, this simple configuration is sufficient. However, most Pandora algorithms are more complex and many different settings are specified for each algorithm, including nested "daughters".
- The parent algorithms dictate the xml configuration details that must be provided for any daughters. Algorithms can ask for named lists of daughter algorithms, or for individually labeled daughter algorithms. This offers much flexibility, and is discussed further at: http://www.hep.phy.cam.ac.uk/twiki/bin/view/Main/PandoraPFAQuestions
- So far, tuning has aimed to optimise the jet energy resolution, obtained as described below:
- Use MC samples of approximately 10,000 $Z \rightarrow uds$ generated with the Z decaying at rest, with E_z = 91.2, 200, 360 & 500GeV.
- Performance is quoted in terms of rms_{90}, defined as rms in smallest range of reconstructed energy containing 90% of events. A cut on polar angle is applied to avoid barrel/endcap overlap region: $|\cos\theta| < 0.7$
- Total energy is reconstructed and jet energy resolution obtained by dividing total energy resolution by $\sqrt{2}$.



$rms_{90}(E_j) / E_j = 3.32 \pm 0.04$

Summary

- The first release of PandoraPFANew (and its associated MarlinPandora processor) is now available. This is a complete replacement for the old version of Pandora and all users are encouraged to upgrade.
- The PandoraPFANew framework offers a great deal of flexibility for investigating new ideas and, in particular, allows for custom energy corrections, particle id and implementation of custom algorithms.
- 'Wrapper' algorithms can very quickly bring results from other packages right to heart of Pandora reconstruction.
- The existing particle identification and energy correction functions are quite simplistic and are only placeholders. We are looking for new ideas, so please get in touch if you'd like to be involved…
- If there is sufficient interest, we are thinking about holding a Pandora workshop in Cambridge in September. Would aim to provide a more detailed description of how Pandora works and discuss all you need to know in order to develop within the Pandora framework. Hands-on sessions; discussions concerning future development, etc.