Introduction to D0 offline analysis

- I will concentrate on the issues relevant for running existing code.

- Outline:
  - introduction to the RCP system
  - introduction to the D0 framework
  - running an existing Ntuple maker (Heidi’s tutorial)
    - interactive
    - using SAM
  - running the fast simulation
The Run Control Parameter system

- RCP scripts are used to control the execution of the framework and of the framework packages.

- RCP scripts can contain
  - `bool`, `int`, `float`, `(double)`, `std::string`, `RCP`
  - `std::vector` of those except `bool`

- Referred to a `<package name>` or by RCPIID (identifier)

- RCP are entered in databases:
  - `global` / `read only`: made by the build system
  - `personel` / `read–write`: that you can manage

- [http://cdspecialproj.fnal.gov/d0/rcp/](http://cdspecialproj.fnal.gov/d0/rcp/)
Exemple of an RCP script

```c
string PackageName = "JetReco"

string data_type = "MC"
string algo_type = "calorimeter"
float ETmin = 8.0
RCP clusterer = <calreco CalCone07>

bool SkipTracking = false
string TrackMatchTo = "JetCells"
float TrackMinP = 0.1
float TrackMatchDR = 0.15

bool SkipPS = false
float PSMinE = 0.01
string PSMatchTo = "JetCells"
float PSMatchDR = 0.15
```
Introduction to the D0 framework

- The framework defines hooks you can register to so that your code is called at predefined moments of the processing ⇒ hook list

- You write a framework package that implements one or several hooks, i.e. a class that inherits from hook classes.

- There are standard framework packages ⇒ package list

- Each package is controlled by an RCP script (see above).

- The execution of the framework program is also controlled by an RCP script.
Event–Oriented Hooks

- **Generator:** Construct a new event.
- **Merge:** Merge events from multiple generators.
- **Decide:** Modify framework work queue.
- **Builder:** Modify event.
- **Filter**: Optionally skip processing for this event.
- **Process:** Modify event.
- **Analyze**: Analyze event.
- **Finish:** End of event processing, e.g. flush Ntuple
- **Dump:** Produce an ascii dump of an event.

* Read only access to event.
Non–Event–Oriented Hooks

- **RunInit**
  - Called at beginning of run.

- **RunEnd**
  - Called at end of run.

- **JobSummary**
  - Called at end of job.
Standard Packages

- Packages available in io_packages library:
  - **ReadEvent**
    - Read events from D0OM file.
  - **NewEvent**
    - Create an empty edm event.
  - **MergeEvents**
    - Merge events from several generators (copies chunks).
  - **DropChunks**
    - Drop chunks by name (builder hook).
  - **WriteEvent**
    - Write D0OM format file.
  - **DumpEvent**
    - Produce an ascii dump of edm events using method edm::AbsChunk::print.
A framework rcp script

string Packages = "geo read weight met cone jet anal"

RCP globalnt = <analyze NtupleMgr>
RCP geo = <calreco CalGeom>
RCP weight = <calreco CalWeight>
RCP read = <local ReadEvent>
RCP cone = <calreco CalCone07>
RCP kt = <kt_jets kT_jets_RUNI>
RCP met = <missingET MissingET>
RCP jet = <jetreco JetReco>
RCP anal = <jetanalyze JetAnalyze>
Next step: run an Ntuple maker

- http://www-d0.fnal.gov/~schellma/d0cpp/
If you want to know more

- Check the D0 computing and the D0 code management pages
  - [http://www-d0.fnal.gov/~schellma/d0cpp/](http://www-d0.fnal.gov/~schellma/d0cpp/)
  - [http://www-d0.fnal.gov/~schellma/runII_cvs/](http://www-d0.fnal.gov/~schellma/runII_cvs/)
  - [http://www-d0.fnal.gov/~cope/l3/L3mainpage.html](http://www-d0.fnal.gov/~cope/l3/L3mainpage.html)

- **framework documentation in framework/doc**
    - has a list of tutorials
      - mc++ and analyze (Ntuple maker)
      - updated framework and code development tutorial
      - more advanced reco package design tutorial
      - some tutorials in French...