An Overview of the DØ/CFT Software

- Where it is
- What it is
- What it does
- How you can help

Mike Hildreth

July 17, 2000
basic facts

- Most (all?) of the DØ code is written in C++
- It all runs on UNIX (offline) or NT (online)
- There is a very flexible “DØ Framework” for running any offline code
  - MC
  - reco
  - analysis

- All of this is still under construction!
Code Management

All “official code” lives in a code-management archive called CVS

- in order to work on code, you “check-out” the code you need, “check-in” when you’re done...
- system prevents collisions between different versions
- there is a weekly “release” of software, with a number like t01.06.00 ⇒ code development
- there are quarterly(?) stable, production releases ⇒ stable, but soon out of date
more code organization

The offline code is managed by the Algorithms Group:

- Level 3 - Amber Boehnlein, Dan Claes
- Alignment/Calibration - Harry Melanson, John Hobbs
- Silicon Tracker - Emanuela Barberis
- Fiber Tracker - Mike Hildreth
- Calorimeter (including CPS and FPS) - Marek Zielinski
- Muon System - Pushpa Bhat, Christophe Royon
- Global Tracking - David Adams, Herbert Greenlee
- Vertex finding - Meenakshi Narain, Guilherme Lima
more code organization

• Each of the coordinators oversees the development of code specific to their area

• For the CFT, “algorithms” includes
  ✷ Track Trigger algorithms for Level 1 and 3
  ✷ Level 3 Monitoring
  ✷ Calibration constants (calculation, use)
  ✷ Alignment constants (calculation, use)
  ✷ Detector Simulation
  ✷ Event Display
  ✷ Tracking Pattern Recognition and Reconstruction
yes, even more...

Each of these tasks is broken down into several different “packages”, which are logical sub-units of code:

* e.g.: CFT Simulation -
  - `sftdigi`
  - `sftdigi_evt`
  - `sftdigi_reco`
The sftdigi package has

BINARIES
CFTDoubletSimulator.cpp
CFTDoubletSimulator.h
CFTDoubletSimulator.sh
CFTDoubletSimulator_t.cpp
CFTFiberGeometry.cpp
CFTFiberGeometry.h
CFTFiberGeometry_t.cpp
CFTFiberSimulator.cpp
CFTFiberSimulator.h
CFTFiberSimulator_t.cpp
CFTReadoutSimulator.cpp
CFTReadoutSimulator.h
CFTReadoutSimulator_t.cpp

COMPONENTS
CftIdealGeometry.dat
D0DetSimBuilder.cpp
D0DetSimBuilder.h
D0DetSimBuilder_t.cpp
GNUmakefile
LIBDEPS
SFTDoubletSimulator.cpp
SFTDoubletSimulator.h
SFTDoubletSimulator_t.cpp
SFTFiberGeometry.cpp
SFTFiberGeometry.h
SFTFiberGeometry_t.cpp
SFTFiberSimulator.cpp
SFTFiberSimulator.h
SFTFiberSimulatorTest.h
SFTFiberSimulator_t.cpp
SFTMCTracks.h
SFTReadoutSimulator.cpp
SFTReadoutSimulator.h
SFTReadoutSimulator_t.cpp
SFTSegment.cpp
SFTSegment.h
SFTSegmentTest.h
SFTSegment_t.cpp
SFTSignal.cpp
SFTSignal.h
SFTSignal_t.cpp
VERSION
doc/
geometry.dat
The DØ Tracking System

• “Global Tracking” attempts to find tracks crossing all of the tracking detectors
  - CFT and SMT code performs local operations, like cluster finding, local pattern recognition.
  - Large overlap between pattern recognition and track-finding
  - Some regions of the detector require special attention (overlap, gap)
    ▲ custom code development
• Find all tracks above a minimum momentum, with minimum cpu time

• Many problems remain:
  ◆ overlap and gap regions not fully efficient
  ◆ code is way too slow
  ◆ some efficiency loss in the central detector when events get very messy
  ◆ ...

tracking
trigger algorithms

- Level 1 trigger has fast local track finding built into hardware processors
  - tracks in 80 trigger sectors found in 500 ns
    - passed on to muon system, Silicon Track Trigger
  - used in Level 1 trigger decision

- Level 3 trigger has fast full reconstruction of tracks in CFT and SMT
  - can find tracks in 10-20 ms/event
    - no error propagation, treatment of material
Level 3 Monitoring

- Monitor quality of CFT data online
- check
  - electronics behavior
  - overall quality of CFT data
  - online calibration (maybe?)
- uses histograms made from Level 3 data stream
Calibration

Big project!

- Keep track of individual channel properties
- Software needed to
  - process data from LEDs to calculate gains, pedestals
  - store calibrations in online database
  - store calibrations in offline database
  - retrieve calibration info, apply to raw data so that reconstruction is “calibrated”
Alignment

- Absolute and relative position of CFT/parts
- “As-built” detector needs to be put into the software
- code for checking measurements with tracking data
  - do the analysis once we have tracks
- MC code for simulating geometric imperfections in CFT
Simulation

• create accurate representation of the physics of signal-generation in the CFT
  ◆ photon statistics, propagation, electronics
• Need to make MC match the real detector data
  ◆ put in real # of photons, electronics behavior
  ◆ some problems with underlying MC simulation
event display

- Allow events to be viewed
  - for PR
  - for debugging!
- No one working on current version
  - massively graphics-intensive
- Older, simpler version could be revived
  - could be worth some effort
so...

- Lots of things to do
- not many people working
- assistance welcome!