Commissioning the CDF Offline

presented by Liz Sexton Kennedy, Fermilab

* Scope and Scale of the Project
* CDF Offlines People
* Releases and Transitions
* Collaborations with Others
* Development of Stable Operations
* Surprises
* Offline is a Success
* Lesson's Learned

`

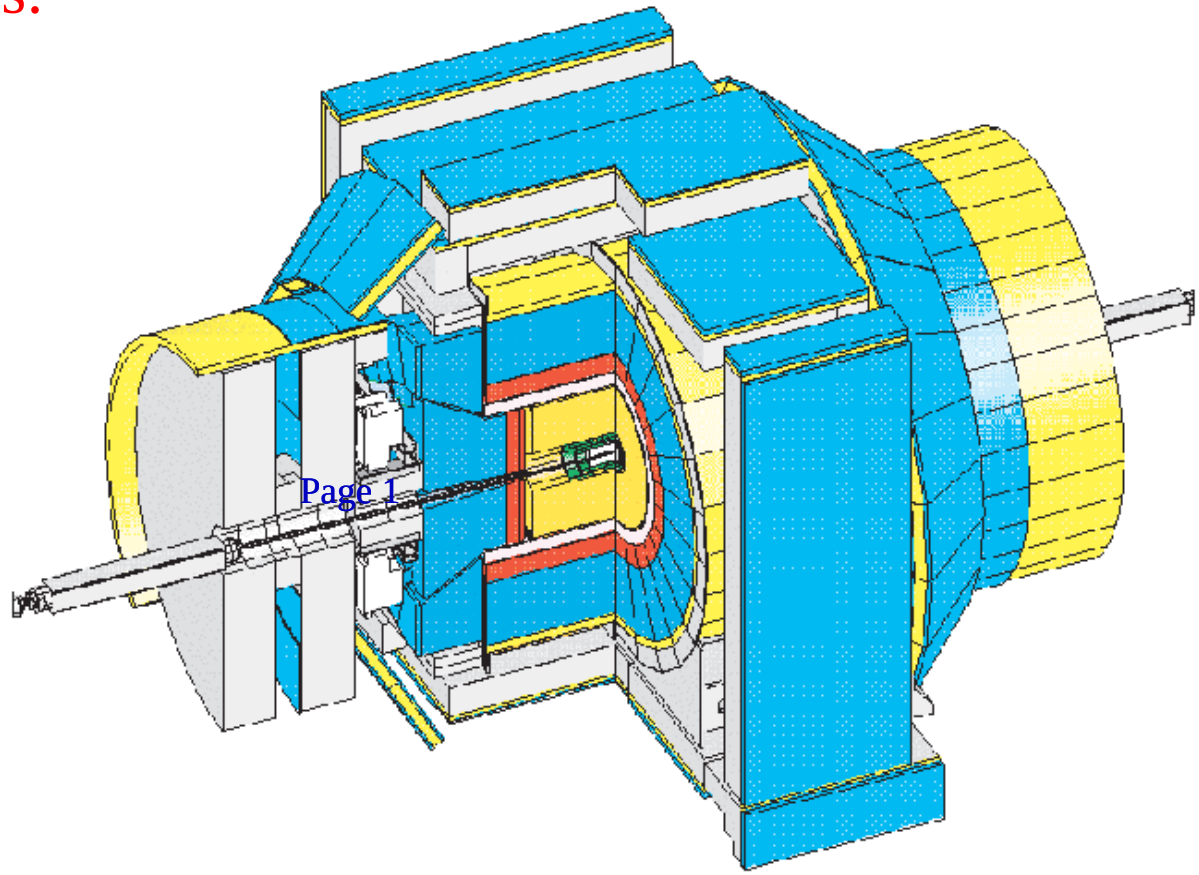
CHEP 2003 San Diego, CA
March 27, 2003

Liz Sexton-Kennedy
28-Mar-2003

Scope and Scale

At CDF Offline includes:

* Level3 Trigger
* Online Monitoring
* Primary Reconstruction
* Simulation
* Tools (evd, performance)
* Physics Group Analyses



This involves 10's of millions of lines of code in 294 packages.

Major systems map to the reconstruction groups Tracking, Calorimetry, and Muon, there are smaller groups for Time of Flight and Luminosity counters

CDF Offline's People

In '96 when we started very few knew C++ we claimed we would use a mixture of C++ and f77 to ease people into it.

For most people entering the project this was their first experience writing C++ code.

CDF is a conservative group. There was a mandate to reuse as much run I code as possible.

You never completely understand a problem until it is solved once , so this was not a bad strategy.

The pioneers started by transcribing f77 to C++. Almost all of that code has been restructured or replaced entirely.

Of order 6 to 10 people are highly productive at the same time, one per subsystem. There are never enough people to do the job you want done.

In 2003 the only Fortran is in generators and other outside products. Very few want to go back to f77. Every subsystem has been written at least twice, each time the rewrite was accompanied by people changes. We now have people entering with prior C++ experience.

Liz Sexton-Kennedy
28-Mar-2003

Releases and Transitions

The release schedule reflects the pace of development and management policies.

3.0.4, 3.3.4, 3.5.4, 3.6.7, 3.11.1, 3.15.0, 3.18.3, 4.0.1, 4.3.2, 4.8.5, 4.9.1e, 4.10.1 version

4/99, 12/99, 6/00, 7/00, 12/00, 4/01, 7/01, 9/01, 1/02, 8/02, 11/02, 3/03 date

Time	Detector commissioning	Software commissioning
* Fall ’99	* Cosmic Run with most of the detector	* f77 array Edm -> root based Edm
* Winter ’00	* Mock Data Challenge I (MDCI)	* Calor/Jet rewrite
* Spring ’00	* Calibrations runs	* Geometry x2, Simulation framework * First Doc of full system
* Summer ’00	* MDC2	* Muon rewrite, Extrapolator started * cdf_software_help@fnal.gov
* Fall ’00	* First Beam full detector	* introduce persistent parameters (rcp)

The system proved flexible, even under the heavy loads.
Transitions occurred when personnel changed.

Collaborations with Others

These are our major collaborators:

BaBar

Donated their framework code, ideas about tracking used by Si., and EvtGen
Fizzled out when BaBar started taking data and we commissioned.

Root

Clear from the beginning that we would be co-developing with the root team. Involvement of FNAL core developer (Phillippe Canal) was essential. Many of the important features of root (StreamerInfo) were not existent when we started and it was too late to adopt them when they came out. Some performance features were CDF motivated (TBuffer I/O).

Zoom/CLHEP

Use many of their packages "off the head". Commissioned works that integrate well at CDF. Lots of useful performance collaboration

KAI

Chosen because it allowed us to code to the C++ standard (use for 10 years). It was being co-developed. There were a few painful transitions to new versions. In Apr-2000 it was acquired by Intel, in the following fall the last version came out.

Development of Stable Operations

In the Fall of ’01 we started defining the offline "shift".
Two main goals for shifters:

1. Help in integration/release task. Improve robustness of result and validate it.

   Shifter given recipes for running purify, debuggers, and software management tools.
   It’s also part of the job to continually update and refine the shift documentation.
   Regression tests are a big part of the job.

2. Answer the questions of the rest of the users through the cdf_software_help@fnal mailing list.

   Started with developers but quickly moved to random collaboration members.
   This works because we use the freeware JitterBug tracking system.

Surprises

1. Memory leaks are supposed to be the biggest problem however uninitialized variables is a bigger problem.
THE DREADED NAN, code instabilities.

2. Inside CDF:
Can't you make the reconstruction faster? Performance is a problem. Complain about I/O (200kB/ev) We don't use compiler optimization yet. Simulation geometry not optimized yet.

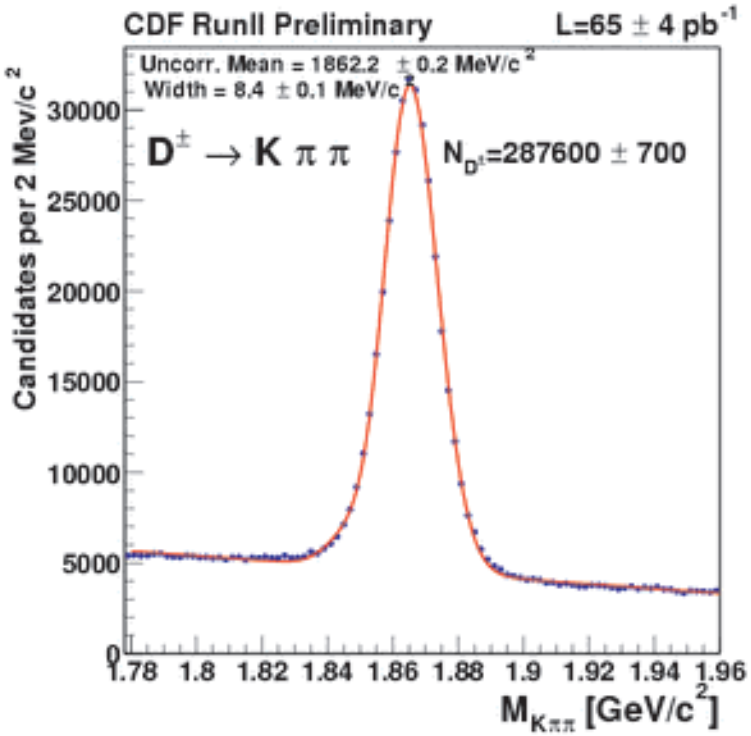
3. Outside CDF:
Why is your reconstruction so fast? (<3sec./event goal was 5)
Capture of successful run1 & TDR algo. in Calor/Jet and Outer tracker systems. Constructive competition among Si. algo. developers.

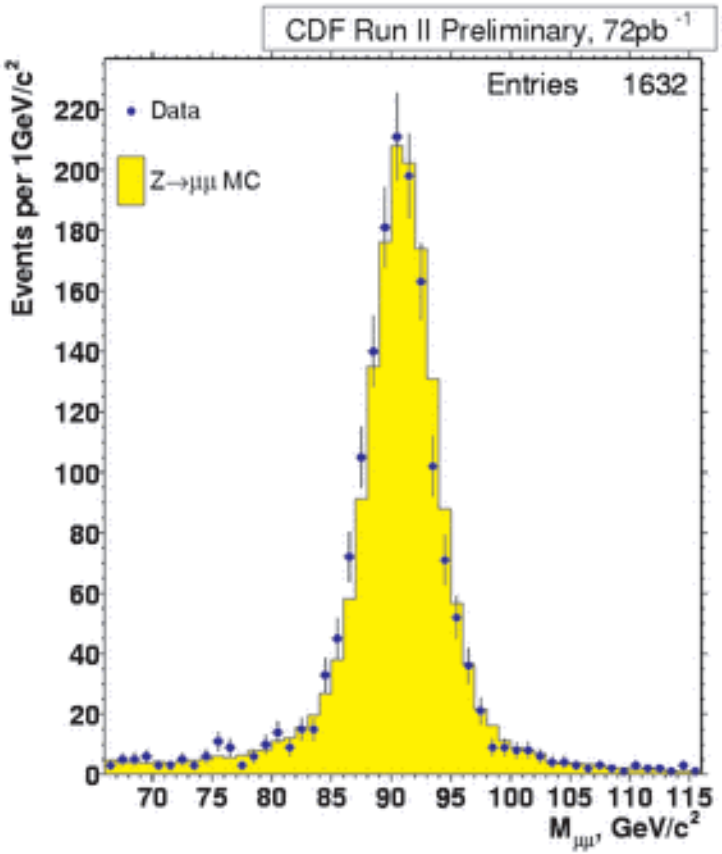
Offline is a success

Data collection to physics presentation was 6 months.

CDF has it's first publication!

Measurement of the Mass Difference m(Ds+)-m(D+) at CDF II





Lessons Learned

1. Start commissioning as early as possible. MDC's didn't completely prepare us. Count on having to change things once you have real customers with real needs.

2. People will leave one year after first physics data arrives. It's then hard to get the last bits done.

3. Beware of code generation, it can produce code bloat.

4. Keep the system clean in terms of physical design and organization. Physics analysis codes will follow the patterns of the reconstruction and be more generally usable if done so. Code browser is important.