1 Data Access Requirements

In order to get an idea of data access requirements it is necessary to understand the data rate and data size that we need to access. Especially in the online environment the data rate is a concern while in the offline environment the ability to handle large data sets is important.

The input rate to the CDF L3 can be or the order of 300 Hz. The maximum output rate is about 75 Hz. With an event size of approximately 150-250 KB we expect data rates of about 15 MB/s written to mass storage.

The total raw data size is estimated to be about 250-500 TB for CDF Run II. It is anticipated that up to five different reconstruction passes may be required. The output of a reconstruction pass is expected to be about 130-260 TB. Concurrent access to old and the new reconstruction versions will be allowed for a three week transition period. The reconstructed data will be subdivided into many data sets for easy access. The exact number and size of the sets is still under discussion. Since large data sets will be analyzed the speed at which the histogramming package can access the data and make selection cuts is important.

We had file size limitations when using HBOOK and one had to explicitly set parameters to allow larger sizes. In Run II we expect the need to be able to write large data sets (both data and MC). We need the capability to be able to concatenate multiple files with a similar format. In PAW this was accomplished with either MERGE or CHAIN. The functionality of the CHAIN command which logically links multiple files to allow easily plotting of quantities across the files is a powerful feature.

We can consider two modes of use. The first is making histograms directly from an event record. In this mode the user needs to link their analysis program to a library containing the histogramming functionality. The second mode is examining the data interactively. In this one one will produce a “ntuple” equivalent summary file which is then accessed via the histogramming package. One will also need the capability to write stand alone programs to read and analyse data in the “nutple”. The package we choose should be able to run in these two modes.

In the “nytple” mode we need to be able to read in part of an event. For example the histogram data can be grouped together logically and one may be interested in only a part of the data to be histogrammed. For example your “ntuple” may contain MC true values and MC smeared values. Part of your analysis may require only the MC smeared values and the program which reads in your “ntuple” can efficiently skip over the MC true values. If your “ntuple” contains 100 entries and you are plotting only one you do not want to have to read in all 100 entries. The histogram package should be able to quickly skip over records and be able to get to particular events efficiently.

Need to be able to save events with variable number of entries, i.e. variable size arrays. One may want to be able to store multiple vertices, results from different jet algorithms, etc.

1.1 Specific Online Requirements

Online monitoring can be divided into several categories. The monitor processes can request specific events from the event server.

- Detector Performance
  During running one needs to check the readout electronics. One needs to identify dead channels, broken front end cards, tripped power supplies, etc. Typically one wants to plot the output of each channel and compare to reference histograms to search for drops in response. The rate of events which are histogrammed can be low (0.2 Hz) but the number of histograms and/or channels per histogram can be large.

- Trigger Performance
  Simulate the hardware trigger and check for discrepancies. Want to sample a fraction of passthrough events to test the performance of each trigger stage.

- Luminosity Monitoring

- Physics Monitoring
  One wants to monitor physical quantities such as reconstructed E_T, mass of dileptons, etc. In this case the total number of histograms is less than required when monitoring the detector performance. More sophisticated analysis tools such as user or predefined function fitting, goodness of fit estimates, etc. will be required. Results of the fit will need to be made available to the user.

- Event Display
  Send an event record to the event display process. Would like to be able to produce a lego plot for example or histogram activity of some detector.

- Data Logger
  Each event output from the L3 is sent to mass storage.

Each of the monitoring processes have similar functionality requirements. The extreme case in terms of the number of histograms and channels per histogram occurs when monitoring the Detector Performance while highest event rate occurs for the Data Logger. Table **??** lists the number of channels for each detector component which needs to be monitored online.

Detector	Channels
COT	30240
CCAL	1728
WCAL	576
PCAL	1824
CSMX	11232
PSMX	10560
SVX	405504
MUON	8003

Table 1: *Total channels for each detector component.*

Need to be able to access/analyze data in real time. Needs to be fast in order to keep up with the data flow. Needs to be robust - cannot have crashing code. Access shared memory. Depending on the implementation one may want the possibility to receive histogram data from several remote clients and accumulate the data into a single histogram. The monitoring programs must be debugged and fully functional at the start of data taking. The resultant histograms and output files should be easily accessible by multiple users from remote sites without disturbing the main monitoring program activities.

To be able to use the histogramming routines from a user program typically written in c++ and/or FORTRAN. We need to be able to link to a library containing the histogramming routines. Need to be able to output a summary file containing the generated histograms which can later be viewed. The monitor code should have the ability to be able to compare the accumulating histogram with a reference histogram. Methods should be available to identify discrepancies in the histograms, i.e. determine the χ^2 between the old and new histogram.

The ability to zero a histogram and restart accumulation is needed.

1.2 Large Data Sets

When interactively running over large data sets it is important to have an efficient method to select events. Sometimes the selection criteria can be quite complicated. In the most general case the plotted quantity can also be a complicated combination which might require calling outside functions. The interpreter which calculates the cut condition and the plotted quantity must not be time consuming.

Typically one plots one or two quantities for a given data set. The mechanism to select the quantities to be plotted should be fast. One can also have data selection masks to speed up the selection process.

1.3 Summary

- We need much of the functionality already present in PAW.
- The ability to create *many* histograms with *many* channels.
- Fit the histograms to complicated predefined or user defined functions.
- The ability to perform a statistical analysis of the histograms.
- The ability to add/subtract/scale histograms and compare the shape of histograms.
- The ability to interactively generate histograms.
- The ability to link to a library of histogram routines and write out a histogram summary file.
- The ability to efficiently analyze large data sets.
- The ability to plot results obtained from multiple files.
- Easily analyze part of the "ntuple".
- Efficiently skip to the variable you are interested in i.e. if you are plotting E_T you don't want to read in all variables of each event record.
- Store variable size structures.
- The ability to plot complicated combinations of the stored variables. You might create user defined functions and plot the functions after passing the "ntuple" variables.
- The ability to create complicated event selection functions.
- The ability to collect histogram results from many different sources and make them available to a remote client.
- The remote client should be able to view the histogram with out affecting the histogram accumulation.