

Conceptual Design for Offline Data Analysis and D0 Streaming

Performing analysis of data using SAM requires a way of easily and unambiguously specifying the data sample to be used and then accurately computing the luminosity of the data sample actually used. In order to do this a number of important assumptions about the way the D0 online and Trigger systems operate must be made. In addition, accurate bookkeeping information about events, files, triggers, streams, luminosity and the processing of data must be kept in the D0 databases. In this paper we explain how this will be accomplished.

We also outline the work steps needed to put this Offline Analysis system in place.

Vicky + all
who wish to contribute

August 8, 2001

1. Introduction

To fully understand how the Level 3 system and the D0 online and data logging system handle triggers, L3 filters and filter scripts, and Physical Streams a companion document must be read in detail. [x]

A number of assumptions and assertions must be made about the workings of these downstream systems that write the raw data into neat little packets (files) ready to start its journey through the processing steps needed for analysis. We will start by laying out the assumptions we make and therefore the ‘rules’ that we expect the L3, luminosity, trigger database, run configuration and online systems to be following. The success and accuracy of the analysis system outlined is dependent on information that is recorded by these systems in trigger, event catalog, luminosity, stream and run configuration databases, as well as dataset and full processing history of data that is recorded in SAM.

2. Goals and Requirements for Analysis

The primary purpose of classifying data into various *Physical Streams* is to guarantee that an analysis, based on a specific *Trigger*, will not have to process every data file in order to gain access to the events of interest. It may also be used to arrive at a small sample of data, with events of particular interest, for immediate rapid processing.

The primary purpose of grouping data files from certain Physical Streams together on a certain tape, or *Family of Tapes*, is to ensure that an analysis will not have to, in effect, access every tape in order to gain access to those files (of particular Physical Streams) containing the events of interest.

The primary purpose of caching data on disk in many locations, both at Fermilab and worldwide, is to minimize the wait time for data to be made available for a particular analysis or processing activity and to minimize the number of times that tapes are mounted and data is read from tape. Managing disks by physics group and allowing each group to specify its own caching policies and desires for ‘pinning’ of certain data on disk is designed to further optimize access times and minimize use of tape and network resources by, for example, segregating the movement of extremely large datasets through disk cache, from residency of smaller static datasets in cache. The strategy of multiple distributed disk caches, as used in the SAM system, may also serve to minimize loss of data due to tape media errors.

The effectiveness of all three of the above mechanisms in ensuring that reprocessing of data and physics analysis can be carried out without overloading the robot, tape drives and network, (and without unduly frustrating the physicist doing analysis) depends on the success of the Physical Streaming algorithm in *clustering Triggers of interest into a small number of Physical Streams*. It also, to a lesser extent, depends on the success of

clustering certain “like” Physical Streams that contain many of the same Triggers onto physical tapes. If we have a robot capable of many hundreds of tape mounts an hour, then this latter clustering of Physical Streams into a Tape File Family is much less important. **It is important, however, that both of these clustering algorithms can be changed and evolved as the experiment proceeds.**

The goal of *Production Processing* of the data is to **reliably pass all data through known and validated reconstruction and processing steps in order to provide summarized data of known content and quality at each stage of processing.** Production Processing of data may produce new refined Physical Streams via a Level 4 or higher Filter processing step. The summary data that results from each processing step in the chain of production processing is termed a *Data Tier*. Higher tiers of data, containing progressively more refined summary data from each event, are produced as the data passes through each step in the Production Processing Chain. With the SAM system neither the steps in the processing chain, nor the data tiers are fixed. **New types and numbers of processing steps can be added at will, and new data tiers can be specified at will.** The file format of data (e.g. EVPACK, ROOT, other?) is flexible and new file formats may be handled as and when needed by the experiment.

SAM must make it easy and intuitive for someone doing analysis, based on data provided through experiment-wide production processing, to specify the data required. For example, they should be able to give as little as a *Trigger name* and a *Data Tier* (e.g. Raw, Reco, Thumbnail, Root-tuple or other to be defined) in order to specify their input dataset. They must then be able to run their analysis application to read only the relevant data files, normally using several different jobs run at different times. **They must end up with a full history of what was processed, together with the luminosity of the data sample processed. They should be assured that no events have been processed more than once.** They should be able to optionally specify further constraints on the input data set to exclude certain Run ranges (e.g. exclude certain explicit Runs, or those where a certain condition occurred or the integrated luminosity was below a certain threshold), or constraints on the chain of processing steps by which the data of that data tier was produced (e.g. only Reco Version p13.02 or higher). They must be able to transparently write out and store back into SAM the data files that resulted from their processing and which are intended for future repeated analyses by themselves or their Physics Group. The *processing history* by which they produced these files (application, version, when, where, etc.) must be *recorded automatically*, without danger of error or compromise, and the full processing history for each file in the input dataset must be recorded. This must include whether the file was successfully delivered to the analysis program, whether it was successfully read, whether events were written out as a result and whether the files written out were successfully stored back in the system. This is true for all stages of analysis/data processing.

SAM/Production Processing must provide the tools for bulk production processing of data through Production Processing chains. **Therefore processing jobs that will take a Primary Input Dataset of RAW data through all stages of processing must be easily**

generated and run, taking into account previous processing history information kept by SAM. This includes jobs that reprocess data or process missing data or otherwise compensate for errors and imperfections in the whole chain of processing. We must be assured that no data is duplicated for a particular defined processing chain and that all files that were not successfully processed, or not successfully stored, have been accounted for.

For analysis of specific runs and types of data, such as during commissioning and detector monitoring, users must be able to specify the input dataset that they wish to use in detail, based on constraints that may involve all possible information available about the nature of the data – from File, Stream, Trigger, Luminosity and Run information recorded in the database. Data thus specified is not intended for physics analysis, but is rather an open query for data that matches broad criteria. **It is the responsibility of the user in this case to ensure that the data sample specified makes sense, does not contain duplicate files or events, and does not mix data from different production processing chains.** This type of specification of an input dataset for processing is all that the current SAM system supports.

3. Assumptions

In the following the term “*Trigger*” is used to represent the name of the Trigger in a trigger list – a name that has associated with it Level 1 and Level 2 trigger terms as well as a L3 filter script consisting of possibly a number of L3 filters.

e.g. [add example couple of lines from real Trigger List]

3.1 Mappings for Trigger to Stream and File to Tape File Family

Since the number of possible combinations of Triggers is extremely large it is impractical to write every unique combination of Triggers to its own separate *Physical Stream* of data. Therefore a compromise grouping of events into Physical Streams of events that share certain physics characteristics has been proposed [Ref x]. The L3 system is responsible for making that mapping. The proposed algorithm involves maintaining a mapping of the Stream Primitives (e.g. ELECTRON, LOWPTMU, etc.) associated with each L3 filter. For each Trigger List used a Stream Primitives to Physical Stream decision tree mapping will be stored. This decision tree is used by L3 to map from the set of Stream Primitives associated with the L3 Filters that the event passes (a Stream Primitives bitmap) to the Physical Stream that the event is to be written to. The details of this mapping and its proposed initial implementation in terms of Physical Streams is described elsewhere [Ref x]. Along with each decision tree a mapping of each Physical Stream back to all potentially occurring *Triggers* (by name) is produced and stored. For each physics run the following complete set of information and mappings will be stored: -

- Trigger List and version

- Map from L3 filter script to set of Stream Primitives, for each L3 filter script (*and I suppose we also will have this mapping for each L3 filter itself, from which the L3 filter script mapping will be constructed?*)
- Name, description and type of all Physical Streams. Currently we see at least three types of Physical Streams – Physics Stream(RAW data), Monitoring Stream and Physics Stream(processed data).
- Stream Primitive to Physical Stream decision tree mapping
- Physical Stream to Trigger Names reverse mapping
- Trigger Name to Trigger bit number mapping (as assigned by COOR at start of run). (*also L3 filter bit to filter name mappings ? or are these fixed?*)
- Physical Stream to Tape File Family mapping.
- Tape File Family parameters – e.g. number of tapes to allow writing to simultaneously, algorithm for organizing data into directories (limits, naming conventions, etc)

It is a fundamental assumption that the Physical Stream to Trigger Names reverse mapping can be constructed, and this relies on the fact that L3 filter scripts, that combine various L3 filters, do so by requiring that an Event that passes the L3 filter script passes ALL of the L3 filters in the script.

3.2 Exclusive Physics Streams and Monitoring Streams

Events are assigned by L3 to one or more Physical Streams based on the Stream Primitive to Physical Stream decision tree mapping discussed above. Physical Streams may be either Physics Streams or Monitoring Streams. **An event may NOT be written to more than one Physics Stream.** *Physics Analysis* and *Production Processing* of the data **work only on data from one or more Physics Streams.** Events once assigned to a Physical Stream are written into a file by the Online. **Each file belongs to just one Physical Stream.**

Note: If we were to decide to write have some form of non-exclusive streaming, where we also wrote some events to an Express-Line Stream, for example, then that would be OK, as long as normal analysis was restricted always to data from Physics Streams.

3.3 Opening and Closing of Files and their relation to Luminosity Blocks

The online system is responsible for writing events, tagged for a specific Physical Stream (Physics or Monitoring or both), into files. A file for a Physical Stream is opened sometime after run start, when the first event tagged for that Physical Stream is encountered. All files are closed at the end of run. In addition files are closed when certain conditions are met including size of file reaches watermark size limit or the time a file has been open exceeds a specified limit. As a file is written the lowest number luminosity block Lum_{min} for which events have been written to that file is noted and the

highest number luminosity block Lum_{max} for which events have been written is noted. **Except in extremely rare error situations, all events for a Physical Stream that belong to any luminosity block between (and including) Lum_{min} to Lum_{max} will be written to a single file.** In the rare error situation where this fails out of range events will be written to a special Error file. (What will we do about them?)

Although each file written is catalogued along with its Lum_{min} and Lum_{max} this gives only a close approximation for the time period of data collection covered by this file, particularly in the case of Physics Streams with a very low rate of events.

3.4 Luminosity Information

The luminosity database contains all information necessary to calculate the integrated luminosity over a Run, or range of luminosity blocks, for a particular Trigger. *(I hope this is true?? here I am using Trigger to refer to the Trigger name in the Trigger list – which has Level 1, Level 2 and Level 3 terms that make it up. This assumes information from the L3, as well as the L1 and L2 that are currently being recorded, is also recorded in the DB eventually)*

The luminosity database also contains a list of luminosity blocks for which luminosity information is suspect. Events from these luminosity blocks should not be included in any final luminosity-sensitive analysis.

3.5 Production Processing of Data

Official production processing of data passes all events through a well defined chain of processing steps. This processing is actually performed using files (which are merely the convenience containers for events). **The processed events from a file belonging to one Physical Stream are written to a file that also belongs to the same Physical Stream, or in the case of a L4 or higher level Filter Process, to a more refined Physical Stream.**

The relationship between Physics Streams (RAW data) and Physics Streams (processed data) will be maintained in the SAM database, including all of the additional Filter information for any L4 or higher filter processing. This higher level filter processing information will parallel the information and relationships kept for L3 filter processing.

3.6 Analysis of Data by Trigger

Physics Analysis is done by processing events of a specific Trigger. *(If the mapping between a Stream and the possible Triggers that may be found in that stream is known, and an event is never written to more than one physics stream, then I do not understand why a user may not specify an OR of two or more Triggers on which to base his/her analysis and process all such events in one processing job??)*

The translation between events of a specific trigger and files from a certain Physical Stream is done by SAM, based on the information recorded about the mapping between Physical Streams and Triggers. Files from one or more Physical Streams, that may

contain events of the requested Trigger are presented to the user via SAM's file request interface. (The order in which such files should be presented is still not well understood. Although Trigger names remain constant there may be different versions of trigger lists and variations in the details of the L3 scripts. All of this is recorded in the database, but it is not clear how this can be taken into account when specifying data for analysis).

We assume that the physics analysis process (I/O package code or user code?) filters out all events that do not contain the specific Trigger requested, and that are present in the Physical Streams. (How is this done exactly? The framework package would need access to the Trigger Name to bit mapping, since the information stored in each event is based on trigger bit numbers?)

4. Physics Analysis

4.1 Current Analysis Projects and Datasets

In the current SAM system a user job is specified by giving the user script or executable to be run on a specific Dataset. This is done using a "sam submit" command. The Dataset is a list of files that is the result of constraining the total set of files in SAM by a set of conditions – e.g Data Tier, Run number range, Stream, dates taken, etc. In addition more complex conditions may be used to constrain the file set that makes up the Dataset, including conditions on whether the file has already been analyzed by a particular application and version or whether it is in the union or intersection of other defined datasets.

Processing of Data currently consists of nothing more than processing of a specific set of files. The intended set of input files is known and recorded and the actual set of analyzed files is also known and recorded. Any output file that is written as a result of processing data using SAM, references the input file or files from which it was derived as well as the actual job or jobs that ran to perform this processing. In the case where processing was not done by a job getting its files from SAM (as in the case of a MC chain of processing) then the output file still references the "parent" files, and thus indirectly the whole chain of parent files, but does not reference a specific job that was run. Details of the Application/Version (e.g. d0gstar, d0sim, etc.) and its parameters are in this case recorded in the metadata of the MC file.

4.2 Future Physics Analysis Jobs and Datasets

Extensions to the current system will be needed in order to handle physics analysis by Trigger in a transparent way and to provide accurate luminosity information for a physics analysis.

We propose to identify Physics Analysis Datasets by the following: -

- a) the primary event data as specified by a Run or set of Runs and a Trigger (or Triggers See ??s above)

- b) the data tier to be used for the data
- c) the Physics Processing Chain that was used to process the primary event data through to the specified data tier

A Physics Processing Chain is a sequence of processing steps which each specify a set of allowed Application/Versions used in the processing of data and where the output data from one step of the processing is used as the input data to the next step of the processing. If multiple Application/Versions are specified in one processing step, then they are given a preferred order. Thus you could specify that Processing Step 1 on the chain must be one of the following: Reco Version p23.02.01, p22.01.05 or p22.01.03, with p23.02.01 preferred.

We propose to identify all production (i.e. official) processing jobs as belonging to a processing step of some Physics Processing Chain. Of course the entire processing for any Physics Analysis Dataset will be done by running many different processing jobs. We allow for the fact that not all processing jobs will use the same versions of Applications at all times by allowing a Processing Step to be defined by one or more official versions of the official processing application for that step. The list of such processing versions must be prioritized, so that files from the preferred version(s) of the processing application are used if they are available.

For an end-user physics analysis we propose to specify the analysis job to be done at a high level by describing only the Physics Analysis Dataset and the Application/version to be used to process the data. This high level “job” might actually be partitioned into several individual jobs that each act on a part of the Dataset. We need a term for this high level “job” – perhaps the term “An Analysis” is sufficient. The end-user must then be able to include and exclude the results of various individual jobs into the Analysis, with suitable checks to ensure consistency. For example, the individual jobs included may not process overlapping parts of the same Physics Analysis Dataset.

A Physics Analysis Dataset, An Analysis and A Physics Processing Chain are more dynamic concepts than the current Datasets, Analysis Projects and Applications of SAM. They represent aggregates of these initial concepts that can evolve over time. In particular Physics Analysis Dataset specifically refers only to the RAW data by Run ranges and to the data tier. It contains no mention of Files and has no capability to include or exclude specific files into the Dataset. The determination of which Files to include is done through SAM, not through user selection at the file level.

4.3 Physics Analysis and Luminosity

Files are an important part of the analysis system in that they are THE way that event data of a certain data tier is delivered for processing and by which the results are output. They are also therefore the units of data which can get lost, be incompletely handled, be temporarily unavailable or otherwise excluded from a Physics Processing Chain or from

an Analysis. But, they are not fundamental in any way to the data processing or analysis processes. The fundamental units of data are Runs, Events and Luminosity Blocks.

We therefore propose to carry out all Luminosity calculations for an Analysis on the basis of the Physics Analysis Dataset on which it is defined - **using file-based accounting only to provide corrections to the Ideal Luminosity of the Physics Analysis Dataset**.

Every Job that is run either as part of a Physics Processing Chain or as part of an Analysis records the disposition of all files that it ‘consumed’ in the processing. This includes recording failure to fully consume the file (for whatever reason), and **recording that no output file was produced as a result of the processing, if that is the case**. In the SAM database the entire processing history of all files is thus known – permitting individual files to be either split or merged or filtered out. Files can thus be traced both forwards and backwards in terms of their parentage and their descendents.

When determining the Luminosity for an Analysis first the Ideal Luminosity for the Physics Analysis Dataset is determined. Then corrections are made in two different ways

- The total luminosity to be included in the final analysis is computed. This is done by tracing back through the parentage of files, back to the RAW data files from which they came, for all those files that were NOT processed but should have been in the ideal world. This is done for every stage of processing in the Production Processing Chain, for the Physics Analysis Dataset chosen. Along with the explicitly unprocessed files all affected files (e.g. Sibling files of a lost file, from a split parent file) that were processed, but cannot be in the final analysis are automatically also traced back to their RAW data file parents. The luminosity block ranges for these RAW files (containing events that will NOT appear in the final analysis) is then computed and any known ‘bad’ luminosity blocks, as recorded by the luminosity monitoring system, are added to this list of luminosity blocks to be excluded. The luminosity for the Trigger(s) of choice for these luminosity blocks can then be calculated and **subtracted** from the total Ideal luminosity for the Physics Analysis Dataset.
- Events that were possibly included in the Analysis, but should not have been, can then be identified using the known mappings in the database between Streams and Triggers, the mapping between Trigger bits and Trigger names, and the Event Catalog mappings between Event and Luminosity Block. This is a non-trivial computation. The list of such events is then used to reprocess the Analysis and exclude those events. Excluded luminosity blocks, because of files lost or not processed for one stream, then only cause exclusion of the minimum number of events (those in the missing luminosity blocks) in other streams containing the Trigger(s) used for the Analysis. They do not cause the jettisoning of whole files in other streams. In fact the exclusion of Events in the final reprocessing can probably be done on the basis of luminosity blocks alone, provided luminosity block number is one of the persistent pieces of information carried through all data tiers and included in summary event information at all levels.

This could, of course, become a non-convergent iterative process, but in fact it will not. The files for a final stage analysis, if they were to become available suddenly as reprocessing is being done, would simply be regenerated. The files from earlier processing steps in the Production Processing Chain might also suddenly become available thus permitting additional higher tier files to suddenly become available. However, in the end this process should not prove so dynamic that it will not converge rapidly. However, some thought is perhaps needed about how to 'fix' the input files for the final analysis during final reprocessing.

PUT DIAGRAMS IN HERE FROM PRESENTATION.

5. Work list

In order to implement the full system that will allow Physics Analysis and Luminosity tracking, using Triggers, Streams and Luminosity block information, a large number of small pieces of work need to be done and integrated.

THIS IS STILL VERY INCOMPLETE – needs flushing out

5.1 Database Designs

- 5.1.1 Database tables for Stream Primitive bitmap to Physical Stream mapping table
- 5.1.2 Database tables for Physical Streams and their hierarchical relationship in the case of L4 or higher Filter
- 5.1.3 Database table for Names (L1? L2? L3 Filter script?) to bit number mapping (as provided by COOR) – part of Run config database
- 5.1.4 Database tables for Reverse Mapping of Streams to their associated Trigger Names
- 5.1.5 Database tables for saving Versioned collection of Trigger List+Stream Primitive bitmap-> Physical Stream mapping + Physical Stream-> Trigger Names mapping
- 5.1.6 Database extension for linking Runs to collection in 5.1.5

5.2 Trigger DB tools

- 5.2.1 Command or GUI to trigger creation of instance of 5.1.6
- 5.2.2 Admin tools to define new stream names and to enter relationship of streams for L4 filter streams

5.3 Run Config tools

5.4 Luminosity tools

- 5.4.1 Extensions to Lum tables – for L3?
- 5.4.2 Tables to store summary information per run per trigger and possibly other summary info.

5.5 SAM processing chain extensions

- 5.5.1 Design of new entities – Physics Analysis Dataset, Processing Chain and Analysis (plus possible others that may emerge)
- 5.5.2 Restructuring of file parentage and reorganization of existing data

- 5.5.3 Definition of processing chain steps including a processing step that potentially contains a sequence of applications, but without storing knowledge of intermediate files in SAM.
- 5.5.4 Tools to include/exclude jobs in an Analysis, with checks
- 5.6 SAM Physics Analysis Dataset definition extensions
 - 5.6.1 Modification and extension of Dataset definition tool towards Analysis and Job definition tool
 - 5.6.2 Query pages for Analysis and Physics Analysis Datasets
 - 5.6.3 Query pages for Processing Chain and Jobs
- 5.7 SAM Job and Workflow tools
 - 5.7.1 Tools to create Production Processing Chain jobs
 - 5.7.2 Extension 'submit' tool to launch Analysis jobs
- 5.8 User code (I/O package code?) for excluding Events or Lum blocks
 - 5.8.1 Is this an I/O package extension? How does it get the bit number to name mapping?
 - 5.8.2 Write actual package for excluding events or lum blocks and server needed to support it.
- 5.9 Luminosity calculation tools
 - 5.9.1 Run summary calculations
 - 5.9.2 Luminosity of a specific Physics Analysis Datasets calculations
 - 5.9.3 Allow searching for data by luminosity
- 5.10 Sample Analysis Test suites and validation
 - 5.10 Simulation of effects of missing files/failed farm jobs
 - 5.11 Create validation test suite of data, processing and luminosity calculations