Notes from SLAC data handling workshop

Some notes and observations on the SLAC data handling workshop.

A two day workshop was held at SLAC January 13-14. The first day concentrated on the pipeline software, and the second day on the data server. In each case we first discussed what currently existed, then covered requirements from different user groups, and finally held preliminary discussions of future directions.

The agenda for the workshop is at:

http://www-glast.slac.stanford.edu/software/Workshops/Jan05DataHandling/agenda.htm

Day 1: Pipeline

Dan described the existing pipeline (Gino). This is implemented using a combination of perl scripts and oracle database stored procedures. The work is performed via an interface to the SLAC LSF farm. Current system supports a chain of processes for each "Task". Processes are submitted to the batch farm when their inputs are ready. The system is working as is, and with minor modifications is expected to be able to handle the pipeline requirements for the next six months.

Since the pipeline is tied closely to the concept of datasets, the main dataset catalog is currently one of the tables in the pipeline database. Several people commented that the data catalog might be better viewed as one of the outputs of the pipeline, rather than part of the pipeline. Dan sees the pros of the current system being fast development, easy interaction with the OS, and wide availability of perl libraries. The cons are igh maintainance cost (easier to rewrite than modify), lack of development tools (e.g. debugger), somewhat slow execution, and non-portability of Oracle stored procedures.

Matt demonstrated the new web frontend to the pipeline system (now at http://glast-ground.slac.stanford.edu/index.cfm?do=pipeline.welcome). This allows users to view the status of running tasks and view the log files of individual processes. In addition users can login (using Yale CAS) and if authorized can submit an XML file describing a new task.

Various groups presented their input on what they would like the pipeline to be able to do. The most often cited requirement was for a more flexible graph of processes, rather than the linear chain currently supported. Many reasons why this was needed were given, such as the need to split incoming data into parallel streams to allow it to be processed on "75 CPU's" and to then reassemble output files produced by these 75 streams. In his talk Warren Focke presented the complex graph of processes they currently need to run, and the hoops they had to jump through to get the current pipeline to support it, which included making numerous symbolic links to data files to get the pipeline to accept then as input to different tasks.

Additional requirements included

- the need to be able to prioritize tasks (so that long running MC processes would not swamp data processing, for example)
- the need to be able to either replicate the entire system elsewhere (Lyon, Perugia) or control remote jobs from the SLAC pipeline.
- the need to rerun parts of the processing, as a result of bug fixes, or improvements (e.g. recon v2, improved calibration). There was some discussion of whether trivial bug fixes would have to go throught a formal review by the CCB, and would always be treated as a new task version, or if some less formal method would be needed to put in small emergency fixes.

Warren suggested the pipeline should work more like make -k, ie you tell it what depends on what, and it figures out how to do the processing, without stopping at the first error.

In the final discussion we discussed a possible next generation of pipeline, which was assumed to consist of some combination of Java for the main server, plus perl scripts for interacting with the batch system. We discussed some of the requirements for the types of processing "rules" which would be supported by the next version of the pipeline, such as:

- Run when job A completes
- Run when all of (A,B,C,D) have completed
- Run every Sunday at 3am if at least one run has been processed in the last week
- Run if A completes and x amount of disk space is available and no batch system shutdown schedule in next 12 hours
- When available MC output files >= 100 MBytes concatenate files
- If <100 queued jobs and no high priority tasks submit more MC
- If > 50 minutes since started data processing started, and jobs not finished, call SCS to complain.

We plan to start looking at existing Java based rules engines (such as JESS http://herzberg.ca.sandia.gov/jess/), and to start designing a component architecture for the next generation system so that we can start implementing different components independently.

Day 2, Data Server

Tom Stevens described the system developed at GSSC for serving data to external users. This system works by filtering the incoming data into different files based on time and position in the sky (the latter using Heirarchical Triangular Mesh developed for Sloan Digital Sky Survey). Incoming requests are then matched against these filtered files, and only if they file lays on the border of the users request does it need to be searched event-by-event.

The events which meet the users requirements are then copied to a new file, stored on an FTP server, and the user is required to download the file within 3 days (after which it is deleted). Separate systems exist for "photons" and "full events". The system keeps a log of all incoming requests. The system will also be able to store pre-filtered data for popular requests, and users asking for the same data sample (or something close) will be directed to the pre-filtered data. Tom presented the set of formal performance requirements for the system. Tom also gave a demo of the web interface to the system.

Navid presented the data server used for DC1. This was a system put together in break-neck speed, and was only meant as a temporary stop-gap. It worled by allowing the user to specify a filter as a Root "TCut", and worked by running through the data serially and using Root macros to apply the cut.

Julie and Navid suggested some possible short-term fixes, including interfacing the system to LSF, for more parallelization, making the Root macros used more robust, and improving the web interface, by using the database to generate catalogs of available data rather than static web pages.

There was some discussion as to what type of searches internal Glast users would need, or whether for the most part they would be satisfied with pre-canned searches (or just get the whole data sample and perform cuts themselves). There was also some discussion of what type of data users would want (full merit tuples, subsets of merit tuples, full Root trees).

There was some discussion of event display requirements as part of the data server. Current plans are to use a "web accessible" version of Fred. (Wired4 seemed like it might be well suited to this, since it already works with the same HepRep interface as Fred, and is already runnable either as a web servlet (no install but very limited interactivity) or as a "web startable" Java application).

Eduardo discussed a web configurable data selection tool that they have already developed (slides are not linked in so I can't look up the details). It would be good to integrate this functionality into future versions of the data server.

As an aside Richard Mount from SCS also gave a presentation on the SCS "tera-memory" machine. This was a proposal originally submitted by SLAC to DOE as part of DOE's super-computing effort. SLAC has received 1M$ this year to built a prototype machine. The prototype will contain 1TeraByte of memory on a small number of AMD64 machines, configured to look like a very high speed file system. This machine is expected to be ready in March.

Since the prototype is smaller than requested, it is not clear how useful it will initially be to Babar. It is available to other groups, so it would be intriguing at least to consider if it could be useful as some part of the data-server setup for DC2.

Some short-term tasks:

- Improve web access to data catalog
- Look at possible improvements to DC1 data server, such as more robust root macros
- Benchmark some possible search strategies. Using an in-memory meta-data catalog using techniques similar to the existing server descibed by Tom, coupled with random access to individual events in root files (possibly in a tera-memory file system) seems like an obvious thing to try first.
- Look into possibility of setting up xrootd servers for serving data offsite
- Look at Wired for web based event display complementation to Fred.