Data Handling Requirements for Intensity Frontier Experiments

Data processing for experiments, particularly in a Grid environment, needs several kinds of data access, all of which may need be handled differently, and in a scalable fashion. These are:

- Experiment software access -- the released software of the experiment, often packaged as ups products, SoftRelTools trees or CMT build trees. This software is usually tagged with an overall version, and is shared by various types of processing (i.e. reconstruction, Monte Carlo, analysis, etc.) Mechanisms to keep copies of this software near executing jobs are needed to prevent excessive copying of code. Also recording which job uses what version of experiment software is critical to maintaining provenance of analyzed data.
- Conditions/Calibration data -- this is data relating to the position and calibration of experiment hardware over time. This is being addressed in detail in another document.
- Input data -- this is the actual event data or processed event data being used by the job. This data will need to be delivered to the job possibly from hierarchical storage from tape. Jobs should be designed to analyze the next available input data file, rather than waiting for a specific input file, so that data back-ends can deliver sets of data in the order that is most efficient to locate (i.e. grouped by tape, etc.).
- Output data -- This is the data generated by the job, which needs to be returned to some central data area, and the fact that this analysis has been completed should be logged, along with what version of Experiment software was used, etc.

Interfaces for each of these areas should be provided, which

- are usable by the experiments without significant changes to their existing code
- provide logging and monitoring so that progress of jobs can be tracked, and problems in particular data access areas above can be identified.

All such interfaces should have optimizations for on-site jobs, where some of the tasks can be accomplished via reasonably direct access, but should also have mechanisms for one-off grid environments where we have to resort to brute force.

Interfaces should be provided at various layers -- at least:

- shell script
- C/C++
- python

callable.

Potentially, a job submission layer should include information about all of these types of data access requests, so that jobs can be preferentially submitted to places where things they need (i.e. experiment software, or cached data) are already present.