

Improving SAM at CDF

Robert Illingworth and Angela Bellavance

4 June 2007

Introduction

For the past few years CDF has been using SAM as its main data handling system, replacing the earlier DFC. SAM was originally designed solely for D0, and some of the SAM concepts and methods did not map very well to the CDF computing model. Various changes were made to SAM to try and make it more compatible with CDF, but there remains a perception that SAM is working less well for CDF than it does for D0. We were asked to investigate the use of SAM at CDF, and to recommend ways in which it could be improved.

Sources of information

Information on CAF came from examining the submission scripts and CAF code, from the mailing lists, and from talking to several people (principally Krzysztof Genser, Doug Benjamin, and Ray Culbertson). We tried contacting a number of users who were running CAF jobs to ask what they were doing and what their experiences were, but we have only received one response so far. Information on the production farms came from talking to Elena Gerchtein.

There are some elements of SAM at CDF we did not investigate, either because they seem to be working well (online raw data storage, file uploads), or because we didn't have enough time to investigate them properly (SAM use at off-site locations). The areas we concentrated on were the CAF, the production farm, and the central server infrastructure.

CAF

The main complaints from those we have talked to about CAF were not SAM related at all, but with the submission tools and the batch system. The Condor batch system restart issue is the most serious as it does not interact well with SAM's file delivery methods. Condor occasionally restarts job sections while they are executing, resulting in the loss of any output that has not been copied to the destination host. In cases where the jobs are using a fixed list of files this just causes a waste of CPU resources as some work is unnecessarily repeated. When SAM is used to keep track of which files have been processed, interrupted files are not redelivered to the restarted sections, and their corresponding output files are lost. A SAM command¹ was written to try and determine the missing files; it relies on the state of the last file in a given section. The command does not work, however, when the section restarts (or crashes) between files, or after the last file is closed but before the output is returned to the user. To address this SAM was

¹ Called "sam generate recovery project"

enhanced with the ability to mark a SAM consumer process² as “completed”. A completed process is one where all the files have been read, the analysis program has terminated normally, and the output has been copied successfully to the destination host. A recovery project then consists of all files in sections that were *not* completed (plus any files that were not read at all for whatever reason). This functionality was implemented in SAM over 19 months ago, but complaints continue because the corresponding updated CAF/SAM interface required is still not deployed at CDF. Once tested and deployed, it should remove the main problem with SAM-bookkeeping-based restarts and provide a more robust recovery mechanism for other possible job failures.

Some users are running SAM jobs on CAF but not using the CAF/SAM interface to start the SAM projects.³ While there may be a genuine need for this in some cases, doing this has the potential to cause problems. For example, a large job which starts multiple SAM projects, one for each section of the job, could overload the station. Such usage should be discouraged except when absolutely necessary.

A minor issue related to CAF is the job submission script. Currently this script checks whether the requested dataset definition exists using a very old - and totally unsupported - version of the SAM client code. The script needs to be either upgraded to be compatible with the newer SAM versions, or replaced with some other method for performing the check, such as an HTTP service.

Reconstruction farm

Again, the main complaints were with the CAF submission process and with Condor, not with SAM itself. The reconstruction scripts do not use the standard SAM/CAF interface, but instead start and stop the SAM projects themselves. Job recovery is simpler than for analysis as only one file is processed per section, and the presence or absence of the output file is used to determine success. There is no need to track the “consumed” status of the file in the SAM database; whether an input file has produced a child or not is used instead. In addition, this parent-child status is not tracked in the SAM database, but instead is tracked in a local, filesystem-based database. The section output files are concatenated and the metadata for both the section output files and the final merged file are declared to SAM when the merged file is complete.⁴ Only the final merged file is stored onto tape.

Switching file tracking from using the local system to using SAM would have the advantages of using a fully transactional, robust, and well tested database. However, if the current implementation works, the benefits of rewriting it are likely to be small compared to the effort involved.

² A consumer process is the SAM entity used to deliver a stream of files to a program. A SAM project contains one or more consumer processes. In CAF terms there is a single SAM project for a job, and a separate consumer process per section.

³ For example, using `diskcache_i` to start the project, or using their own script to do so.

⁴ This contrasts with the SAMGrid processing system used by D0, where the metadata for the unmerged files are declared to SAM when they are produced, and the parentage chain through the SAM database is used to determine whether any are missing from the final merged files.

Replacing files

SAM was originally designed with a model where files are immutable, and once a given file has been declared and processed that file name always refers to the same file. An unwanted file can be marked as bad so it will no longer be given to projects, but it should not be removed entirely. This concept is enforced by database constraints. However, this policy conflicts with CDF's file naming conventions, and deletion of existing files is a common request. Deleting a file entirely is in fact possible, but doing so requires expert knowledge, results in the loss of historical information about the file, and leads to problems such as station crashes caused by files suddenly disappearing from the database. Adding the ability to "retire" a file name, making it available for reuse in SAM has been investigated and some coding work has been done, but the implementation is not yet usable. The option would be convenient since it would make replacing file names something that could be handled by a SAM shifter rather than requiring an expert. However, the assumption that a file name uniquely identifies a file is fundamental throughout much of the SAM code, so there may be a considerable amount of work required to make the retire functionality work correctly.

Servers and infrastructure

The primary SAM server infrastructure consists of an Oracle database, the SAM DB servers, and the SAM stations. SAM uses CORBA for distributed inter-process communications, and relies on a "CORBA nameservice" server which maps abstract server names to the corresponding physical host and port. The nameservice makes it possible to migrate servers to new hosts fairly easily by simply updating their nameservice entry to point at the new location. However, clients must locate the nameservice itself somehow, and this is done by distributing its location reference as text to all the clients. Moving the nameservice is therefore potentially disruptive as it requires either distributing updates to all clients everywhere or moving it to a new host with the same name (or alias) as the old location.⁵

The main SAM DB servers are set up in a load-distributing, redundant, configuration; new requests are sent to a single server⁶ which distributes each of them to one of four backend servers located on different hosts. If one of the machines hosting a backend server dies the load should be transparently redistributed to the other three servers.

The CDF CORBA nameservice and the primary DB servers run on cdfsam01, which is a 3 year old Linux server. It is on 24x7 support⁷, but since it is an older machine some thought needs to be given towards migrating the services to a newer host. This will lead to the complications from moving the nameservice mentioned above.

⁵ At D0 the CORBA nameservice runs on the Oracle database machine, which is a Sun server with a longer life cycle than the Linux servers. However, D0 has been through the disruption of moving the nameservice host at least once.

⁶ Usually referred to (somewhat inaccurately) as a "multiplexing" server.

⁷ Or so we were told, but there's no mention of this in the CD MISCOMP database.

The versions of the DB servers, stations, and client code have tended to lag behind the latest versions. While stability is a good thing, delayed updates mean that bug fixes and improvements are not available in a timely manner. In addition, the deployment of critical fixes can be complicated by the forced introduction of several backlogged changes all at once. D0, by contrast, has generally upgraded versions much more aggressively, and normally runs the latest DB server and client versions available. This policy has very rarely caused problems.

SRM

The dCache interface was grafted somewhat inelegantly onto the original SAM station design. The normal SAM station manages cache disks by executing physical file copies and deletes and keeping track of which files are on the disk. For dCache the method used was to make the SAM station manage a “virtual” cache, where it assigns file names to “disks” which map to dCache doors, but does not execute any file transfers itself. The station ultimately returns a dcap URL to the client, and it is accessing this that causes dCache to stage the file if it is not already available. An important part of this design was the need to spread the load over the doors and to throttle access in order to prevent the overloading of the system that had been an intermittent problem before.

SRM provides a uniform interface to storage resources, including dCache. An SRM based SAM station exists and has undergone limited testing. The SRM based station provides more active cache management than the dCache station because it makes SRM transfer requests itself, rather than leaving the physical cache request to come from the end user. It therefore brings benefits such as being able to pre-fetch files in advance of the clients requesting them. It should be tested more extensively at CDF with a view to eventually replacing the existing dCache SAM station. As the current configuration does work, however, we should be careful that we are not replacing it with something newer but less effective. Some of the problems that the existing system was designed to improve, such as the load balancing, would become the responsibility of SRM, and the data handing system would be left dependent on another software product with as yet unproven robustness.⁸

Conclusions and recommendations

We have not identified any issues with SAM at CDF which require major changes from our group. The opinion among those that we talked to is that SAM is working reasonably well for them.

The most significant SAM related problem is the unreliability of recovery projects on CAF, and this is already being addressed. It is possible that removing that major obstacle will reveal other problems, or increase their apparent importance. Therefore we need to monitor the system status, keeping an eye out for other issues that may require work. It is also important that we watch for and respond to user reported problems on the relevant

⁸ However, the SRM dCache interface is used by the FNAL CMS tier 1 computing centre, among others: if it works for them, it ought to work for CDF.

mailing lists. Additionally, since we were not able to get much response regarding individual user analysis we should continue to try to collect such information.

Our recommendations are, in rough order of priority:

1. Complete the testing and deployment of the new SAM/CAF interface with the “completed” process functionality. Future new features, if any, need to be deployed in a much more timely fashion – especially when the delays are caused by those who made the original request.
2. Complete the implementation of the retired file functionality.
3. Update the CAF submission to upgrade or remove the dependence on a very old SAM client.
4. Make a plan for migrating the CORBA nameservice to a new host.
5. Deploy updated SAM product versions more rapidly. The frequency and magnitude of the changes, in new releases is less than it used to be.
6. Test and ultimately deploy the SRM based station.