Software for Batch Systems

W. Akers, I. G. Bird, B. Hess, A. Kowalski

Jefferson Lab, Newport News, Virginia, USA 23606

Abstract

Over the past few years, LSF has become a standard for job management on batch farms. However, there are many instances where it cannot be deployed for a variety of reasons. In large farms the cost may be prohibitive for the set of features actually used; small university groups who wish to clone the farms and software of larger laboratories often have constraints which preclude the use of LSF.

This paper discusses a generic interface developed at Jefferson Lab to provide a set of common services to the user, while using any one of a variety of underlying batch management software products. Initially the system provides an interface to LSF and an alternative - Portable Batch System (PBS) developed by NASA and freely available in source form. It is straightforward to extend this to other systems. Such a generic interface allows users to move from one location to another and run their jobs with no modification, and by extension provides a framework for a ”global” batch system where jobs submitted at one site may be transparently executed at another. The interface also provides additional features not found in the underlying batch software. Being written in Java, the client can be easily installed anywhere and allows for authenticated remote job submission and manipulation, including a web interface.

Keywords: batch,LSF,PBS

1 Introduction and Background

At Jefferson Lab we run several clusters of batch computing systems. The largest is that used for data analysis of the physics experiments and presently consists of 150 Intel Linux CPU, with a total power of some 2650 SPECint95. This will be doubled this year. In addition, there is a small development cluster of 20 Alpha systems also running Linux. This is a development system for the lattice QCD community and will run parallel applications. At the present time there are plans to expand this cluster to some 256 CPU, and in concert with collaborators to provide a service over the wide area. On the timescale of a few years, the lab also expects to have a significant simulation facility in support of it’s Free Electron Laser (FEL) program. That facility would also be on a scale similar to that of the LQCD systems.

In all cases, these facilities have very similar user communities, partly based at the lab, but with significant numbers of collaborators wishing to have access remotely. Many of the collaborating institutes have built or will build clones of the lab systems, and wish to use the software developed in the lab to manage them.

The batch management system in use in the reconstruction farm is LSF. However, due to the pricing structure of LSF[1] it has not been feasible to use it in some of the other batch clusters at the lab, or at collaborating institutes. For farms that have a single use and few users other systems such as NQS or DQS have been used. However, since our batch farm is used by many experiments and has many users within the experiments, the scheduling facilities within LSF have been essential for managing the system. In particular we have made use of several hierarchies of scheduling within the system.

2 A high level interface

We had previously written an interface[2] to LSF that added the following abilities:

- The ability to submit and interact with jobs from any system on site. With LSF alone that ability would necessitate installing LSF on every machine, which was precluded by cost.
- The ability to submit multiple identical jobs, differing only in the files being processed. This is suitable for running large productions of reconstruction jobs for an experiment where there are frequently thousands of files to be processed. It is also useful in processing parametrized jobs, for example in large simulations investigating the various parts of a problem space.
- In our case, a transparent interface into the mass storage systems, with tape file requests being handled by the system. This ability also permits file pre-staging which will be discussed later.

In order to satisfy the additional demands discussed earlier, we have extended this interface to permit the use of other batch management systems (Figure 1). We have specifically developed an interface to PBS[3], but this can easily be extended to any other batch system that provides the basic job submission and manipulation facilities, with NQS and DQS being obvious candidates.

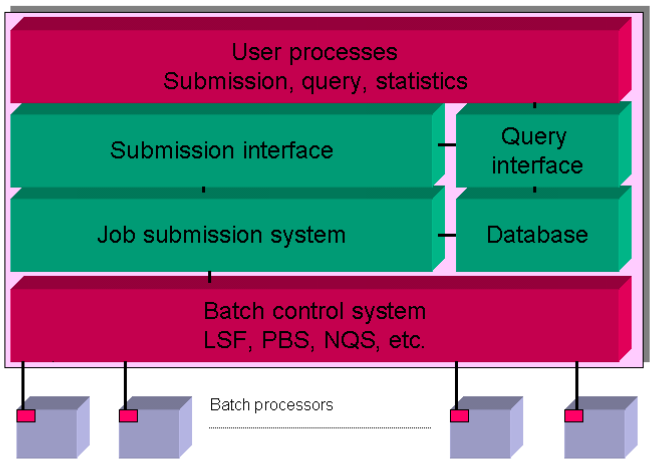

Figure 1: System interfaces

The advantage of this approach is that:

- users have a consistent view and interface, no matter which systems the job is being submitted to. The same job scripts can be used on any cluster.
- The batch system itself can be administered by the software appropriate to the cluster, whether it is LSF, or an alternative. The system used can be that appropriate to the site and is not mandated by the user needs.
- The system itself constructs the scripts that the underlying batch system uses, which means that they can be customized to site-specific needs where necessary.
- The clients are simple java programs and can be easily deployed to any local or remote client system without the need to install the sometimes complicated batch system software.

In the present system, security is minimal for remote clients. However, it is being rewritten and will include a more robust security infrastructure as an inherent part of the system. The Globus security infrastructure (GSI)[4] is the security component currently being considered for this purpose. The LQCD collaborations in particular want to eventually have a meta-cluster that will move work between clusters and run the jobs where resources permit.

2.1 Batch systems

The choice of underlying batch system is driven by the needs of the site. Many labs have standardized on LSF in the past few years, but the relative complexity of this product compared with the basic needs of most clusters is not always a good match given the cost of LSF. A good alternative system is PBS - the Portable Batch System written at NASA Ames. This is open source software and has many of the features of commercial batch systems, and is actively supported and developed by its user community. For Jefferson Lab it was a good choice as it well supports the batch submission of large parallel jobs as does LSF. The support of parallel jobs was built into PBS as a large part of the user community needed that ability. For the use at Jefferson Lab, however, a big disadvantage was that its scheduler was extremely basic compared to that built into LSF. However, the design of the system is such that one is able to plug in an arbitrary scheduler. We have written a scheduler that provides some of the complex needs that we have in a multi-purpose farm. The prototype of this scheduler, described below, has been submitted to the PBS developers and user community for comment and testing, and perhaps eventual inclusion in PBS as an alternative to the basic schedulers.

2.1.1 PBS scheduler

The scheduler for PBS that we have developed consists of 6 stages that can be configured. The combination of these results in an extremely powerful and flexible scheduling engine. The stages that have been defined are the following:

- Match-Making stage. Matches job requirements to available system resources.
- System Priority stage. Orders jobs by their system-assigned priority. This priority is normally the same for all jobs, except for those jobs that are part of a larger task, for example a job that has been held pending a file staging may get a higher priority once the file is available.
- Queue selection stage. Selects the next queue that should have work run.
- User Priority stage. User assigned priority can be used by the submitter to order his own work.
- User Share stage. Provides a weight to determine which user should be serviced next. User priorities can depend on group and individual time allocations, historical use, etc.
- Job Age. Considers how long the job has been waiting in the queue.

These stages may be used in any order, allowing additional flexibility depending on requirements. The prototype of this scheduler has been released to the PBS community for comment and testing.

2.2 File pre-staging

File pre-staging, that is, extracting a file from the tape mass store and having it available on disk before a job begins execution, has become more and more crucial. At Jefferson Lab our STK silo is equipped with only 8 RedWood drives (although it will have a larger number of 9840 drives soon), which has meant that as the batch farm has grown we have frequently seen a situation where a batch node is idle while its job is waiting for data. This results in an inefficiency in the farm usage.

To counteract this problem we are implementing pre-staging. This is straightforward to implement in the interface layer. As the batch scripts are constructed, a pre-cursor job is built to extract the data files, (as many as possible with a single tape mount), and until this extraction is successful the remaining jobs are held in the queue. Once the data is available, the jobs are released and queued with a very high priority, so that they will begin execution as soon as there is an available job slot.

The advantage of doing this in the interface layer is that it provides the needed feedback between the batch and mass storage systems when needed without further complicating or connecting those systems unnecessarily.

References

1 Load Sharing Facility, Platform Computing Corp. `http://www.platform.com`.

2 I.G. Bird et al , "Database Driven Scheduling for Batch Systems", CHEP'97, Berlin, Spring 1997.

3 Portable Batch System, `http://pbs.mrj.com`.

4 The Globus Project, `http://www.globus.org`