Thoughts on Requirements for ad-hoc VOs

Frank Würthwein

January 2005

Abstract:
We present thoughts on how ad-hoc VOs would be useful in HEP in order to implement sharing of compute and storage resources on the grid in a way users are used to for centralized resources in HEP today.

The Data Sharing Use Case

An individual user produces data on the grid. She stores the data. She records/registers the metadata. She wants to share both with her friend(s) in private prior to presenting her results to a larger audience in her experiment. She needs to provide read access to both data and metadata to a limited set of people for a limited set of time. Current practice is to put the data on some shared disk, and send the metadata by email. This is clearly not a viable way of operating in a grid environment.

The Resource Sharing Use Case

A set of individuals have rights to resources that they want to pool to advance a common goal. This may in general include disk space as well as CPU. They want to do so such that one of them can run a workload that they will all benefit from while all get "charged for the activity".

Current practice in CDF is for users to add each others krb5 principal into each others account on fileservers with user quotas. This is problematic even today as it is no longer possible to trace who put which file onto the fileserver. The alternative method of creating groups, and making disk space group writeable is not viable because it requires a system administrator to add people to the groups on the servers, thus making ad-hoc groups impossible, and adding a management nightmare.

Sharing of CPUs on an ad-hoc basis happens in one of two ways today in CDF. First we provide groups priority to a resource quota, and an interface such that they can manage access to those resources themselves. The creation of the group requires manual admin intervention, and is thus not really an option for truly ad-hoc groups. Second, one person creates an executable, and all people in the group submit it, each on a different subset of the data to analyze. That way they get N times the amount of CPU with N being the number of people in the small group. This works well as long as N << total number of active people on the cluster. As the latter is ~100 at any one day, and most ad hoc groups are ~5 or so this works nicely pretty much always.