Contribution ID: **374** Type: **Poster**

A FairShare Scheduling Service for OpenNebula

Tuesday, 11 October 2016 16:30 (15 minutes)

In the ideal limit of infinite resources, multi-tenant applications are able to scale in/out on a Cloud driven only by their functional requirements. A large Public Cloud may be a reasonable approximation of this condition, where tenants are normally charged *a posteriori* for their resource consumption. On the other hand, small scientific computing centres usually work in a saturated regime and tenants are charged *a priori* for their computing needs by paying for a fraction of the computing/storage resources constituting the Cloud infrastructure. Within this context, an advanced resource allocation policy is needed in order to optimise the use of the data center. We consider a scenario in which a configurable fraction of the available resources is statically assigned and partitioned among projects according to fixed shares. Additional assets are partitioned dynamically following the effective requests per project; efficient and fair access to such resources must be granted to all projects.

The general topic of advanced resource scheduling is addressed by several components of the EU-funded INDIGO-DataCloud project. In this context, dedicated services for the OpenNebula and OpenStack cloud management systems are addressed separately, because of the different internal architectures of the systems. In this contribution, we describe the FairShare Scheduler Service (FSS) for OpenNebula (ON). The service satisfies resource requests according to an algorithm which prioritises tasks according to an initial weight and to the historical resource usage of the project, irrespective of the number of tasks she has running on the system. The software was designed to be less intrusive as possible in the ON code. By keeping minimal dependencies on the ON implementation details, we expect our code to be fairly independent on future ON internals changes and developments.

The scheduling service is structured as a self-contained module interacting only with the ON XML-RPC interface. Its core component is the Priority Manager (PM), whose main task is to calculate a set of priorities for queued jobs. The manager interacts with a set of pluggable algorithms to calculate priorities. The PM exposes an XML-RPC interface, independent from the ON core one, and uses an independent Priority Database as data back-end. The second fundamental building block of the FSS module is the scheduler itself. The default ON scheduler is responsible for the matching of pending requests to the most suited physical resources. The queue of pending jobs is retrieved through an XML-RPC call to the ON core and they are served in a first-in-first-out manner. We keep the original scheduler implementation, but the queue of pending jobs to be processed is the one ordered according to priorities as delivered by the PM.

After a description of the module's architecture, internal data representation and APIs, we show the results of the tests performed on the first prototype.

Tertiary Keyword (Optional)

Secondary Keyword (Optional)

Virtualization

Primary Keyword (Mandatory)

Cloud technologies

Primary authors: Dr VALLERO, Sara (INFN Torino); Dr BAGNASCO, Stefano (INFN Torino)

Presenter: Dr VALLERO, Sara (INFN Torino)

Session Classification: Posters A / Break

Track Classification: Track 7: Middleware, Monitoring and Accounting