Opportunistic Resource Usage in CMS

Opportunistic Resource Usage in CMS

Peter Kreuzer[1], Dirk Hufnagel[2], D.Dykstra[2], O.Gutsche[2], M.Tadel[3], I.Sfiligoi[3], J.Letts[3], F.Wuerthwein[3], A.McCrea[3], B.Bockelman[4], E.Fajardo[5], L.Linares[5], R.Wagner[6], P.Konstantinov[7], B.Blumenfeld[8], D.Bradley[9]

On behalf of the CMS Collaboration

[1]Rheinisch-Westfalische Technische Hochschule (RWTH)
[2]Fermi National Accelerator Laboratory (Fermilab)
[3]University of California, San Diego (UCSD)
[4]University of Nebraska-Lincoln (UNL)
[5]Universidad de los Andes (UNIANDES)
[6]San Diego Supercomputer Center, La Jolia, California/USA (SDSC)
[7]Institute for Nuclear Research and Nuclear Energy (Sofia-INRNE)
[8]John Hopkins University
[9]University of Wisconsin-Madison

Peter.Kreuzer@cern.ch

Abstract. CMS is using a tiered setup of dedicated computing resources provided by sites distributed over the world and organized in WLCG. These sites pledge resources to CMS and are preparing them especially for CMS to run the experiment's applications. But there are more resources available opportunistically both on the GRID and in local university and research clusters which can be used for CMS applications. We will present CMS' strategy to use opportunistic resources and prepare them dynamically to run CMS applications. CMS is able to run its applications on resources that can be reached through the GRID, through EC2 compliant cloud interfaces. Even resources that can be used through ssh login nodes can be harnessed. All of these usage modes are integrated transparently into the GlideIn WMS submission infrastructure, which is the basis of CMS' opportunistic resource usage strategy. Technologies like Parrot to mount the software distribution via CVMFS and xrootd for access to data and simulation samples via the WAN are used and will be described. We will summarize the experience with opportunistic resource usage and give an outlook for the restart of LHC data taking in 2015.

1. Introduction

The computing resource planning model of the Compact Muon Solenoid (CMS) experiment is based on a yearly resource request in terms of CPU, Disk and Tape needs, at the ~60 CMS computer centers located worldwide. This request relies on data models and processing strategies that are represented in well established spreadsheets and based on the CMS Computing Model. The request is reviewed bi-yearly by the Resource Review Board of the World Wide LHC Grid (WLCG) organization. The yearly CMS resource utilization is reviewed in the same process.

Since 2009 the CMS planning fits the constraints of a flat budget (imposed by the level of funding), with fluctuations mainly driven by the Large Hadron Collider (LHC) program. While the data storage utilization matches with the planning pretty well, the processing (CPU) utilization regularly goes beyond. So far extra CPUs were typically opportunistic resources that are already linked to CMS (non-pledged Tier-1/2/3 resources).

In this paper we are covering a more general concept of opportunistic resources that can be academic or commercial and that can be very useful in case of deviations from a flat budget.

2. Motivations and Challenges of Opportunistic Resources

The LHC physics program invests more money into people than computing hardware hence we need to optimize the productivity of human assets. Our human assets are distributed worldwide with access to shared resources. One solution to take advantage of such large human potential is to take advantage of the evolution of the CMS Computing Model and of the technology, by encouraging an open architecture that allows dynamic integration of shared resources, i.e. that are working around missing services at remote computer centers.

Moreover, one of the standard CMS workflows is the organized data processing, which is an issue of short-term peak computing demands (typically prior to large physics conferences). It makes little sense in that case to solve this issue via "steady state hardware capacity" only.

The short/mid-term future plan by the CMS collaboration regarding opportunistic resources is that they cover our resource needs at the 10% level average over a whole year, while they may cover a much larger fraction of the overall resource needs during short time periods. The goal is to integrate the opportunistic use cases as transparently as possible into the CMS processing machinery, namely to embed the needed setup into the CMS Workflow Management System (GlideIn WMS [1]).

In the next section we cover the main software challenges related to opportunistic resource usage, that can be categorized into either runtime and submission challenges. In addition, such solution also creates potential networking and operational challenges.

3. Software Challenges

The main question to answer is: how to get a CMS job into an opportunistic resource?

To answer this question, we distinguish between 3 types of use cases:

- Non-CMS WLCG[1] resources
- Non-WLCG resources
- Cloud (EC2 [2]) type of resource

In what follows, we address the main challenges in each of the above use case.

3.1. Use case I: Non-CMS WLCG resources

CMS has been working with such use cases, in particular the FermiGrid [3] resources, expanded more generally to OSG [4] sites. The main challenge in that case is to solve the runtime problem, namely to get the environment needed by the CMS application emulated on the local resource.

This is solved via the Parrot-Wrapper [5], which is a script around the CMS payload in the GlideIn WMS pilot. It simulates the CMS software environment, in particular the SW application framework (CMSSW) and the Workload Management python runtime code. It uses the Virtual File System Parrot [6] to make remote IO accessible from local jobs, without root privileges, effectively allowing to mount CVMFS[2]. The wrapper also detects site proxies to put them into the CMS configuration, for example the Frontier [7] proxy to integrate the detector conditions data into the processing. More generally it provides the CMS specific site configuration files and the grid UI in the local file system for remote data access via "xrootd" and stage-out via "lcg-cp" to a CMS site [8].

3.2. Use case II: Non-WLCG resources

[1] We call "WLCG resource" a site accessible through a GRID gateway (e.g. a Computing Element)

[2] CVMFS: CERN VM File System

CMS has been running production workflows with such cases, for example at the San Diego Supercomputer Center, La Jolia, California/USA (SDSC). In that case the runtime challenge is the same as for use case I. Given the absence of grid-interface to the site, the additional challenge in that case is the job submission.

The solution in that case is to use the BOSCO [9] job submission manager and tunnel to the site via "ssh". Such process is integrated into the CMS GlideIn WMS machinery and was successfully used to perform a large scale CMS re-Reconstruction workflow at SDSC during Spring 2013. Even if this was not a fully opportunistic use case (since part of the CMS site configuration was locally deployed), it is considered as a proof-of-principle for large scale opportunistic processing by CMS.

3.3. Use case III: Cloud (EC2) type of resources

In this case the aim is to run CMS jobs on a Cloud resource. CMS is currently exploring several such options.

Again the runtime challenge is identical to use cases I and II. In terms of jobs submission, the solution in that case is to make use of OpenStack/EC2 [10] resources integrated into the CMS GlideIn WMS software. CMS has been commissioning such solution and running production workflows in 2013 [11]. The promising aspect if this use case is that it opens a large spectrum of commercial Cloud solutions for the future.

4. Scalability, Networking and Operational Challenges

The 3 use cases discussed above have common scalability challenges, in terms of proxy's and more generally in terms of networking capacity, depending on the size and the number of opportunistic resources that are used for a given CMS workflow.

In addition, the operational challenges should not be neglected, the main goal being to integrate an opportunistic resource in terms of job management and monitoring as transparently as possible into the centrally organized processing. We will discuss some of these challenges in the following sections.

4.1. Proxy Setup

The challenge is to ensure that the Frontier and CMVFS requests are cached appropriately such that they scale with the number of requested jobs.

The design foresees that proxies are auto-detected at the site if available, with a fallback solution to the site configuration provided through the Parrot-Wrapper described in Sect.3.1. In addition, CMS is in the process of setting up global fallback proxies at its major computing facilities (CERN and Fermilab), which have been already partially deployed and tested. Finally, for large opportunistic resources, dedicated efforts for special proxy setup may be spent, by using a local or a "close" CMS-owned proxy server.

4.2. Stage-out challenge

As already described in Sect.3.1, the site dependency regarding stage out is removed by using an EMI-2 CVMFS grid UI[3] and then stage out to a CMS site via "lcg-cp". The site and stage-out settings are taken from the site configuration provided through the Parrot-Wrapper. As the size of opportunistic sites is scaling up, one could run into networking issues, which need to be addressed for example through stage out to multiple CMS sites. Another complexity may come from the site policies, for example related to restrictions in outbound connectivity.

4.3. Operational challenges

The goal is to render the opportunistic use cases as transparent as possible for the CMS central computing operations.

[3] EMI-2 CVMFS grid UI repository (grid.cern.ch)

This is achieved by creating a "fake CMS site" that maps to the opportunistic resource and configures the fake site in all the relevant CMS information systems so that it can be transparently used by the standard production tools. There are two options in the way a physical site may be mapped to a "fake site":

- A meta site with a shared wrapper configuration among several physical resources, that means one single site configuration
- A dedicated opportunistic site, with an optimized custom site configuration

The choice between these two options depends on a balance between optimizing a specific resource and keeping the number of sites manageable in terms of operations overhead. The SDSC example described in Sect.3.2 is of the latter type, where some effort was dedicated to optimize the site configuration to the CMS use case, given the substantial short-term processing gain it represented.

Another operational challenge is to monitor and trouble-shoot the production workflows at opportunistic sites, in particular in case of Meta sites, since the physical site is not "visible" in that case. This is the price to pay for the transparent use of opportunistic resources, and future operational experience will show where the optimal working point is for various use cases.

5. Conclusions

Given the evolution of the CMS computing model to a more dynamic usage of the worldwide distributed resources, it is natural to consider the opportunistic resources, in particular to optimize the productivity of human assets and to cover short-term peak computing demands.

Depending on the type of opportunistic resource, the software challenges to use such resources are at runtime or at submission time, or both. CMS has already tested at large scale several such use cases, with successful and encouraging results. The scalability in terms of networking and the optimization of the operational overhead using such resources needs to be determined in the upcoming years.

References

[1] GlideIn WMS: Workflow Management System used by CMS: http://www.uscms.org/SoftwareComputing/Grid/WMS/glideinWMS/doc.prd/index.html
[2] Amazon Elastic Compute Cloud: http://aws.amazon.com/ec2/
[3] FermiGrid: http://fermigrid.fnal.gov/
[4] OSG: Open Science Grid: http://www.opensciencegrid.org/
[5] Parrot-Wrapper: https://github.com/dcbradley/parrot_glidein_wrapper
[6] Parrot: http://www3.nd.edu/~ccl/software/parrot/
[7] Frontier distributed database caching system: http://frontier.cern.ch
[8] K.Bloom et al., CMS Use of a Data Federation, these proceedings.
[9] D.J. Weitzel et al., Accessing Opportunistic Resources with BOSCO, these proceedings.
[10] EC2: Amazon Elastic Cloud: http://aws.amazon.com/ec2/
[11] D. Colling et al., Usage of the CMS High Level Trigger Farm as Cloud Resource, these proceedings.