DISTRIBUTED CMS ANALYSIS ON THE OPEN SCIENCE GRID

O. Gutsche*, Fermi National Accelerator Laboratory, Batavia, IL 60510, USA
K. Bockjoo†, University of Florida, Gainesville, FL 32611, USA

Abstract

The CMS computing model provides reconstruction and access to recorded data of the CMS detector as well as to Monte Carlo (MC) generated data. Due to the increased complexity, these functionalities will be provided by a tier structure of globally located computing centers using GRID technologies. In the CMS baseline, user access to data is provided by the CMS Remote Analysis Builder (CRAB) analysis tool which enables the user to execute analysis applications on locally resident data using GRID tools independent of the geographical location. Currently, mostly two different toolkits provide the needed functionalities, the Worldwide LHC Computing Grid (LCG) and the Open Science Grid (OSG). Due to infrastructure and service differences between the two toolkits, analysis tools developed for one are frequently not immediately compatible with the other. In this paper, we will describe the development of additions to the CRAB tool to run user analysis on OSG sites. We will discuss the approach of using the GRID submission of the Condor batch system (Condor-G) to provide a sandbox functionality for the user's analysis job. For LCG sites, this is provided amongst other things by the resource broker. We will discuss the differences of user analysis on LCG and OSG sites and present first experiences running CMS user jobs at OSG sites.

THE CMS COMPUTING MODEL

The Compact Muon Solenoid (CMS) [1] is one of the four experiments at the Large Hadron Collider (LHC) [2]. From the beginning of data taking on, the CMS detector will record large amounts of events which have to be available to all CMS collaborants for analysis.

After the trigger process and the reconstruction (see Fig. 1), the event information is split in $\sim$ 50 primary datasets according to trigger criteria.

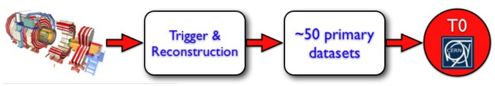

Figure 1: Data taking and reconstruction flow from detector to $T0$ at CERN.

The cumulated size of the recorded event data including reconstructed information plus the needed Monte Carlo (MC) data samples for analysis is expected to be in the PetaByte range per year and poses challenging demands on the CMS computing model [3].

For technical and administrative reasons, the CMS computing model is build upon a tier structure of globally located computing centers (see Fig. 2).

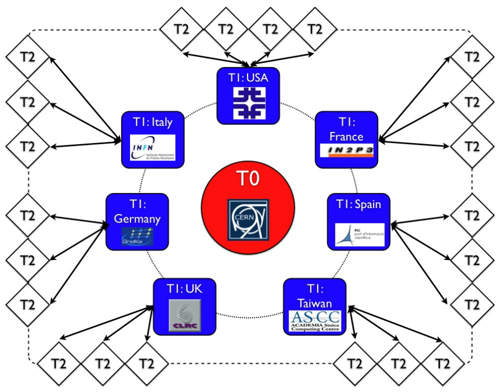

Figure 2: Tier structure of the CMS computing model.

The hosting laboratory of the LHC and its four experiments, CERN, represents the lowest tier ($T0$) and archives the recorded data. The available computing infrastructure at CERN represents $\sim 20\%$ of the complete computing requirements of CMS. The remaining $\sim 80\%$ are located at the subsequent tiers.

The ~ 50 primary datasets of raw and reconstructed data are distributed amongst the 7 regional tier centers ($T1$) on the next tier level. The $T1$ centers will provide a second archive of the subsets of the primary datasets and infrastructure for detailed calibration and re-reconstruction, analysis and skimming. In addition, all MC samples will be archived at this level of the tier structure.

The bulk of the user analysis is performed on the $T2$ level by 25 $T2$ centers which are geographically grouped and associated to one of the 7 $T1$ centers. In addition, MC simulation is performed and the simulated samples are copied back to the $T1$ level for archiving.

Due to the globally distributed computing infrastructure and the data distribution by physics content, all user have to be able to access all datasets regardless of their location. To accomplish this requirement, the user access is realized using GRID technologies where the user interacts with the GRID infrastructure via a common analysis submission tool: CMS Remote Analysis Builder (CRAB) [4].

* gutsche@fnal.gov
† bockjoo@ufl.edu

MIDDLEWARE WITHIN THE CMS GRID

The CMS tier structure uses two main middleware architectures. The LHC Computing Grid (LCG) [5] / Enabling Grid for E-sciencE (EGEE) [6] middleware is primarily deployed at European computing centers whereas the Open Science Grid (OSG) [7] middleware is used at American centers. Both architectures use compatible underlying components but differ in their general approach.

The LCG/EGEE middleware follows a more LHC experiment specific approach using more higher level tools deployed to fulfill the needs of the participating GRID users. Several LHC specific components are available within the LCG/EGEE middleware. Amongst those, the Resource Broker (RB) plays a prominent role as it is the central place for job submission and user interaction. The RB provides load balancing between the LCG/EGEE GRID infrastructure and contains a sandbox for user file input and output to and from the remote analysis application.

The OSG middleware uses more lower level tools to provide access to OSG resources. The access is handled by the underlying Globus [8] toolkit and the experiments have to install own tools to provide the required functionalities for analysis and MC production. The OSG middleware does currently not foresee a RB to centrally handle and balance job submission. Therefore, also the sandbox functionality known from LCG/EGEE is missing.

The contribution of OSG based resources to the CMS GRID infrastructure is sizeable. Summarizing the US centers, the $T1$ center at the Fermi National Accelerator Laboratory (FNAL) has 7 associated $T2$ centers which represent a significant percentage of the overall CMS computing needs (see Tab. 1).

Site	Processors	Disk [TB]
Caltech	153	44
Florida	240+	73
MIT	coming soon	coming soon
Nebraska	256	19
Purdue	228	~25
San Diego	228	44.5
Wisconsin	400	50

Table 1: OSG $T2$ contribution to the CMS GRID infrastructure.

CMS REMOTE BATCH BUILDER (CRAB)

The CMS Remote Analysis Builder (CRAB) provides the user with an interface to use the CMS GRID infrastructure. It provides a framework to submit user analysis jobs to the CMS $T1$ and $T2$ centers. The interaction of the user with the GRID is limited to the usage of CRAB. The user does not need any specific knowledge about GRID tools and their usage.

CRAB takes care of the generic steps of user analysis:

User code: The CMS analysis execution on GRID resources is based on pre-deployed software releases which are installed at the resources. The user's additions and changes are packed by CRAB, shipped to the resource and executed.

Execution and status information: CRAB prepares the execution of user code on the GRID resource and collects all necessary information like the local location of the data or MC sample. In addition, it provides an interface for the user to inquire job status information.

Output handling: CRAB handles output retrieval for the user. It either can collect the output after the job execution has finished or it can save the output to a specified storage element (SE).

The interaction of the user with CRAB is split into 4 steps:

1. Job creation:
   - Resolve requested dataset/MC sample using a global CMS dataset bookkeeping service.
   - Collect list of sites which provide access to the requested dataset/MC sample using a global CMS dataset location service.
   - Create job submission scripts (JDL) and job execution scripts adding site specific information.
2. Job submission
3. Status check
4. Output retrieval

The current CRAB implementation uses the LCG/EGEE resource broker to handle the job submission including load balancing, shipping of the user code to the resource and the output retrieval. The sandbox of the RB plays a significant role for this. As the OSG middleware does currently not provide a RB or RB functionalities, functionality has to be added to CRAB to also be able to use OSG resources. In the following, a first approach is described and further plans are summarized.

OSG ADDITIONS TO CRAB

The OSG middleware is based upon the Virtual Data Toolkit (VDT) [9] which itself contains the Globus [8] toolkit and Condor [10], a workload management system for large collections of distributively owned computing resources. Apart from the functionalities of a full-featured batch system, Condor provides functionalities for interaction with GRID resources. This Condor-G mode provides:

- GRID submission using the Globus toolkit,
- access to Globus based GRID resources independent of the locally used batch system,

- a sandbox for insertion and retrieval of job input and output.

The Condor-G mode is chosen for the first implementation of OSG submission functionalities in CRAB. There are no requirements for OSG sites to be able to support access for CRAB analysis jobs via Condor-G. Only the user system has to provide a local Condor installation with activated Condor-G mode and a running instance of the Condor scheduler on the user machine.

This first implementation is a direct submission to a single OSG resource without load balancing between resources which all provide access to the required dataset/MC sample. A dedicated OSG mode has been added to CRAB steered by a command line parameter as an exclusive mode. CRAB can be used with LCG/EGEE via the RB or with OSG resources where the analysis jobs are submitted to only one OSG resource.

The OSG mode has to change all LCG/EGEE steps in the CRAB user interaction. Apart from the replacement of LCG/EGEE middleware commands with Condor commands, the following functions had to be adapted in OSG mode:

Job creation:

- OSG sites and their published datasets/MC samples are already available from the global CMS dataset location service but are ignored by the LCG/EGEE RB used for job submission. In the list of sites which provide the requested dataset/MC sample, non OSG sites have to excluded in OSG mode. The very first implementation contained the site names of the 7 OSG $T2$ centers hardcoded in CRAB. This approach is now replaced using the GridCat [11] service of the Open Science Grid. To choose a site for submission, the first site of the list of all resources is chosen.
- The created job submission script format has to be changed from the LCG/EGEE syntax to the OSG syntax. For the OSG syntax, the CE host name and the batch system type of the chosen OSG resource has to be defined. The very first implementation contained these information hardcoded but is also replaced by the GridCat service.
- The job execution script has to be changed to accommodate the OSG specific running environment. The most significant change is the location of the CMS software releases at the site. The LCG/EGEE approach of providing an environment variable for the location has to be replaced by an OSG specific solution. The very first implementation used hardcoded information which is replaced now by the CMSSoftDB [12] service.

Job Submission: The job submission uses Condor commands with the locally running Condor scheduler instead of LCG/EGEE commands.

Status check: The status check uses Condor commands and is required to be executed on the same machine where also the submission was performed. The queue of the locally running Condor scheduler has to be the same as for the submission.

Output handling: A dedicated output handling after the job execution has finished is not necessary. The output is automatically retrieved in Condor-G mode.

Compared to the LCG/EGEE CRAB mode, the OSG CRAB mode is approx. an order of magnitude faster due to the direct submission to a single dedicated site. But the OSG mode does not provide load balancing between several sites. Also the load on the site's head node is increased during submission in OSG mode.

SERVICE CHALLENGE 3

In 2005, CMS exercised a Service Challenge (SC3) to test the computing infrastructure. The goal of SC3 from 11/17/2005 to 12/09/2005 was to exercise a realistic startup scenario for CMS consisting of the transfer of MC samples instead of datasets from the $T0$ to the tier structure and their publication in the global services. To verify the transfered samples, analysis jobs have been run on all the samples on all sites using CRAB. The first OSG implementation in CRAB was used to enable the participation of 6 OSG $T2$ centers.

Over 13,000 analysis jobs have been submitted against the transfered MC samples to OSG sites (see Tab. 2). Each job processed 1000 events and finished well below 8 hours.

Of the 13,000 analysis jobs, 48% finished successfully with exit code zero, 36% finished with an non-zero exit code which can be attributed to executable failures and 16% of the jobs were aborted by the Grid. Breaking up the 16% aborted jobs into the individual GRID/Globus errors (see Tab. 3), the dominant contribution of 71% of the 16% aborted jobs is originated from output retrieval problems which can be attributed mostly to application failures not producing output. In the end, only $\sim 5\%$ of the 13,000 jobs have been aborted due to GRID failures.

The above presented analysis of jobs executed on SC3 MC samples on OSG sites during SC3 shows the successful and stable usage of the OSG addition to CRAB with a failure rate of 5% due to GRID failures.

PLANS FOR THE OSG ADDITION TO CRAB

The OSG additions to CRAB will be finalized in the next major CRAB release. The usage of hardcoded parameters for the OSG sites will be avoided and the GridCat service of the Open Science Grid will be used consistently.

Two submission possibilities will be implemented:

Site Name	Caltech	Florida	Nebraska	Purdue	San Diego	Wisconsin	ALL OSG T2 Sites
All jobs with zero status	8	39	2626	360	733	2496	6262
All jobs with non-zero status	374	61	1172	148	2128	834	4717
All Aborted jobs	409	90	183	629	574	194	2079
All Completed jobs	791	190	3981	1137	3435	3524	13058

Table 2: Jobs run against MC samples at OSG sites during SC3. *zero status*: job finished with success, *non-zero status*: specific executable failure exit code, *aborted*: problems in GRID submission or output retrieval.

Globus error	Description	[%]
7	an authorization operation failed	12.67
10	data transfer to the server failed	0.96
17	the job failed when the job manager attempted to run it	0.43
22	the job manager failed to create an internal script argument file	4.38
24	the job manager detected an invalid script response	0.05
30	the job manager failed to open the user proxy	1.78
43	the job manager failed to stage the executable	2.79
48	the provided RSL could not be properly parsed	0.10
73	the job manager failed to open stdout	0.19
74	the job manager failed to open stderr	0.19
79	connecting to the job manager failed.	0.24
93	the gatekeeper failed to find the requested service	0.10
111	the job manager timed out while waiting for a commit signal	0.14
121	the job state file doesn't exist	0.05
129	the standard output/error size is different	0.10
131	the user proxy expired (job is still running)	1.01
135	the job manager could not stage in a file	3.76
136	the scratch directory could not be created	0.24
156	the job contact string does not match any which the job manager is handling	0.05
158	the job manager could not lock the state lock file	0.05
155	the job manager could not stage out a file	70.71

Table 3: Error codes and of aborted jobs on OSG sites during SC3.

Condor-G: The Condor-G submission to a dedicated OSG site will be implemented to use the submission speed advantage under special conditions.

Resource Broker: OSG sites will be enabled to report their status to the LCG/EGEE RB to enable also submission to OSG sites via the RB.

To support both planned submission possibilities, the job execution script will be instrumented to discover the middleware flavor when the job is started at the site.

CONCLUSIONS

The first implementation of OSG functionalities to CRAB and its first usage during the Service Challenge 3 have been described. The OSG $T2$ sites have participated successfully in SC3 demonstrated by 13,000 analysis jobs executed against the transfered MC samples on OSG $T2$ sites. In the end, only 5% of the jobs have been aborted due to GRID failures.

The OSG additions to CRAB will be finalized in the next major CRAB release providing submission via Condor-G and the resource broker.

ACKNOWLEDGMENTS

We would like to thank the CRAB developers and OSG site administrators for their help and support.

REFERENCES

[1] "CMS, the Compact Muon Solenoid: Technical proposal," CERN-LHCC-94-38

[2] O. Bruning, P. Collier, P. Lebrun, S. Myers, R. Ostojic, J. Poole and P. Proudlock, "LHC design report. Vol. I: The LHC main ring," CERN-2004-003

[3] CMS Collaboration, "CMS: The computing project. Technical design report," CERN-LHCC-2005-023

[4] S. Lacaprara et al, "CRAB: a tool to enable CMS Distributed Analysis", *Prepared for Computing in High-Energy Physics (CHEP '06), Mumbai, India, 13 Feb - 17 Feb 2006*

[5] I. Bird *et al.*, "LHC computing Grid. Technical design report," CERN-LHCC-2005-024

[6] The EGEE Project, "EGEE Middleware Architecture and Planing (Release 1)", 2004. Available at edms.cern.ch

[7] R. Pordes, "The Open Science Grid," *Prepared for Computing in High-Energy Physics (CHEP '04), Interlaken, Switzerland, 27 Sep - 1 Oct 2004*

[8] I. Foster, C. Kesselman, "Globus: A Metacomputing Infrastructure Toolkit", Intl J. Supercomputer Applications, 11(2):115-128, 1997

[9] Virtual Data Toolkit (VDT): http://vdt.cs.wisc.edu/index.html

[10] Douglas Thain, Todd Tannenbaum, and Miron Livny, "Distributed Computing in Practice: The Condor Experience", Concurrency and Computation: Practice and Experience, Vol. 17, No. 2-4, pages 323-356, February-April, 2005

[11] GridCat - The Catalog of Sites and their Status: http://osg-cat.grid.iu.edu/

[12] CMSSoftDB: http://lynx.fnal.gov/runjob/CMSSoftDB