CRAB: a tool to enable CMS Distributed Analysis

F. Fanzago, INFN Padova, Italy, S. Lacaprara, INFN Legnaro, Italy
D. Spiga, INFN Perugia, Italy
M. Corvo, CERN/INFN Geneva, Switzerland
A. Fanfani, INFN Bologna, Italy
N. Defilippis, INFN Bari, Italy
S.Argiró, CERN/INFN Geneva, Switzerland
G. Ciraolo, INFN Pisa, Italy
N. Smirnov, INFN Padova, Italy

Abstract

CRAB (Cms Remote Analysis Builder) is a tool, developed by INFN within the CMS collaboration, which provides to physicists the possibility to analyze large amount of data exploiting the huge computing power of grid distributed systems. It's currently used to analyze simulated data needed to prepare the Physics Technical Design Report. Data produced by CMS are distributed among several Computing Centers, and CRAB allows a generic users, without specific knowledge of grid infrastructure, to access and analyze those remote data, hiding the complexity of distributed computational services and making job submission and management as simple as in a local environment. The experience gained during the current CMS distributed data analysis effort is reported, along with CRAB ongoing developments. The interaction of CRAB with the current and future CMS Data Management services is described, as well as the usage of WLCG/gLite/OSG middleware to provide access to different grid environments. Finally, the use within CRAB of BOSS for logging, bookkeeping and monitoring is presented.

INTRODUCTION

CMS (Compact Muon Solenoid) is one of the four particle physics experiments that will collect data at LHC (Large Hadron Collider) starting in 2007 at CERN, Switzerland.
This experiment will produce a large amount of data (few PBytes each year) that will be stored in many computing centres (Tiers) in the countries partecipating to the CMS collaboration and made available for analysis to world-wide distributed physicists.
CMS will use a distributed architecture based on grid infrastructure to ensure remote resources availability and to assure remote data access to authorized user (belonging to CMS Virtual Organization).
Tools for accessing distributed data and resources, provided by the Worldwide LHC Computing Grid (WLCG)[1] and the American Open Science Grid (OSG)[2], are under evaluation.

DISTRIBUTED ANALYSIS

Data analysis in a distributed environment is a complex computing task that CMS decided to accomplish exploiting a computing infrastructure based on many geographically distributed Tiers (see Fig. 1).

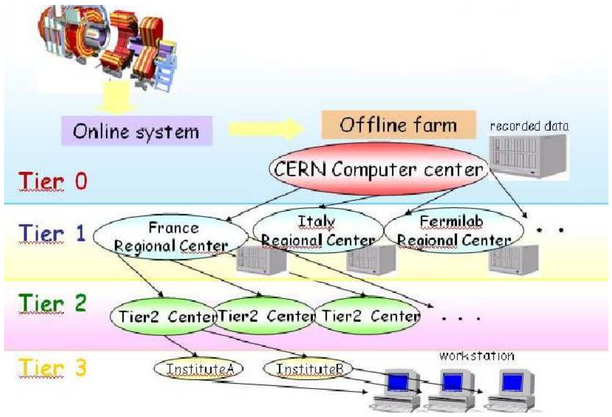

Figure 1: CMS Computing infrastructure organized in geographically distributed Tiers.

This distributed analysis assumes to know which data are available, where they are stored and how to access them. It also needs to understand which resources are able to comply with user analysis job requirements. The CMS collaboration is developing some tools, interfaced with available grid services, to achive these purposes.
They include:

- installation of CMS software via grid on remote resources[3]
- "data transfer service" to move and manage a large flow of data among Tiers[4]
- "data validation system" to ensure data consistency and readiness[5]
- "data location system" to keep track of data available in each site and to allow data discovery. Now it is composed by a central database located at CERN (RefDB) containing information about what kind of

data have been produced; a local database (PubDB) in each Tier with information about where data are stored (which local Storage Element - SE) and the access protocol; local POOL catalogs with physical data location on SE

- A friendly interface to simplify the creation and the submission of analysis jobs to grid environments: **CRAB** (**C**MS **R**emote **A**nalysis **B**uilder). Its purpose is to allow users with no knowledge of grid infrastructure to run their analysis code on data available at remote sites as easily as in a local environment hiding grid infrastructure details

- job monitoring and logging-bookkeeping system to keep trace of submitted jobs and to identify grid middleware or user application problems

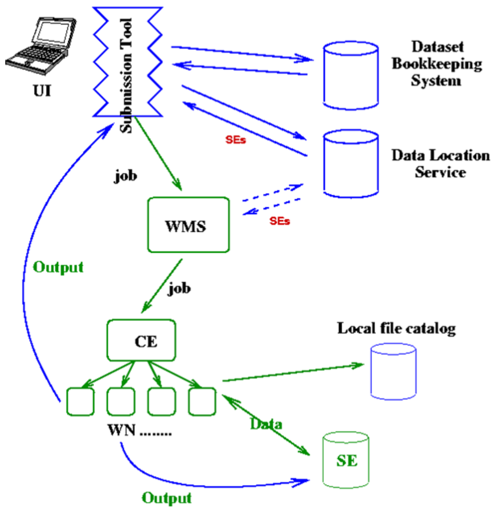

Figure 2: CRAB workflow.

Crab architecture

Crab has been designed to allow physicists to access efficiently distributed data hiding the complexity of Grid infrastructure and reducing the need of knowledge of Grid commands (Fig 2). It's main components are:

- data location classes which interact with the current CMS infrastructure to discover data

- job creation engine, which collects users' parameters along with data location information and generate wrapper scripts and jdl files suitable to the jobtype specified by the user

- job submission, monitoring and output retrieval through BOSS (**B**atch **O**bject **S**ubmission **S**ystem)[6]

- job logging and bookkeeping through BOSS and MonaLisa[7] and postmortem information through LCG Logging and Bookkeeping services

The user interacts with CRAB via a simple configuration file called crab.cfg. Along with user specific parameters, like the dataset/owner pair that the user wants to access, the specific CMS software version he wants to use, the executable developed to perform a given analysis and the total number of events to access, the configuration file contains values to instruct CRAB on how to handle the specific job type (Orca, Famos, Alpgen) and how to interact with Grid middlewares and its submission layer, that is BOSS. The user can also specify a different wrapper for his executable, in case he wants specific controls which are not foreseen on the standard one.

Once CRAB has read the configuration file, it starts querying CMS specific services which are designed to return the list of sites where data are phisically located. Up to now these services are RefDb and PubDb, which provide a php interface to allow direct query to the backends, but they're currently being substituted by two redesigned services: DBS (**D**ata **B**ookkeeping **S**ervice)[8] and DLS (**D**ata **L**ocation **S**ervice)[8]. RefDb/DBS are responsible to tell CRAB which data are available, along with specific data parameters which describe the particular set of events, while PubDb/DLS are responsible to tell CRAB where data are phisically hosted and which is the local protocol suitable to access them.
The script generator engine collects all the information and starts creating jobs, splitting the payload according to the number of events requested by the user and the total number of events available for the given dataset. It also generates the jdl files specifing the list of sites hosting data retrieved from DBS and DLS.
Before passing the control for job submission to BOSS, CRAB checks that at least one site fulfills the requirements put into the jdl files. This avoids submitting jobs to sites which, for some reasons, don't have the required CMS software version or have been removed from the Information System due to scheduled downtime or Grid middleware problems.
The control is then passed to BOSS, which performs job submission to the Grid. Currently CRAB is able to submit jobs, via BOSS, both to WLCG and OSG. After jobs submission the user has to check only the status to tell CRAB to retrieve the output once they finish. The user can decide, via the crab.cfg file, how to manage output files. He can copy its output directly on the Storage Element close to the Worker Node which run the job, store them on Castor or have it back on the User Interface from which he run CRAB.
CRAB provides also a postmortem functionality to trace

back the Grid history of all jobs in order to collect information useful for bebugging purposes.

CRAB CURRENT USAGE

CRAB is currently used by CMS physicists to analize data for the Physics Technical Design Report, which is being written now. It's also been used during the service phase of LCG Service Challenge 3, performed in November '05. The main purpose of CMS for the service challenge was data transfer and data serving infrastructure known to work for realistic use cases. This also included testing the LCG workload management components and sites reliability through the continuous submission of analysis jobs.
CRAB has been used through an automatic tool to spread jobs continuously over all published data (Fig 3). This wide usage allowed the understanding of the computing infrastructure weaknesses, tracing failures and problems.
For the future CRAB will continue to be used to submit analysis jobs to test the whole CMS computing infrastructure, providing immediate feedback on sites misconfigurations, data flow bottlenecks and CMS software failures, allowing prompt reaction to fix problems.

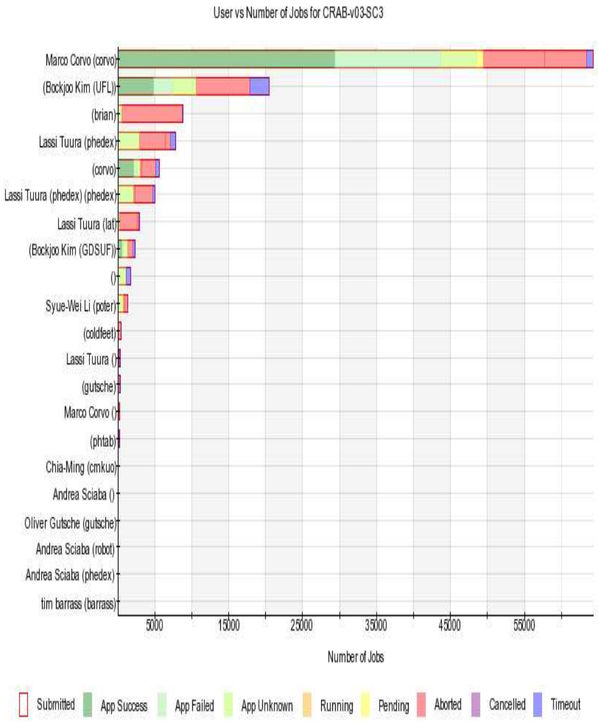

Figure 3: CRAB usage during the service phase of SC3.

FUTURE DEVELOPMENT

CMS computing tools are evolving to fit and fulfill the new requirements collected through the past experiences in the DC04 and WLCG Service Challenges. CMS core software is also evolving into a new Event Data Model which will replace the current analysis framework.
Part of these changes will reflect also in CRAB structure and interfaces, like those for DBS, DLS and the new CMS analysis software, called CMSSW. Currently a version of CRAB with DBS and DLS interfaces is under testing. This implies the use of new components based on API, rather than on php.

CRAB developments forsee also its integration into a pool of services which will allow users to interact only with a thin client, eventually their laptop. In this scenario CRAB will act as a lite interface to these services, which will provide submission, tracking, logging and bookkeeping of jobs created.

REFERENCES

[1] http://lcg.web.cern.ch/LCG/

[2] http://www.opensciencegrid.org/

[3] http://cmsdoc.cern.ch/cms/oo/repos_standalone/download/
K. Rabbertz et al. "CMS Software Distribution on the LCG and OSG Grids". CHEP '06, Mumbai.

[4] J. Rehn et al. "PhEDEx high-throughput data transfer management system", CHEP '06, Mumbai/
T. Barrass et al. "Techniques for high-throughput, reliable transfer systems: break-down of PhEDEx design", CHEP '06, Mumbai.

[5] N. De Filippis, "Data Validation Tools for CMS analysis", CMS Internal Note.

[6] C. Grandi, D. Colling et al. "Evolution of BOSS, a tool for job submission and tracking". CHEP '06, Mumbai.
C. Grandi, A. Renzi "Object Based System for Batch Job Submission and Monitoring (BOSS)". CMS Note 2003-005.

[7] http://monalisa.cacr.caltech.edu/monalisa.htm

[8] The Computing project TDR, CERN/LHCC-2005-023, Jun 2005