CMS/ARDA ACTIVITY WITHIN THE CMS DISTRIBUTED SYSTEM

J. Andreeva, C. Cirstoiu, J. Herrala, M.Lamanna , CERN, Geneva, Switzerland
T.S. Chen, S.C. Chiu , Academica Sinica, Taipei, Taiwan
A. Berejnoi, O. Kodolova , Moscow State University, Moscow, Russia
C.Munro , Brunel University, London, United Kindom

Abstract

The ARDA project focuses in delivering analysis prototypes together with the LHC experiments. The ARDA/CMS activity delivered a fully-functional analysis prototype exposed to a pilot community of CMS users. The current integration work of key components into the CMS system is described: the activity focuses on providing a coherent monitor layer where information from diverse sources is aggregated and made available for the services supporting the user activity, providing a coherent global view of the CMS activity. Of particular interest is a high-level service controlling the job execution and implementing experiment-specific policies for efficient error recovery. This control layer is in the state of advanced prototyping and experience in developing, deploying and some users feedback is presented and discussed.

INTRODUCTION

ARDA[1] is an LCG[2] project. The goal of the ARDA project is to collaborate with the LHC experiments in the distributed analysis activity and to deliver analysis prototypes constructed on top of the new generation of the middleware gLite [3] developed by the EGEE [4] project.

There are two main directions of the ARDA activity within CMS:

- Analysis framework capable of creating user jobs, submitting them, following their progress on the Grid, constructing monitoring web pages, retrieving output and resubmitting failed jobs;
- Global monitoring system, providing a coherent view of the CMS jobs processing, combining experiment specific information with the job status information provided by the Grid Logging and Bookkeeping System.

ANALYSIS FRAMEWORK

The development of the ASAP system (ARDA Support for Analysis Processing) [5] started in a very early phase of the EGEE/gLite middleware project. To get early feedback from the physics community on the new middleware, the ARDA team needed to attract pilot users and encourage them to try the new system. This was not an easy task taking into consideration the unstable and evolving environment of the development test bed. We decided to shield the user activity from possible instabilities introduced by the execution back end and more generally from the interaction of the application with the grid environment. This requires a system capable of following the progress of user jobs and resubmitting failed jobs on user behalf. A sophisticated monitoring system is therefore an essential ingredient allowing the system to manage users jobs and provide real time information for the users, this feature received very positive feedback from the CMS pilot users.

The ASAP system consists of two major parts. The client side is responsible for the job preparation and provides a simple interface to submit/resubmit jobs, query their status, cancel jobs, retrieve the output. The second part is the ASAP Task Manager service, which, on user behalf, submits the jobs, follows their progress, analyses eventual failures, resubmits failed jobs, retrieves the output and saves it at the final location. The Task Manager requires the user to delegate his credentials using the MyProxy[6] service and generates web pages illustrating the task processing progress. An example of the ASAP task monitoring page is shown in Figure 1.

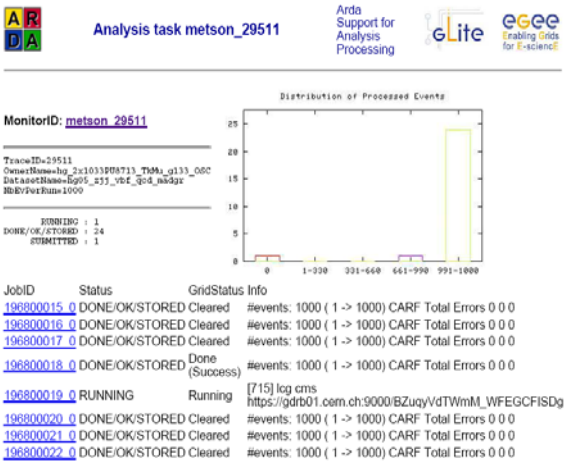

Figure 1: Example of the ASAP task monitoring page

The typical analysis activity consists of a collection of similar jobs analysing in parallel a given input dataset with the same application setup. Every job is processing a certain portion of the input dataset. A user task consists of a collection of jobs.

Job generation is performed on the client side. Once this is completed the task can be registered to the Task Manager which submits it and controls its evolution till the completion.

After the task was delegated to the Task Manager at any given moment the users can regain control over their task back and complete it in the interactive mode or delegate it later again to the ASAP task manager. The user has full flexibility either to use a simple command line interface to manage the tasks or to fully rely on the Task Manager. In this latter case, user effort to accomplish the analysis task is minimized.

ASAP takes care to prepare the jobs analysing (in parallel) all the input files in order to best profit from the available resources, minimising the user intervention (and manipulation errors). ASAP takes advantage of the notion of a task to better analyse execution errors and optimise the resubmission strategy.

The ASAP monitoring is based on the MonAlisa [7] system developed by the California Institute of Technology. The experience of CMS in using of the MonAlisa monitoring system will be described later in this paper.

CMS pilot users started to work with the ASAP system in the beginning of February 2005, initially on the gLite development test-bed. Currently two Grid middleware back ends are enabled in ASAP: gLite and LCG. The system received a positive feedback from the user community. More than 30 CMS physicists are using ASAP for their everyday work, submitting about 90K jobs per month and processing more than 30M events per month on the LCG distributed infrastructure.

The CMS experiment has developed another job submission tool CRAB [8]. Though ASAP has certain overlap with CRAB in the job generation part, the task monitoring and the task management functionality does not exist in the current CRAB implementation, therefore we are collaborating to integrate them into the final CMS analysis system as a part of the CMS integration program.

CMS DASHBOARD

The CMS Dashboard project aims to provide a single entry point to the monitoring data collected from the CMS distributed system [9]. The Dashboard development is a part of the CMS integration project [10], a joint effort of ARDA and MonAlisa team in the close collaboration with the CMS developers working on the job submission tools for production and analysis.

Recently the ATLAS experiment expressed its interest to have a monitoring system with the similar architecture and functionality and the ARDA team is preparing to provide them similar tool.

Currently the main development is concentrated on the job monitoring. The objective is to provide a complete view of the CMS activity independently of the Grid flavour. The dashboard keeps and displays the quantitative and qualitative characteristics of the experiment Grid usage by including experiment-specific information and it has to be able to indicate problems of any nature.

Some of the main monitoring indicators are listed below:

- Quantities – how many jobs are running, pending accomplished successfully, failed, per user, per site, per input data collection, Distribution of these quantities over time;
- Usage of the resources (CPU, memory consumption, input-output rates), distribution over time, aggregation on different levels;
- Sharing of the resources between analysis and production, different analysis groups, individual users;
- Grid behaviour – success rate, reasons of failures as a function of time, execution centre, data collection etc…;
- CMS application behaviour;
- Distribution of the data samples over the sites and analysis group.

Since the error rate can have different reasons (e.g. lack of stability, scalability or performance of the Grid, site misconfiguration, wrong data publishing, data access or software distribution problems, application errors, etc…) the dashboard should be able to help the user (or whenever appropriate the service providers) to detect and identify the problem.

Information Sources

Currently the dashboard uses two main sources of information – R-GMA [9] and MonAlisa.

R-GMA is used for retrieving job status information from the LCG Logging and Bookkeeping system[10]. R-GMA allows us to extract sophisticated service information from the LCG service.

For the experiment specific information and worker nodes system monitoring the Dashboard uses the MonAlisa monitoring system. MonAlisa can provide information not only from LCG but also from jobs running on other infrastructures (such as OSG[11] or even traditional batch facilities) where R-GMA is not available.

The R-GMA and MonAlisa systems are complementary regarding the type of information provided. However certain data is available via both sources. In the latter case we use the opportunity to investigate the behaviour of the two systems.

Overview of the architecture

The CMS job submission tools and the corresponding job wrappers have been instrumented to send messages containing simple key-value pairs to the MonAlisa server.

These messages are sent at job submission time and from the worker node. The messages sent at the submission time contain information about the task and the Grid job identifier for every submitted job. The Grid job identifier is used to correlate information for the same job obtained from R-GMA and MonAlisa.

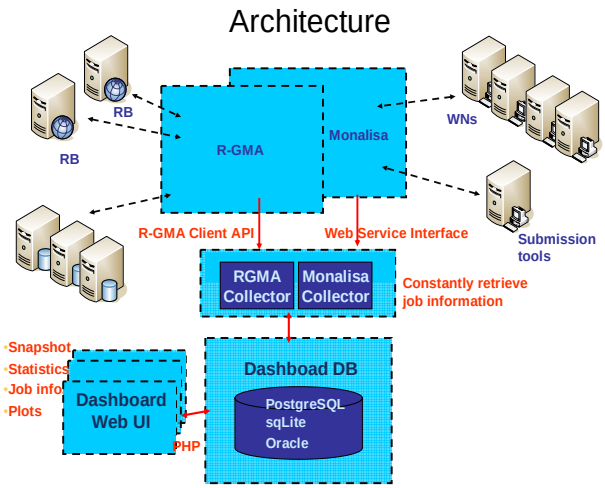

Figure2 : The dashboard architecture

A Web User Interface is provided to access monitoring information.

The schema of the Dashboard architecture is presented in Figure 2.

First experience

The first prototype of the Dashboard was developed for the CMS service challenge SC3[12] at the end of September 2005. The service challenge team provided a lot of feedback and contributed to the improvement of the system.

The Dashboard provided Grid and experiment specific monitoring information for service challenge jobs and demonstrated the capability to follow the processed jobs in real time, aggregating monitoring data at different levels and indicating different kind of problems.

The R-GMA information collected during SC3 is being used to provide detailed information on the behaviour of the grid services. In the future this functionality will be automatised in order to provide useful information for the deployment of the grid infrastructure and for the improvements to the middleware itself.

An example of the Dashboard monitoring page displaying the distribution of the SC3 jobs by site is shown in Figure 3.

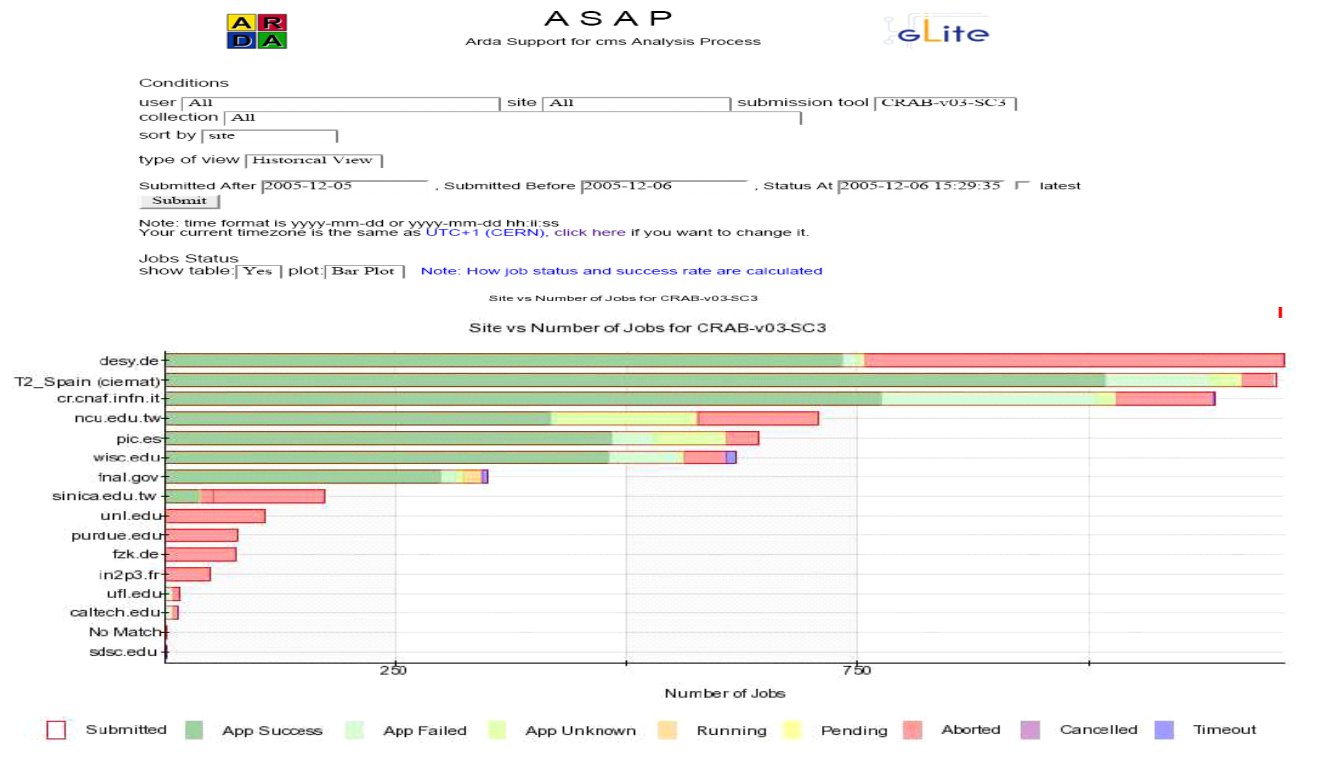

Figure 3: Example of the Dashboard monitoring page

The graphs are clickable which allows users to “drill-down” to find more specific information. For example clicking on a certain site provides the distribution of datasets at the site.

Monitoring information is also displayed in the tabular form, showing Grid and application success rates. The table is also clickable and user can get very detailed information for a given group of jobs - Grid job identifier, name of the task, job failure reason from the Grid or application point of view, time stamps of job processing.

The Dashboard supports interactive queries to the data base aggregating the available information and the user can define the time range, the category by which the jobs should be classified, and in what form information should be presented (a plot or a table or both).

In addition to the interactive mode a set of the predefined views is provided. Some level of interactivity is required to enable to pin down the origin of a given problem, but this may impact performance. A set of well defined predefined views would cover most of the use cases and should improve the overall system performance.

In the future the Dashboard can play a more active role, not only collecting and displaying information, but also analysing it and sending alarms in the case of the evident problems.

CONCLUSIONS

The ASAP prototype has been developed in our team and benefit by a close feedback loop with our pilot users. The positive users’ feedback validated our approach and we are preparing for the integration of our components in the final CMS analysis system. Our experience with ASAP suggested the development of the Dashboard.

The first prototype of the Dashboard and its use during CMS Service challenge demonstrated the need in the experiment for such a monitoring system combining Grid and experiment specific information and not depending on the middleware platform(s) used by the experiment.

We are currently using the experience we gained from this first prototype to produce a reliable product.

ACKNOWLEDGEMENTS

We would like to thank the LCG and EGEE projects for support and useful discussion (in particular the CERN IT/GD group and the gLite team). We acknowledge a very fruitful collaboration with the MonAlisa team. A special thank you is due to the entire IT/PSS/ED section for fruitful collaboration and stimulating discussion. We are very grateful to the CMS ASAP users for their valuable feedback and suggestions and to the CMS SC3 team which contributed to the improvement of the Dashboard monitoring system.

This work was performed within the EGEE project, which is funded in the European Commission under contract INSO-RI-508833. It also received support from Federal Ministry of Education and Research (Bundesministerium fur Bildung und Forschung), Berlin, Germany.

REFERENCES

[1] M.Lamanna, “ARDA experience in collaboration with the LHC experiments”, CHEP06, Mumbai, India, Feb 2006.

[2] L.Robertson, “The LHC Computing Grid Service”, CHEP06, Mumbai, India, Feb. 2006.

[3] http://egee-jra1.web.cern.ch/egee-jra1/.

[4] http://egee-innntranet.web.cern.ch/egee-intranet/gateway.html.

[5] http://www-asap.cern.ch/.

[6] http://grid.ncsa.uiuc.edu/myproxy/.

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

[8] M.Corvo, “CRAB-a tool to enable CMS Distributed Analysis”, CHEP06, Mumbai, India, Feb 2006.

[9] http://hepunx.rl.ac.uk/egee/jra1-uk/glite-r1/.

[10] http://glite.web.cern.ch/glite/lb/.

[11] F.Wuertwein, Pordes Ruth, “The Open Science Grid”, CHEP06, Mumbai, India, Feb. 2006.

[12] L.Tuura, “CMS experience in LCG SC3”, CHEP06, Mumbai, India, Feb.2006.