# Table of Contents

EGI User Forum 2011 SA3 Contributions

## Abstracts

Add your abstract here

HammerCloud: An Automated Service for Stress and Functional Testing of Grid: D van der Ster

CORAL - A Relational Abstraction Layer for C++ or Python Applications

An insight into the ATLAS Distributed Data Management

Usage and monitoring of transfer statistics in the ATLAS Distributed Data Management

An overview of CMS Workload Management for data analysis

Monitoring of the LHC computing activities during the first year of data taking

Experiment Dashboard providing generic functionality for monitoring of the distributed infrastructure

Ganga-based tools to facilitate distributed analysis, job monitoring and user support in a Grid environment

## Training Proposals

Example

Title

Contributions

Ganga User Tutorial

Developing Grid Applications with Ganga: A Case-Study of the HammerCloud Stress Testing System

Using HammerCloud: A Site Stress Testing Tool for HEP VOs

How to enable monitoring of the infrastructure from the point of view of a given VO

Managing a relational database schema using the Python API of CORAL
EGI User Forum 2011 SA3 Contributions

This is an internal page to collect, edit and manage contributions for EGI 2011 User Forum:
http://uf2011.egi.eu/Call_for_participation.html
**Abstracts**

**Add your abstract here.**

**HammerCloud: An Automated Service for Stress and Functional Testing of Grid: D van der Ster**

**CORAL - A Relational Abstraction Layer for C++ or Python Applications**

Presenter: R. Trentadue or A. Loth

**Overview** The huge amount of experimental data from the LHC and the large processing capacity required for their analysis has imposed a new approach involving distributed analysis across several institutions. The non-homogeneity of policies and technologies in use at the different sites and during the different phases of the experiment lifetime has created one of the most important challenges of the LHC Computing Grid (LCG) project. In this context, a variety of different relational database technologies may need to be accessed by the C++ client applications used by the experiment for data processing and analysis. The Common Relational Abstraction Layer (CORAL) is a software package that was designed to simplify the development of such applications, by screening individual users from the database-specific C++ APIs and SQL flavours.

**Description**

CORAL is a C++ software package that supports data persistency for several relational database backends. It is one of three packages (CORAL, POOL and COOL) that are jointly developed by the CERN IT Department and the LHC experiments within the context of the LCG Persistency Framework project. The CORAL API consists of a set of abstract C++ interfaces that isolate the user code from the database implementation technology. CORAL supports several backends and deployment models, including local access to SQLite files, direct client access to Oracle and MySQL servers, and read-only access to Oracle through the Frontier/Squid and CoralServer/CoralServerProxy intermediate server/cache layers. Users are not required to possess a detailed knowledge of the SQL flavour specific to each backend, as the SQL commands are executed by the relevant CORAL implementation libraries (which are loaded at run-time by a special plugin infrastructure, thus avoiding direct link-time dependencies of user applications against the low-level backend libraries).

**Impact**

CORAL provides generic software libraries and tools that do not specifically target their data models and could therefore be used in any other scientific domain to access relational databases from C++ or python applications.

**Conclusions**

The CORAL software is widely used for accessing from C++ and python applications the data stored by the LHC experiments using a variety of relational database technologies (including Oracle, MySQL and SQLite). It provides generic software libraries and tools that do not specifically target the data models of the LHC experiments and could therefore be used in any other scientific domain.

Type: oral presentation

Time: 30 minutes
An insight into the ATLAS Distributed Data Management

ATLAS, one of the four LHC experiments, fully relies on the use of grid computing for offline data distribution, processing and analysis. This presentation will give an insight about how the experiment's Distributed Data Management project, built on top of the WLCG middleware, ensures the replication, access and bookkeeping of multi-petabyte data volumes across more than 100 distributed grid sites. Those in attendance will get an overview of the architecture and operational strategies of this highly automated system, as well as learn details about different subsystems and monitoring solutions that could be of interest for other communities. The ideas and concepts presented will provide inspiration for any VO that is currently planning to move their data to the grid or working on improvements to their usage of grid, network and storage resources.

Type: oral presentation

Time: 30 minutes

Infrastructure required: Projector

Presenter: Fernando Barreiro

Usage and monitoring of transfer statistics in the ATLAS Distributed Data Management

The data placement and dashboard frameworks in the ATLAS Distributed Data Management project have been instrumented to measure the durations of gLite File Transfer Service (FTS) transfers between grid sites and store them in a historic database. These transfer durations are then used to generate periodic throughput statistics that are made available through an open API. The transfer statistics are reused for optimization of the source selection and for efficient cross-cloud data transfers between end-points which are not communicated through dedicated FTS channels in the hierarchical tier model. Additionally, a visualisation framework has been put in place to estimate the throughput performance of the network links. This presentation proposes to give a practical overview of the system and show how the collected statistics can be fed back into the system in order to optimise network usage and source selection. Type: oral presentation

Time: 30 minutes

Infrastructure required: Projector

Presenter: Fernando Barreiro

An overview of CMS Workload Management for data analysis.

CRAB (CMS Remote Analysis Builder) is the CMS tool that allows the end user to transparently access distributed data. CRAB interacts with the local user environment, the CMS Data Management services and with the Grid middleware; it takes care of the data and resource discovery; it splits the user's task into several processes (jobs) and distributes and parallelizes them over different Grid environments; it performs process tracking and output handling. This presentation will give an overview about architecture adopted with the aim to highlight the possibilities for eventual extension of the tool to non HEP specific use cases. Current usage, scalability and operational strategies of the system will be also presented.
Monitoring of the LHC computing activities during the first year of data taking

Presenter: Edward Karavakis

The Worldwide LHC Computing Grid provides the Grid infrastructure used by the experiments of the Large Hadron Collider at CERN which this year started data taking. The computing and storage resources made available to the LHC community are heterogeneous and distributed over more than a hundred research centers. The scale of WLCG computing is unprecedented; the LHC virtual organisations (VOs) alone run 100,000 concurrent jobs and the ATLAS VO can sustain an integrated data transfer rate of 3GB/s. Reliable monitoring of the LHC computing activities and the quality of the distributed infrastructure is a prerequisite of the success of the LHC data processing. The Experiment Dashboard system was developed in order to address the monitoring needs of the LHC experiments. It covers data transfer and job processing and works transparently across the various middleware flavours used by the LHC VOs. The system plays an important role in the computing operations of the LHC virtual organisations, in particular of ATLAS and CMS, and is widely used by the LHC community. For example, the CMS VO’s Dashboard server receives up to 5K unique visitors per month and serves more than 100,000 page impressions daily. During the first year of the data taking the system coped well with growing load both in terms of the scale of the LHC computing activities and in terms of number of users. This presentation will describe the experience of using the system during the first year of LHC data-taking, focussing on the Dashboard applications that monitor VO computing activities. Those applications that monitor the distributed infrastructure are the subject of a different presentation, “Experiment Dashboard providing generic functionality for monitoring of the distributed infrastructure”. Though primarily the target user communities of the Experiment Dashboard are the LHC experiments, many of the Experiment Dashboard applications are generic and can be used outside the scope of the LHC. Special attention in this presentation will be given to generic applications like job monitoring, and the common mechanism to be used by the VO-specific workload management systems for reporting monitoring data.

Experiment Dashboard providing generic functionality for monitoring of the distributed infrastructure

Presenter: Pablo Saiz

The Worldwide LHC Computing Grid delivered a scalable infrastructure for the experiments of the Large Hadron Collider at CERN which this year started data taking. Reliable monitoring is crucial for achieving the necessary robustness and efficiency of the infrastructure and, to a big extent, defines the success of the LHC computing activities. On the other hand, monitoring of the WLCG infrastructure is a challenging task since the infrastructure is huge and heterogeneous; it comprises different middleware platforms (gLite, ARC and OSG) and integrates more than 170 computing centers in 34 countries. In order to provide monitoring of the distributed sites and services the Experiment Dashboard system developed several generic solutions which are shared by the LHC experiments but can be also used by other virtual organisations. The Dashboard
applications for infrastructure monitoring are used by the LHC virtual organisations for the computing shifts and site commissioning activities. This presentation will describe site/service monitoring applications and highlight the possibility of using these applications outside the LHC domain.

Type: oral presentation

Time: 30 minutes

Infrastructure required: Projector

**Ganga-based tools to facilitate distributed analysis, job monitoring and user support in a Grid environment.**

Presenter: Mike Kenyon

The end users of Grid computing resources demand that the tools they use are reliable, efficient and flexible enough to meet their needs. Most users, irrespective of the research community to which they belong, are generally not interested in developing Grid-access tools, and nor should they be. Their role is to exploit the resources available as effectively as possible, and with minimum knowledge of how the underlying technologies function.

To facilitate this, a wide range of Grid-enabled tools have been developed which aim to shield the user from the complexity of distributed infrastructure technology. Ganga is one such tool, designed to effectively provide a homogeneous environment for processing data on a range of technology "back-ends", ranging in scale from a solitary user's laptop, up to the integrated resources of the Worldwide LHC Computing Grid.

Initially developed within the high-energy physics (HEP) domain, Ganga has since been adopted by a wide variety of non-HEP user communities as their default analysis and task-management system. This presentation will use recent case-studies to highlight some of these successes and illustrate the ease with which users can start working productively with Ganga.

In addition to providing a stable platform with which to conduct user analysis, the Ganga development team have deployed a range of monitoring tools and interfaces. We will present developments of the GangaMon service, a web-based tool that allows users to monitor the status of tasks submitted from within the Ganga environment. This service is also an integral part of the user-support infrastructure, as it allows users to directly upload "task crash reports" from within Ganga to a repository that can be accessed by the support team. This error-reporting tool will be described, with specific reference to how it has been adopted by the CMS VO, a community who have their own task-management system in place of Ganga, yet who were able to easily integrate their system with the technology underlying the GangaMon service.

Type: oral presentation

Time: 30 minutes

Infrastructure required: Projector
Training Proposals

Example

Please provide your contributions in the following format:

Title

Abstract. Just copy this entire section and paste it at the bottom of the page and modify it. Type: oral presentation, demo, hands on workshop, ...

Time: n hours

Infrastructure required: describe what you need here or what you require from your participants

- hardware (projector, participants’ laptops, training workstations,...)
- software (OS, grid certificates, VO, ...)
- network and services, ...

Number of participants: min-max (if applicable)

Comments: anything else which is worth mentioning

Contributions

Ganga User Tutorial

The tutorial will allow the participants to understand basic concepts of Grid job management: configuration, submission, monitoring of jobs and retrieval of results with Ganga -- an easy-to-use frontend for the configuration, execution, and management of computational tasks in a variety of distributed environments including Grid, Batch and Local resources. Participants will learn how to make use of basic mechanisms such as file sandboxes, datasets and job splitting to best address their application needs. They will also learn how locally available resources may be used for running small-scale tasks and how to subsequently easily transition to using Grid resources for large-scale tasks. The hands on sessions will also cover monitoring: participants will learn how to keep track of their jobs through several web-based interfaces, including the Dashboard services. The hands on exercises are provided online: https://twiki.cern.ch/twiki/bin/view/ArdaGrid/EGEETutorialPackage

Type: presentation + hands-on session

Time: 4 hours (half day)

Infrastructure required:

- projector
- laptops may be used if remote training accounts are provided (see below), else local training accounts on training workstations should be provided
- training accounts with gLite UI installed (and preferably a local batch system)
- user certificates and VO access (e.g. gear or gilda)
- network access

Number of participants: 5-10 per trainer
Developing Grid Applications with Ganga: A Case-Study of the HammerCloud Stress Testing System

Probably this should be merged with the Ganga tutorial above

This demo/tutorial will present an introduction to developing EGI grid applications using Ganga. Ganga is most commonly used as an end-user interface to the EGI and other grids, but its included Grid Programming Interface (GPI) also allows application developers to easily submit and manage jobs to the various grid backends using Python. The tutorial will be formulated as case study of the development of HammerCloud, a distributed analysis testing tool employed by three HEP VOs. The tutorial will also highlight some of the more powerful features of Ganga, such as the GangaRobot module and how to incorporate multi-threading in your Ganga applications.

Type: demo/tutorial Time: 30-60 minutes Infrastructure required: Projector, PC with network access

Using HammerCloud: A Site Stress Testing Tool for HEP VOs

Probably I withdraw this in favour of one HammerCloud talk (above)

HammerCloud (HC) is a distributed analysis stress testing tool that is available for three HEP VOs: ATLAS, CMS, and LHCb. This tool enables site and regional administrators to customize and schedule on-demand tests of their computing facilities using typical analysis jobs drawn from the user communities, without requiring VO-specific knowledge. The tests sent by HammerCloud are useful to help commission new sites, to evaluate changes to software or configurations and to benchmark sites for comparison purposes. The results of the HammerCloud tests are presented in a friendly web interface and users can drill down into the results to get performance statistics and detail metrics related to the job performance (e.g. CPU times, storage I/O times, etc...).

Type: demo/tutorial Time: 30 minutes Infrastructure required: Projector, PC with network access

How to enable monitoring of the infrastructure from the point of view of a given VO.

In order to use the distributed infrastructure in an efficient way it is important to enable monitoring of the infrastructure from the VO perspective. The training will describe the existing systems which provide this functionality. Namely, the new implementation of Site Availability Monitor (SAM) based on Nagios, Dashboard Site Usability user interface, Dashboard Site Status Board and SiteView application. The participants will learn how to design VO-specific SAM tests, how to provide a description of the topology of the infrastructure used by a particular VO, how Site Status Board can be populated and used to show the status of the infrastructure and various VO computing activities.

Type: demo/tutorial

Time: 30-60 minutes

Infrastructure required: Projector, PC with network access

Managing a relational database schema using the Python API of CORAL

Presenter: A. Loth or R. Trentadue

Overview

Developing Grid Applications with Ganga: A Case-Study of the HammerCloud Stress Testing System
The CORAL C++ software is widely used in the LHC Computing Grid for accessing the data stored by the LHC experiments using relational database technologies.

**Description**

CORAL supports data persistency for several backends and deployment models, including local access to SQLite files and remote client access to Oracle and MySQL servers, either directly or through intermediate server/cache layers. In this demonstration, PyCoral will be used to show how CORAL allows users to create, populate and read relational tables.

**Impact**

CORAL provides generic functionalities that do not specifically target the data models of high-energy physics experiments and could be used in any other scientific domain. In addition to its C++ API, CORAL also provides a Python API (PyCoral) which is particularly useful for fast prototyping of relational applications from an interactive shell.

**Conclusion**

In this demonstration, PyCoral will be used to show how CORAL allows users to create, populate and read relational tables. In particular, it will be shown how the same CORAL code can be used to store and retrieve relational data on the Grid using different backends, such as SQLite files, Oracle databases or the Frontier read-only servers and caches.

Type: demo/tutorial

Time: 30 minutes

Infrastructure required: Projector, PC with network access