Grid-based Simulation Computing in ALICE

P. Buncic, F. Carminati, P. Saiz
CERN, CH
D. Mura
INFN, Cagliari, Italy
S. Bagnasco, P. Cerello, S. Lusso
INFN, Torino, Italy
M. L. Luvisetto
Università di Bologna and INFN, Bologna, Italy
R. Barbera
Università di Catania and INFN, Catania, Italy
M. Sitta
Università del Piemonte Orientale and INFN, Torino, Italy
M. Masera
Università di Torino and INFN, Torino, Italy

The ALICE Collaboration is an active member of the EDG (also known as DataGrid) project since the very beginning; it took active part in the definition of user requirements and detailed use cases, as well as in the evaluation of the EDG testbed, all along its evolution. In December 2001, ALICE ran the first application job on the testbed, the simulation of a low multiplicity Pb-Pb event at the LHC energy.
In parallel, because of the deadlines set by its scientific program, ALICE developed the *AliEn* environment, on which simulated data are regularly produced, reconstructed and analysed on a distributed system scattered over four continents. *AliEn* has been interfaced with EDG, in order to allow the coordinated management of both ALICE-owned and EDG resources: recently, the simulation of a central *Pb-Pb* event, submitted as an *AliEn* job with the requirement to run on EDG resources, was successfully completed, with output registration in the Data Management System on both sides and retrieval of the output, stored on EDG, from the *AliEn* shell.
Meanwhile, making use of an ALICE developed WEB interface, a continuous monitoring of the EDG testbed, with an increasing job and data rate, is in place to evaluate the system stability and to define its breaking point.
After two years of work several goals have been achieved, while some items still need improvements in order to run large productions and analysis on a stable, distributed, scalable and interoperable system.

1. Introduction

The unprecedented scale of LHC computing, with respect to both the amount of data to be managed and the CPU required to process and analyse them, and the intrinsically distributed nature of LHC Collaborations, with participating institutions and researchers scattered all over the world, are strong requirements pushing offline projects toward a GRID approach.

In the specific case of ALICE, a distibuted approach was already applied to the application software development, which has been taking place for almost five years, involving a team of about 50 people from several member institutions. On the *AliRoot* framework, generated by that effort, all the simulations being produced, reconstructed and analysed in view of the *Physics Permormance Report* are based.

Software development does not impose a GRID approach, since it does not involve a huge amount of data neither it requires large computing power; however, in the case of ALICE, it proved to be an important factor for the definition of the basic nodes of the *Virtual Organisation* (VO), which are generally located in sites where a sizable software development is taking place.

Presently, about 30 sites in four continents are contributing with storage and computing resources to ALICE physics data challenges.

2. The AliRoot framework

The *AliRoot* development started in 1998: a vision common to the whole team took to the decision to base it on the *ROOT* framework [1] and to concentrate the whole available manpower on the creation of that new object-oriented framework for the ALICE-specific software. The system, altough reflecting the intrinsic modularity set by the sub detector units, required also a set of steering classes. The sharing of responsibilities mapped those for the detector construction: the CERN core team, with few contribution from other institutions is in charge of the issues common to all detectors, while most of the detector simulation and reconstruction algorithms were developed and tuned by the experimental groups based outside CERN.

Presently, the system is mature enough to be used for large scale data challenges and for the optimisation of algorithms for the selection of rare events. In the coming years, all the issues related to the processing of real data (i.e. conditions databsed interface, detector calibrations, alignment, ...) will be addressed.

3. The ALICE Physics Performance Report

ALICE data-taking conditions will significantly differ with respect to the other LHC experiments. ALICE will run at low luminosity (up to 6 orders of magnitude smaller than ATLAS and CMS), but will detect events with a very large track multiplicity, expected in the range 50,000-85,000 primary particles/event in the case of central $Pb-Pb$ collisions.

The simulation of such events (Monte-Carlo based particle transport through the detector and detector and electronic response) with *AliRoot* takes several hours (about 24 hours/event on a 800 MHz PIII CPU), requires a large amount of memory (400 MB average, with temporary peaks up to 1 GB during the execution) and generates a "monster" output of about 2 GB/event.

The event reconstruction, altough not comparable, is also demanding and requires several minutes.

Moreover, central events are typically used as "background" and superimposed to physics signals (i.e., the $D^0 \rightarrow K\pi$ decay): that procedure is called "event mixing" and, in order to get a meaningful statistics on the signal acceptance and detection efficiency, it is often required to mix the same background event to several simulated signals. In other words, the smaller time needed for event reconstruction with respect to its simulation is almost compensated by the number of times the operation is repeated.

As a reference, a useful simulation sample is composed of several thousand background events and therefore generates a global output size of several TeraBytes.

Such an effort could be carried on with standard tools, with a predefined load sharing between several computing centres. However, the availability of reliable GRID services would simplify the task and allow the real time access to distributed data to all the analysis team members.

4. AliEn Grid Services

AliEn (Alice Environment) [2] is an Open Source tool for the configuration of a *Virtual Organisation* and the corresponding management of a set of distributed computing and storage resources. Its development started in 2001, based on the already existent $NA49$ Data Catalogue and on the assumption that, whenever possible, available Open Source tools should be used and integrated to provide Grid-like functionality.

Since October 2001, *AliEn* is in use for all the ALICE physics data challenges; it presently manages more than 30 sites, in 4 continents, with related ComputeElements (CE) and StorageElements (SE).

Three main features distinguish its functionality with respect to other available Grid infrastructures:

- the job management is based on a *pull* mechanism: a central server collects all the job descriptions, and stores them in a *first-in, first-out* queue. The attached CE's, whenever available to process a job, connect to the server and pull the job from the queue. At first sight, it may seem that a central queue could be a bottleneck. In fact, that approach minimises the number of operations to be performed by such service, delegating all of them to the client CE's. On the other hand an EDG ResourceBroker collects a large number of job requests and is also in charge of optimising the CE selection; therefore, the load is dramatically increased.

- the Data Catalogue is based on a MySQL layer, and stores files as entries in MySQL tables. Therefore a file collection, stored in the same table, emulates the concept of a directory. However any entry could also be a pointer to another table: that feature allows the creation of a hierarchical, UNIX file system - like structure. That, in turn, allows to navigate it through a command line interface and a WEB interface, as well as to minimise the query response time.

- the Data Catalogue service allows to store metadata, in the form of MySQL table(s) attached to any Data Catalogue entry describing a file.

5. ALICE and Grid Projects

Several projects focused on GRID middleware developments were launched in recent years, all over the world. Because of the structure of the ALICE Collaboration, with a core team at CERN, a large European participation and a small US component, ALICE was mainly involved in the EDG (*aka* DataGrid) and EDT (*aka* DataTag) projects.

However, the ALICE Offline Team deployed *AliRoot* on the EDG testbed with an approach that kept the implementation of simulation and reconstruction algorithms completely decoupled. from the invocations of EDG Services and the deployment on the EDG testbed. Interfacing *AliRoot* to Grid Services from another "provider" would therefore be almost straightforward.

The quick and successful development of the *AliEn* framework triggered the decision to select its Grid Services as the base for ALICE simulation data challenges; that, in turn, made it extremely important to develop an *AliEn* interface to the EDG *middleware*, with the goal of reaching a full interoperability between the two systems and therefore a full exploitation of the available EDG-managed resources. In the long

term, such an approach would allow to take advantage of available resources regardless of the running *middleware*.

ALICE decided to concentrate the activity on three items, which are related to its medium term strategy in the approach to Grid computing:

- monitor the EDG-testbed efficiency as a function of time in a non invasive way;
- run a large size test production on it;
- develop and test the *AliEn*-EDG interface.

In fact, some of the boundary conditions played an essential role in the definition of the ALICE strategy:

- the future availability of the LCG Testbed, which will be mostly based on the middleware developed by EDG;
- the existence of a certain number of sites directly managed by ALICE, with no expected resource sharing with other Virtual Organisations, and not enough manpower to configure and maintain an EDG site;
- the extremely satisfactory behaviour of the *AliEn* system, on which the ALICE productions run were based until now, up to several TB's of produced output.

ALICE would like to be able to fully exploit both kinds of resources (owned and not owned). In order to do that, an interface between the *AliEn* and the EDG services is required: its development is presently in an advanced status, and it already allowed the use of EDG resources under the *AliEn* control.

The entire EDG (and, later on, LCG) testbed will be seen by *AliEn* as a single (few), huge Compute Element(s) and Storage Element(s). In addition, the *AliEn* Data Catalogue and Storage Elements will be writable and readable by EDG nodes.

The key idea is that data produced by EDG(LCG) will be registered both in the *AliEn* and in the EDG(LCG) Data Catalogue, with *AliEn* PhysicalFileName equal to EDG(LCG) LogicalFileName. In this way, any data produced by EDG(LCG)/*AliEn* will be accessible for *AliEn*/EDG(LCG) nodes.

6. AliRoot and AliEn deployment on the EDG testbed

The modular structure of all the components of the *ALICE Offline Framework* made it easy to distribute, install and configure them everywhere required.

The source code of *ROOT*, *AliRoot* and *AliEn* is available from their respective CVS Servers for all the tagged versions: all these products are portable on several platforms, including all the UNIX flavours.

In order to be compliant with the EDG procedure for the installation of application software, all these tools are also published in the form of RPM files on the *ALICE-Grid* and the *AliEn* web sites.

In parallel, the dynamical installation and configuration of the application software, considered as one of the most important GRID use cases, was successfully deployed on the EDG testbed: we could submit jobs that, on the selected Worker Node (WN), connected to the *ROOT* and *AliRoot* CVS Servers, downloaded and compiled the requested version of the source code and ran a simulation and/or reconstruction algorithm.

The first version of the EDG testbed was opened to the applications in December, 2001; few hours later, ALICE could successfully run the first *AliRoot* simulation job, a peripheral *Pb-Pb* event with fairly low multiplicity (fig. 1). However, it proved that *AliRoot* was fully Grid-compliant.

The job output, as well as its input (describing the kind of simulated event), was stored in the ReplicaCatalogue (RC) set up at NIKHEF for the ALICE Virtual Organisation.

That very promising start, which proved the EDG functionality was suitable for running simulation productions, was unfortunately not associated to the required system stability: EDG Grid Services were still in an early development stage and were not capable of managing thousands of jobs, lasting for more than 24 hours each, at the same time.

Also, ALICE felt the lack of a hierarchical (i.e., file-system like) structure for the ReplicaCatalogue was a serious limitation in case of registration of several thousands files. Even worse, the lack of a command line and C++ Application Program Interface (API) to the ReplicaCatalogue made its use unfriendly, both for browsing and for accessing data.

The instability of Workload Management Services on the time scale of a few hours was particularly bad in case of our long jobs, turning out into a high inefficiency.

Finally, the EDG application testbed was not conceived as a framework for large scale data challenges, since the overall amount of available storage, to be shared with other applications, was too small.

7. Monitoring the EDG testbed

Given the quickly evolving situation, ALICE felt important to develop a tool which could regularly monitor the testbed efficiency with a minimum interference on its status. A test suite, composed by a Perl script activated every day by a UNIX cron, a database powered by MySQL DBMS and a web interface to query the database, ws set up: it must be installed on an EDG User Interface (UI), so as it can interact with the Resource Brokers (RB) and test the

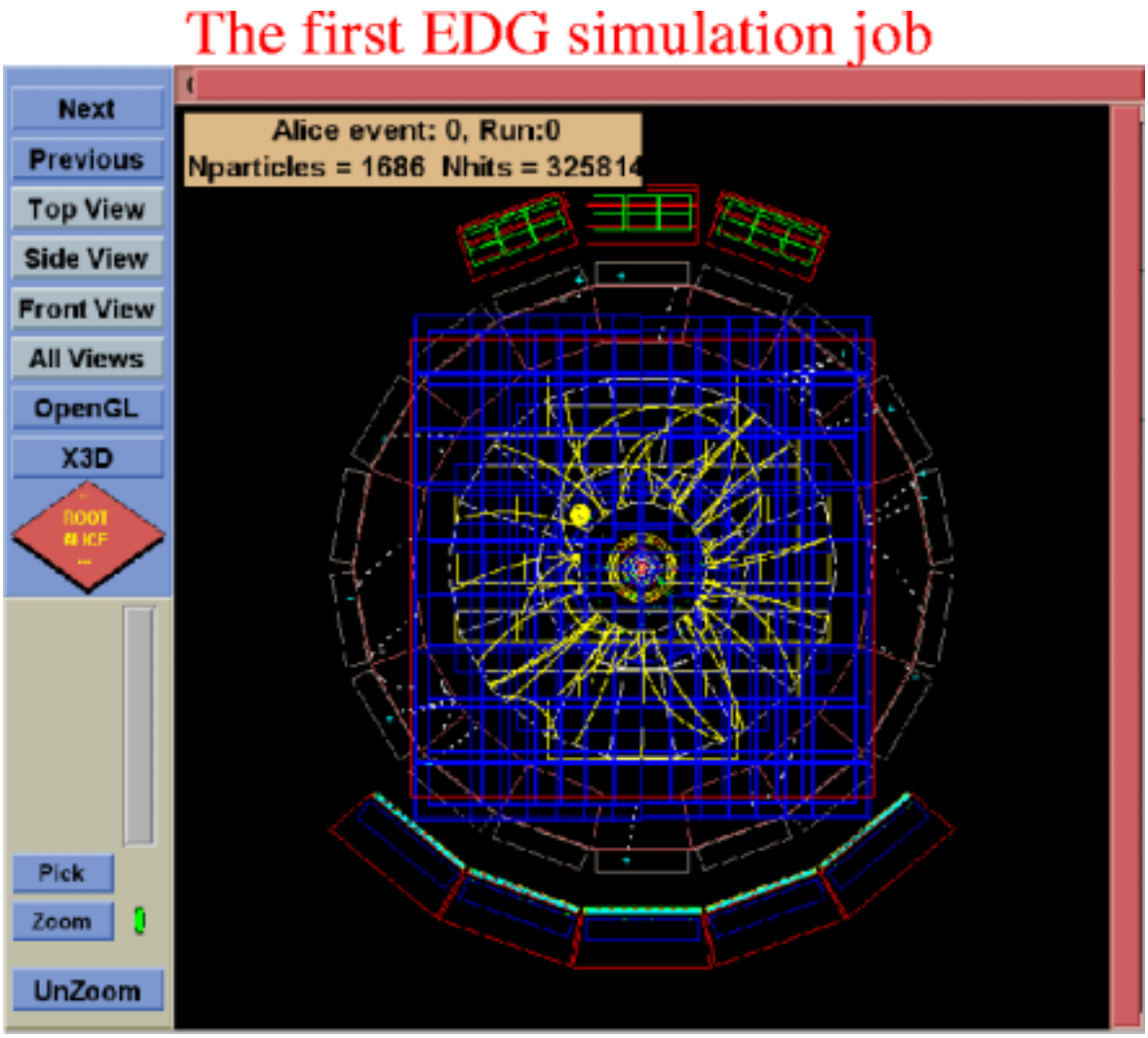

Figure 1: A peripheral Pb-Pb event tracked through the ALICE detector: the first full event simulation run on the EDG testbed, in december 2001.

Compute Elements (CE) and Storage Elements (SE) availability.

A UNIX cron job launches every night a Perl script, which drives the job submission and evolution.

It is important to remark that the jobs sent and monitored by the script, being described by a *Job Description Language* (JDL) file, could be of any kind; therefore, the test suite could be used by other applications with a minimum effort in the configuration change. On the other hand, it is obviously useless from the middleware point of view, since on that side all the information on the system status is directly available and evaluated on the full job sample. In particular, ALICE send event simulation and reconstruction jobs with only few hundred primary particles (to be compared to about 50,000-85,000 particles for a full central event).

The script starts by getting the list of available CE's, using the proper information provided by the Resource Broker(s); for each active CE that meets the job requirements, a short *AliRoot* test job is submitted. The evolution of the test job is monitored by a routine within the script, which periodically retrieves the job status. Once the job execution is terminated, the second part of the script makes an attempt to contact, for each CE, the closest SE and copy a file onto it. Each test ends with the creation of a log file, which can be picked-up, inspected and analyzed from the web interface. The result of each test job is stored in a database structure, based on a MySQL server/client architecture, updated by the Perl script using Perl/DBI.

The website [3] is an interface to the database of results: it offers the possibility to query the test results for a selected time interval. It is also possible to select by Resource Broker, jdl file and CE name.

Each of the options performs a query to the MySQL database, using PHP as server-side scripting language;

RB	Status Sub.	Status Sch.	Status Runn.	Status Done	Status Out. Ready
CERN	648	240	239	238	238
CNAF	599	253	241	239	239
Total	1247	493	480	477	477

Table I Results obtained with the automatic submission tool, corresponding to the CERN and CNAF resource brokers and connected compute elements. Sub. = submitted, Sch. = scheduled, Runn. = running, Out. = output

the outputs are arranged in a tabular format and shown on the screen. Each row contains the node name as provided by the Resource Broker, the submission date, the log file, the RB status, the closest CE, followed by the run information and the storage information, according to the actual database layout.

The results of the EDG testbed monitoring confirmed the qualitative understanding given by the first period of operation: the functionality is fairly satisfactory for simulation *production jobs* (i.e., jobs with a small input and - usually - a large output, sent in a coordinated way). In fact, when the job submission is successful and the job duration is short (i.e. for events with low multiplicity), the completion rate is about 97 %. We report in table I the results obtained for the CERN and CNAF resource brokers and the connected CE's, for a period of about two months.

Altough the results, at a first sight, seem quite satisfactory, we remark that, for a large fraction of time, the workload management services are not active, due to system instabilities. That feature turns into a large failure rate, up to 60-70 %, for long jobs, corresponding to central Pb-Pb events and makes it very difficult to use the EDG testbed for an ALICE physics production. ALICE decided to try a physics production anyway, since it provides an evaluation of the EDG testbed behaviour in realistic conditions.

8. Running a production on EDG

The ALICE use of the DataGrid testbed is still ongoing at the time of writing. Our goal is running a p̈hysicsp̈roduction of $5,000$ central $Pb-Pb$ events (order of $84,000$ primary particles per event in the whole phase space), generated by the HIJING ion-ion event generator, tracked through the ALICE detector by the GEANT3 program and reconstructed by the ALICE reconstruction program, all of it managed by the *AliRoot* framework.

Those events will then be the input of Hanbury-Brown-Twiss particle correlation analysis, needed for the preparation of the ALICE Physics Performance Report. The required computation effort is challenging: the production of those events would take more than 13 years on a single $100 SpecInt2000$ CPU and the expected size of all the output files will be in the order of $8-9\ TB$.

Given the restrictions on SE disk space availability and the expected output global size, only a small fraction (about $5-6\%$) of the generated files will be stored on d̈iskS̈E's, mainly to allow the testing of the *AliEn*-EDG interface. All the rest will be staged on ẗapeS̈E's at sites where a Mass Storage System (MSS) is available (CERN, CNAF, Lyon, RAL and SARA), which will be selected by a tag included in the JDL file describing the job.

In order to reach the ambitious goal of such a p̈hysicsp̈roduction ALICE decided to proceed by steps.

8.1. Simulation and reconstruction of single events

The typical job of this type is described by a configuration file which, together with the macros to be executed, the shell script starting the execution and the Replica Catalogue configuration file, is sent via the InputSandbox to the RB and, therefore, to the CE's/WN's.

The execution script is rather simple. The first part contains the settings describing the Virtual Organization and the definition of the environment needed by *AliRoot* at run time. After that, the sequence of *AliRoot* macros which actually perform the event simulation and reconstruction is executed. The script checks that all output files have been created and are not empty (no errors during the reconstruction chain), saves them on the Close SE (automatically determined by the *edg-brokerinfo* commands) and registers them on the ALICE Replica Catalogue at NIKHEF, together with some logging information (i.e., job related metadata).

Alice jobs require an automatic proxy renewal (using MyProxy) because of their duration usually exceeding the proxy default validity. The requirements and the custom rank are carefully chosen to ensure an effective occupancy of all the available CPUs at the various sites minimizing, at the same time, the job queueing time. However, in the medium term, we expect the load balancing optimisation to be transparent to the user.

Almost 1000 s̈hortëvents of this type have been submitted and successfully executed before moving on.

8.2. Submission of bunches of central events to different RB's

A script to submit bunches of jobs to the various Resource Brokers keeping JobID's and submission status

information was developed and is now driving the actual production. Its last part manages the generated output, by:

- creating the LFN to be assigned according to a given rule;
- defining the Close SE and/or the MSS;
- storing the file on the Close SE and/or staging it on a MSS;
- registering it on the ALICE RC at NIKHEF.

Up to know, about 430 full events jobs have successfully been executed, with an average completion efficiency of about 40%.

9. Remarks on EDG Services

The production test has been going on for two and a half months, and only 15% of the workplan was completed. That is mainly caused by the system instability and the limited resources allocated to the application testbed. The generation of the first sample (300 events), with data stored on disk SE, was completed in about 45 days, with an average system efficiency of about 56%. The second part, with output storage on Mass Storage Systems, was more painful: only about 150 events were successfully run, with an efficiency of about 35%.

Our present remarks and concerns about the status of EDG services will be outlined in the following. However, it must be clearly stated that the situation is quickly evolving: these remarks should be considered as a sampling in time, just to give an idea of the difficult items; most of the problems they describe will be soon solved, while others will soon come out.

9.1. Software Configuration

The EDG site managers have been very efficient and quick in the installation and configuration of ALICE software. Soon after the announcement of its availability, it was installed on all but one the EDG sites participating to the testbed.

9.2. Job Submission

The present restrictions on the number of simultaneously running jobs should progressively be removed. In fact, even in the case of known limitations, a stable system should eventually enforce the mechanism by automatically rejecting jobs in excess to the VO quota.

An upper limit of 100 jobs/day/RB, for a total of 400 jobs through the 4 available RB's (1 at CERN, 2 at CNAF, 1 at IC) would meet our present requirements; however, it's probably better to set a limit to the number of jobs queued or under execution by a Virtual Organisation at any time.

In view of the expected growth of the system, it is our opinion that the present Workload Management System architecture should be revised, so as to meet the scalability requirement. In particular, the present implementation of the push model does not seem to scale easily, as the WMS services get overloaded very soon.

Another possible option would be the implementation of RB functionality at the level of the Virtual Organisation..

Given the limit on the maximum numer of concurrent jobs, however, the performance of the Job Submission Service is quite satisfactory. In fact, at the beginning of the tests, the main reason of job submission failures was that the number of concurrent jobs per RB was too large. When this was lowered down to 100 jobs per day per RB, all job submissions went well with a reasonable average submission time per job of about 12-15 seconds. However, not all the submitted jobs ran successfully. The main reasons of failure, in decreasing order of importance, were the following:

- instability of WMS services over a time scale smaller than the job duration;
- error in the automatic renewal of the user proxy for long jobs. The subjects of the certificates of some RB's were not correctly included in the *myproxy-server.config* file of the MyProxy server and the current configuration has, a too short (15 minutes) proxy renewal mechanism in place which fails in most cases. An automatic subscription procedure for new RBs installed and configured on the testbed and a test-suite to be executed on the MyProxy Server would be very welcome;
- the disk space on the worker nodes filled up;

The OutputSandboxes of all correctly finished jobs were retrieved without problems.

9.3. Data Management

The ReplicaManager and the Brokerinfo services worked well and more than 3000 files are already registered in the ALICE Replica Catalogue. However, it should be noted that ALICE jobs are not particularly stressing from the point of view of registrations or queries to the Replica Catalogue. The average job duration of several hours, combined with the small number of output files (currently 8 files per job) keeps the number of service requests per day very small.

On the other hand, a problem has been encountered in the use of Replica Manager commands:

due to a misconfiguration of the RC Logical Collection created for the physics production, about 80 LogicalFileNames (LFN's) were registered without the information on the corresponding PhysicalFileNames (PFN's). That happened because file registration in the Replica Catalogue, currently, is not an ätomicp̈rocedure and has no roll-back mechanism in place to ensure error recovering; for the same reason, also de-registration and subsequent correct re-registration of those critical LFN's were not at all straightforward operations and could not be performed via the edg-replica-manager interface.

ALICE believe the present management of the ReplicaCatalogue, which does not allow users to define new Logical Collections, is too restrictive. In view of the increase of the number of users per VO, it should be possible for them to define collections of entries in the RC without any request to the RC manager.

9.4. Mass Storage

ALICE succeeded in storing files both to SE's without a MSS and to SE's with a MSS. Files were also successfully replicated between two sites with a MSS. The availability on the Information Index, for all the sites with a MSS, of a RunTimeEnviroment variable defining whether a site is connected or not to a MSS allows to steer jobs to sites with or without a MSS by simply modifying the JDL file.

10. The AliEn - EDG interface

Since the *AliEn* system is already in place and working (fig. 2), and since its design makes it easy to implement components like a queue system (PBS, for instance) or a mass storage system (CASTOR, as an example), the design philosophy has been to implement the interface in such a way to see EDG computing resources (namely, the resources seen by a given Resource Broker) as a single *AliEn* Computing Element, and the whole EDG storage as a single, large *AliEn* meta-Storage Element. An interface site is thus an EDG User Interface machine running also the *AliEn* Computing Element, Storage Element and ClusterMonitor software suite 3.

It is important to remark that, in order to implement that architecture, outbound connectivity on EDG Worker Nodes is required. Some people object that this feature could generate security problems; in fact, ALICE believe that, once the control on user credentials is enforced strictly enough before granting access to the system, the restrictions on the actions that can be performed by authorised users should be minimised.

The selected architecture is intrinsically hierarchical, and therefore intrinsically scalable: whenever the size of a connected EDG-system reaches a critical value, it is possible to configure it as an isolated subsystem, interfaced to *AliEn* thorugh a dedicated Interface Site.

In the following, the interface is described in more detail, for the different services.

10.1. Job Submission and Control

Job submission from *AliEn* to EDG is straightforward. The Interface Site is seen by the *AliEn* Server just as another Computing Element asking for a job; the *AliEn* JDL is translated, adding some specific requirements (namely, the *AliEn* and ALICE run time environments and outbound connectivity from the Worker Node) and the job is submitted to the EDG RB via the usual EDG tools (fig. 3).

As soon as the job reaches an EDG Worker Node and it starts, it also reports on its status to the *AliEn* Server that takes care of job monitoring. A barebone interface to the *dg-job-status* command is however provided to keep track of jobs before they reach their final destination, by keeping (on the interface site) a database of correspondences between *AliEn JobTokens* and EDG *JobID's*, also useful for reaching the job through EDG (e. g. for killing a running one).

10.2. Data Registration

The EDG storage interface is based on Replica Manager commands. All data files generated by a job running on EDG and stored on EDG resources are registered in both the EDG Replica Catalogue and in *AliEn* Data Catalogue, in order to implement data retrieval from both worlds (and guarantee file survival on EDG Storage Elements). The job, making use of *AliEn* services dynamically started on demand on the Worker Node, registers the file to the EDG Storage Element close to the CE that generates it, by using a LFN generated by the job. That LFN and the RC name (to allow different VOs to use the same interface) are then used to build an intermediate filename, used as a PFN by *AliEn* to register the file in its own Data Catalogue. The uniqueness of file names and consistency between LFN and intermediate filename is thus enforced by *AliEn*; files registered from the *AliEn* command line on the EDG meta-SE will be stored on a default EDG SE defined (by *AliEn*) as c̈loseẗo the Interface site (fig. 3).

10.3. Data Access

The retrieval of PFNs from an EDG site is always a two-step process:

- the job queries the *AliEn* Data Catalogue, sending an LFN, getting back the intermediate FN and translating it to the EDG LFN;

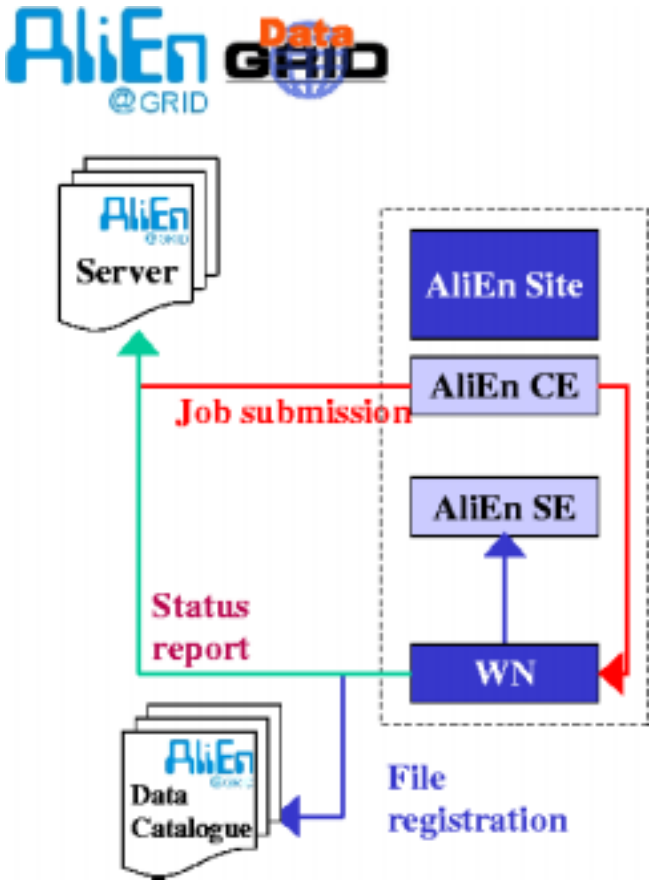

Figure 2: The interaction between the *AliEn* Server, running job scheduling tools and the Data Catalogue and an *AliEn* Client, which pulls up jobs from the Server queue, executes them on the attached CE/WN's and stores the output on a SE, connecting to the Data Catalogue for the registration.

- the EDG LFN is used to query the EDG Replica Catalogue, get the EDG PFN and retrieve the physical file.

In the simplest data access scenario an *AliEn*-submitted job requires *AliEn*-generated data residing on an EDG Storage Element and runs on an EDG ComputeElement. Since the default policy is to send jobs wherever the needed data are stored, most of the jobs requiring EDG-stored data will most likely run on EDG nodes. However, the need may arise for cross-system jobs, for example for the analysis of a composite data sample stored partially on *AliEn* and partially on EDG resources. In such occurrences, the interface site acts as a staging area, exactly in the same way of a StorageElement staging files from a tape vault or some other mass storage device. It is clear that the Interface Site may become a bottleneck, mainly for reconstruction jobs (Monte Carlo productions always write data to the C̈loseS̈torageElement). However, nothing prevents the deployment of many such sites (for example, one per EDG RB), thus effectively sharing the load of data staging and job submission.

10.4. Results and Problems

A prototype interface site is installed in Torino, based on the EDG 1.4.3 and AliEn 1.29 versions. The first ALICE simulation job on EDG via *AliEn* job submission was recently run: the job was sent to *AliEn* with the requirement to run on the ComputeElement identifying the Interface Node. The job was picked up and sent to the EDG ResourceBroker for submission to the selected (or required) CE. At the simulation completion, the output was stored on the EDG StorageElement and registered both in the EDG and in the *AliEn* (Replica)Catalogue. The physical output was then retrieved on the local machine via the *AliEn* command line.

To our knowledge, that is the first achievement of full interoperability between different GRID Systems concerning job submission, data registration and data access; it is also a fundamental milestone for our work program, aiming at the full exploitation of both ALICE owned and common computing ang storage resources.

However, some possible sources of problems have been identified:

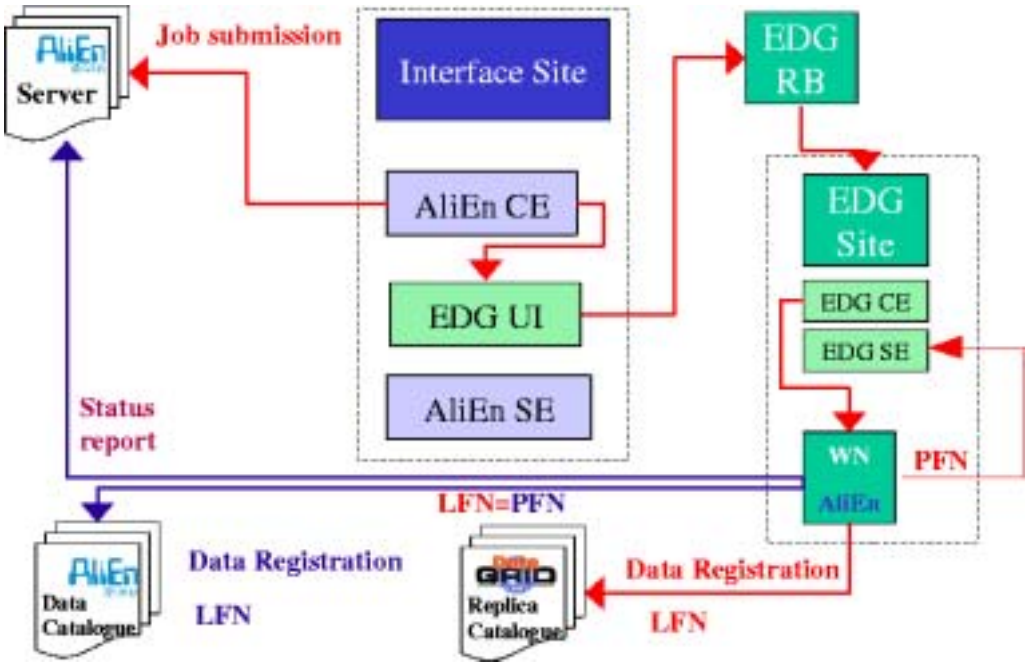

Figure 3: The scheme of the *AliEn*-EDG interface architecture. A job, configured on the Server queue, is picked-up by the *AliEn* CE at the interface site, its description is adapted to EDG and then the job is sent to the EDG Resource Broker. When it starts at the EDG WN, it also begins reporting about its status to the *AliEn* Server. After the processing algorithm is completed, output and eventually input files are stored on the "Close" EDG Storage Element registered both in the EDG Replica Catalogue and in the *AliEn* Data Catalogue.

- many things can go wrong before the job reaches a worker node and reports to the *AliEn* server. It is difficult to foresee all the possible failures, particularly since EDG commands do not always respond consistently when problems occur; it is not unusual to see core dumps, for example, from which it is awkward to recover. It is thus possible, in the present situation, that some job get lost: for this reason, after a given time, *AliEn* deletes these jobs and re-schedules them. The solution is rough but, provided the fraction of EDG failures is small enough, effective.

- as already mentioned, the Interface Site may become a bottleneck due to the need to stage files from EDG to *AliEn* and vice-versa for jobs running on one system needing data from the other, but it is foreseeable that multiple interfaces can reduce the load on each of them. The number of jobs on any interface node will depend on the availability of resources on the EDG side and in any case can be set to a predefined maximum value.

- outbound connectivity from the worker node is required by the job to communicate with the server. This is not presently a problem, since most of the EDG testbed Worker Nodes have public IP addresses, but this situation may well not be the final one.

11. Analysing EDG events from AliEn

Our present short term goal is the analysis of simulated and reconstructed events produced on EDG. That is not possible with the presently available EDG functionality and must therefore by carried on using *AliEn*.

That requires, as a preliminary step, the registration all the content of the EDG Replica Catalogue in the *AliEn* Data Catalogue, which was performed via a dedicated script.

Presently, the setting up of the analysis algorithm is ongoing and our first results will soon be available.

12. Plans

ALICE plan to include the LCG testbed in their production environment, as soon as it will be available: all the resources will be driven by *AliEn*, thanks to its interface with EDG services. The full integration of the *AliEn* managed resources with the LCG testbed will allow to maximise the amount of available resources and to use them for the deployment of higher level use cases ("random" analysis, use of metadata).

The use of the EDG Application Testbed will continue until the availability of the LCG testbed, to com-

plete the physics test production and to develop analysis scripts.

Acknowledgments

The efficient and quick feedback by the EDG application testbed support team wasgreatly appreciated.

References

[1] http://root.cern.ch
[2] http:/alien.cern.ch
[3] http://alice-grid.ca.infn.it/edg