CREAM Computing Element: a status update

Paolo Andreetto\textsuperscript{a}, Sara Bertocco\textsuperscript{a}, Fabio Capannini\textsuperscript{b}, Marco Cecchi\textsuperscript{b},
Alvise Dorigo\textsuperscript{a}, Eric Frizziero\textsuperscript{a}, Alessio Gianelle\textsuperscript{a}, Massimo
Mezzadri\textsuperscript{c}, Salvatore Monforte\textsuperscript{d}, Francesco Prelz\textsuperscript{c} David Rebatto\textsuperscript{c},
Massimo Sgaravatto\textsuperscript{a} and Luigi Zangrando\textsuperscript{a}

\textsuperscript{a} INFN Padova, Via Marzolo 8, I-35131 Padova, Italy
\textsuperscript{b} INFN CNAF, Viale Berti Pichat 6/2, I-40127 Bologna, Italy
\textsuperscript{c} INFN Milano, Via Celoria 16, I-20133 Milano, Italy
\textsuperscript{d} INFN Catania, Via Via Santa Sofia, 64, I-95123 Catania, Italy

E-mail: paolo.andreetto@pd.infn.it, sara.bertocco@pd.infn.it,
fabio.capannini@cnaf.infn.it, marco.cecci@cnaf.infn.it,
alvise.dorigo@pd.infn.it, eric.frizziero@pd.infn.it, alessio.gianelle@pd.infn.it,
massimo.mezzadri@mi.infn.it, salvatore.monforte@ct.infn.it,
francesco.prelz@mi.infn.it, david.rebatto@mi.infn.it,
massimo.sgaravatto@pd.infn.it, luigi.zangrando@pd.infn.it

Abstract. The European Middleware Initiative (EMI) project aims to deliver a consolidated
set of middleware products based on the four major middleware providers in Europe -
ARC, dCache, gLite and UNICORE. The CREAM (Computing Resource Execution And
Management) Service, a service for job management operation at the Computing Element (CE)
level, is a software product which is part of the EMI middleware distribution. In this paper
we discuss about some new functionality in the CREAM CE introduced with the first EMI
major release (EMI-1, codename Kebnekaise). The integration with the Argus authorization
service is one of these implementations: the use of a unique authorization system, besides
simplifying the overall management, allows also to avoid inconsistent authorization decisions.
An improved support for complex deployment scenarios (e.g. for sites having multiple CE
head nodes and/or having heterogeneous resources) is another new achievement. The improved
support for resource allocation in a multi-core environment, and the initial support of version
2.0 of the Glue specification for resource publication are other new functionalities introduced
with the first EMI release.

1. Introduction
The CREAM (Computing Resource Execution And Management) Service [1] is a Grid
Computing Element (CE) implementation and part of the gLite middleware stack.

It allows the submission of computational jobs to the resources available on a site (managed
by a batch system, such as LSF or PBS) and provides the other usual job management
functionalities: job cancellation, job status monitoring, etc.

The interface with the Local Resource Management System (LRMS) is fully implemented
and hidden to CREAM by the BLAH (Batch Local Ascii Helper) [2] component. BLAH is also
used, through its BLParser component, to notify CREAM about job status changes in the batch
system.
Both serial and parallel jobs are supported: a Condor classified advertisements (classads) [3] based language called Job Description Language (JDL) [4] is used by end users to describe job characteristics and requirements.

CREAM is implemented as an extension of the Java-Axis servlet, running inside the Apache Tomcat container. It provides a Web Service based interface.

Users can interact directly with the CREAM service. For this purpose a CREAM Command Line Interface (CLI) written in C++ is provided. Being CREAM a Web Service, users can also easily implement their own clients, if needed, using a suitable Web Service framework.

CREAM is also used through higher level job management services: in particular the gLite WMS (Workload Management System) [5] service and Condor-G [6] both support the submission to CREAM based CEs.

CREAM is currently deployed in several production Grid infrastructures. In particular it is widely deployed in the WLCG (Worldwide LHC Computing Grid) Grid infrastructure, where it obsoleted the Globus based LCG-CE. CREAM is also heavily deployed in the EGI (European Grid Infrastructure) [7] Grid (at the time of writing this article, a top BDII (Berkeley Database Information Index) of the EGI infrastructure publishes information about 400 CREAM CE head nodes, spread in more than 230 sites) where it is used by several user communities for different applications.

2. CREAM in the EMI project

At the time of writing, the CREAM CE software service is being maintained, supported and evolved in the context of the European Middleware Initiative (EMI) project [8].

The EMI project is a cooperation among the major European middleware stacks: ARC, gLite, UNICORE and dCache. Its purpose is to consolidate, harmonize and support the original software platforms, evolve and extend them based on existing and new requirements. Goals of the project are also the standardization and development of common interfaces, to achieve interoperability between different Grid infrastructures in different areas (such as authentication, authorization, job submission, accounting, resource information exchange).

EMI is currently the major software provider for the EGI and PRACE [9] Grid infrastructures.

EMI releases are time-based. Major releases are delivered once per year, offering a good balance between the conflicting requirements of stability and innovation.

The first EMI release (EMI-1, codename Kebnekaise) was released on May 2011, while at the time of writing this paper the second EMI major release (EMI-2, codename Matterhorn) is being released. The third and final EMI release (EMI-3, codename Monte Bianco) is scheduled for the end of February 2013.

Besides several bug fixes, many new functionalities were introduced in the CREAM CE service released in EMI-1. The most relevant ones are:

- Integration with the ARGUS authorization service.
- Easier deployment for complex environments, with multiple CE head-nodes and/or heterogeneous sets of resources.
- Preliminary support for publication of resource information in Glue 2.0 format.
- Support for execution on multi-core environments.
- Improved batch system support: support for Grid Engine was implemented (support for LSF, Torque/PBS and partly Condor was already available).

The rest of this paper focuses on some of these new implementations. Other on-going or planned developments are instead discussed in another paper presented at this conference [10].
3. Integration with the ARGUS authorization service

In the CREAM CE implementation available before the EMI project, authorization was managed through gJAF (Grid Java Authorization Framework).

gJAF provides a way to invoke a chain of policy engines and get a decision result about the authorization of a user. The policy engines are divided into two types, depending on their functionality. They can be plugged into the framework in order to form a chain of policy engines as selected by the administrator in order to let him set up a complete authorization system. A policy engine may be either a PIP (Policy Information Point) or a PDP (Policy Decision Point). A PIP collects and verify assertions and capabilities associated with the user, checking her role, group and VO (Virtual Organization) attributes. A PDP may use the information retrieved by a PIP to decide whether the user is allowed to perform the requested action, whether further evaluation is needed, or whether the evaluation should be interrupted and the user access denied.

Local user mapping was instead managed via gLExec [11], a program to make the required mapping between the Grid world and the Unix notion of users and groups. gLExec in turn makes use of the LCAS (Local Centre Authorization Service) and LCMAPS (Local Credential Mapping Service) software components.

The CREAM CE runs also a gridFTP server, used for different purposes (in particular to support client data push for input data and to allow the publication of the installed software runtime environments). Authorization for this gridFTP server is also implemented through LCAS and LCMAPS, but using different configuration files with respect to the ones used by gLExec.

The use of multiple authorization systems is clearly an issue: because of misconfigurations, inconsistent authorization decisions can happen (e.g. a security component could grant authorization for a certain Grid user, while another one could deny the operation). The use of multiple components implementing the same or similar functionalities is also a concern in terms of operations of the system.

All these problems have been addressed in the CREAM CE released with EMI-1, where the integration with the ARGUS authorization service [12] was implemented.

ARGUS is a system meant to render consistent authorization decisions for distributed services. Through the Policy Administration Point (PAP) component it is possible to publish and maintain consistent authorization policies. Published policies are then evaluated in a consistent manner by the Policy Decision Point (PDP) component. Finally the Policy Enforcement Point (PEP) component of ARGUS is used to request authorization and enforce decisions. Fig. 1 shows the interactions among these components.

![Figure 1. ARGUS components and their interactions.](image-url)
In the ARGUS integration scenario achieved with the EMI-1 release, ARGUS is the only component used within the CREAM CE. Besides implementing the functionality previously done by gJAF (i.e. allowing or denying a certain operation issued by a Grid user) it also manages credential mapping. Also the gridFTP server on the CREAM CE is integrated with ARGUS.

So in the scenario where CREAM is integrated with ARGUS, gJAF, gLEexec, LCAS and LCMAPS are therefore not needed anymore. Besides simplifying the overall management, inconsistent authorization decisions can not happen anymore, since a single authorization system is used.

4. Deployment in complex environments

Small Grid sites using CREAM to access the available computational resources using a Grid interface have usually a quite simple scenario: a small set of worker nodes managed by a local resource managed system, and accessible through a single CREAM CE head node.

This is not the case for large sites, where it is quite common to deploy multiple CREAM CE head nodes referring to the same set of worker nodes. This is done to spread the load between multiple service instances and to assure continuous availability of the computing service.

In such environments each CREAM CE would publish, through its Resource BDII, the information about all the physical resources available in the site. Besides the problem of needlessly publishing the same information multiple times, this would overcount the installed capacity at the site. The workaround is to publish zero values for the installed capacity from all but one of the nodes, but this is clearly far from being an ideal solution.

This issue was addressed in the first EMI major release by the gLite-CLUSTER Grid node. Basically gLite-CLUSTER allows the configuration of information related to the physical resources to be separated from the configuration concerning the job submission interface. gLite-CLUSTER is a Grid node type responsible to publish information related to the computational resources available in the site (which in Glue 1 are modelled through the Cluster and SubCluster objectclasses). This information is therefore published just once. The gLite-CLUSTER node can then be referenced by all the compute elements available in that site and using those resources: these computing elements will just publish information relevant for job submission, i.e. modelled by the Glue 1 GlueCE and GlueVOView objectclasses.

gLite-CLUSTER allows also to address another issue related to sites having heterogeneous hardware. In this case usually separated batch queues are defined for each hardware configuration (e.g. low/high memory queues), and then attaching the corresponding GlueCE objects to the relevant Cluster/SubCluster pairs. However the YAIM configuration tool (used to configure CREAM Computing Elements) allows configuring only a single cluster per CREAM head node.

This problem was solved with gLite-CLUSTER, which allows the publication of several clusters, each one composed by any number of subclusters.

The following deployment models are therefore possible for a CREAM-CE:

- CREAM-CE can be configured without worrying about the gLite-CLUSTER node. This can be useful for small sites with a very simple setup. In this case CREAM-CE will publish a single cluster/subcluster. This is called no cluster mode.
- CREAM-CE can work on cluster mode using the gLite-CLUSTER node type. This should be considered if in the site there are multiple CE head nodes, and/or if in the site there are multiple disjoint sets of worker nodes, each set having sufficiently homogeneous properties.

It must be noted that the gLite-CLUSTER node can be deployed together with a CREAM CE node, or in a different host.
5. Publishing in Glue 2.0 format
The CREAM CE has been publishing information related to physical resources and information relevant for job submission since the very beginning. This information is published in Glue 1 format in the LDAP based Resource BDII running on each CREAM CE, and it is then usually propagated to higher level BDII (site-BDII, top-BDII). Both static and dynamic (e.g., number of running jobs, number of available job slots, etc.) information is provided. The former is provided by LDIF files that are created at configuration time. The latter is instead provided by dynamic GIP (Grid Information Provider) plugins, scripts that gather and publish fresh information, in particular batch system related.

One of the goal of the EMI project is the support for the Glue 2.0 format [13] in all the services. This was implemented also for CREAM: a preliminary incomplete support for Glue 2.0 publishing was introduced in the CREAM CE in the first EMI major release; it was then finalized in the EMI-2 release.

Therefore, besides publishing in Glue 1 format, in the Resource BDII information is also available in Glue 2.0. The following Glue 2 object classes are published:

- ComputingService
- ComputingEndPoint with the relevant AccessPolicies
- ComputingManager
- ComputingShares (which correspond to GlueVOViews in Glue 1.3) along with the MappingPolicies
- ExecutionEnvironments (one per Subscluster)
- Benchmarks
- ToStorageServices
- ApplicationEnvironments (one per installed software)
- EndPoint for the ApplicationPublisher (the service publishing information about the installed software)
- EndPoint for the CEMon service (published only if CEMon is deployed).

As shown in fig. 2, support for no-cluster mode and cluster mode are both provided. In the first case all the objectclasses are published by the Resource BDII running on the CREAM CE node. If instead a CREAM CE is configured in cluster mode, the Resource BDII running on the CREAM CE publishes just the information relevant for job submission, i.e. the ComputingEndpoint objectclass (with the relevant AccessPolicies) and the EndPoint objectclass for CEMon (if CEMon is deployed). All the other objectclasses are published by the Resource BDII running on the gLite-CLUSTER node.

More information about the Glue 2.0 support in the CREAM CE is available at [14].

6. Support for multi-cores environments
The Grid projects where CREAM was considered were initially focused on the HEP (High Energy Physics) community, where typically only serial programs are considered. Other user communities were added later: chemistry, biology, medical imaging, etc. In these communities the use of parallel programs is quite common and therefore the need for MPI support emerged.

This was implemented in CREAM allowing multiple CPUs to be requested, and then using the MPI-start package [15] to hide some of the complexity of the MPI setup from users.

The MPI user community then made a new request: how to get the allocated cores to be all on the same physical machine, or packed into as few physical machines as possible.
Figure 2. Glue 2.0 publication in no-cluster and cluster modes.

This was implemented in the first EMI release, where first of all users can specify how the cores can be distributed for the allocation: a new JDL attribute (\textit{SMPGranularity}) determines the number of cores any host involved in the allocation has to dedicate to the application.

Users can also specify the total number of required nodes: this is done using an ad-hoc JDL attribute called \textit{HostNumber}.

For multi-threaded applications, instead of letting users figure out how to restrain their programs to single cores, it is more convenient to allow users to reserve whole nodes. This is possible in the CREAM released with EMI-1 through another boolean JDL attribute (\textit{WholeNodes = True}).

It must be noted that recently the requirement for multi-core support has emerged also in the HEP community. Traditionally, WLCG experiments have been running their workloads using one core per job. However, WLCG experiments have for some time required the possibility to run their jobs on more than one core. The general observation is that the aggressive parameters foreseen for the LHC in 2012 will push pileup much higher than in 2011 and this may lead to issues related to memory consumption which can be mitigated if memory is shared across several cores on worker nodes.

7. Conclusions
In this paper we described the main new functionalities in the CREAM Computing Element achieved in the first EMI major release.
More detailed information can be found on the CREAM web page [16].

Acknowledgments
The research results of the EMI project are co-funded by the European Commission under the FP7 Collaborative Projects Grant Agreement Nr. 261611.

References
[1] C. Aiftimiei et al, Design and Implementation of the gLite CREAM Job Management Service, Future Generation Computer Systems, Volume 26, Issue 4, April 2010, pp. 654-667, doi: 10.1016/j.future.2009.12.006.
[2] E. Molinari et al, A local batch system abstraction layer for global use, Proc. of XV International Conference on Computing in High Energy and Nuclear Physics (CHEP’06), Feb 13–17 2006, Mumbai, India.
[3] R. Raman, Matchmaking Frameworks for Distributed Resource Management, Ph.D. thesis, University of Wisconsin-Madison, 2001.
[4] CREAM Job Description Language Attributes Specification, available online at https://wiki.italiangrid.it/twiki/bin/view/CREAM/JdlGuide.
[5] P. Andreetto et al., Evolution of Grid meta-scheduling towards the EMI era, in Proc. of CHEP’10, 18th International Conference on Computing in High Energy and Nuclear Physics, 18-22 October 2010, Taipei (Taiwan)
[6] J. Frey, T. Tunnenbaum, M. Livny, I. Foster, S. Tuecke, Condor-G: A Computation Management Agent for Mulled-Institutional Grids Proc. of of HPDC 2001.
[7] European Grid Infrastructure (EGI) Website, http://www.egi.eu
[8] European Middleware Initiative (EMI) project Website, http://www.eu-emi.eu/home
[9] PRACE project Website, http://www.prace-project.eu
[10] P. Andreetto et al., New developments in the CREAM Computing Element, to appear in Proc. of International Conference on Computing in High Energy and Nuclear Physics (CHEP’12), May 21–25 2012, New York, United States.
[11] D. Groep, O. Koeroo and G Venekamp, gLEexec: gluing grid computing to the Unix world, Journal of Physics: Conference Series, volume 119, number 6, year 2008
[12] ARGUS Website, https://twiki.cern.ch/twiki/bin/view/EGEE/AuthorizationFramework
[13] S. Andreozzi, S. Burke, F. Ehm, L. Field, G. Galang, B. Konya, M. Litmaath, P. Millar, JP Navarro, GLUE Specification v. 2.0, available online at http://www.ogf.org/documents/GFD.147.pdf
[14] Glue 2.0 support in the CREAM CE, https://wiki.italiangrid.it/twiki/bin/view/CREAM/CreamGlue2
[15] MPI-Start Website, http://grid.ifca.es/wiki/Middleware/MpiStart
[16] CREAM Website, https://wiki.italiangrid.it/CREAM