Workload management in the EMI project

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}, Aristotelis Kretsis\textsuperscript{e}, Massimo Mezzadri\textsuperscript{c}, Salvatore Monforte\textsuperscript{d}, Francesca 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 Santa Sofia, 64, I-95123 Catania, Italy
\textsuperscript{e} Research Academic Computer and Technology Institute (RACTI), Patras, Greece

Abstract. The EU-funded project EMI, now at its second year, aims at providing a unified, high quality middleware distribution for e-Science communities. Several aspects about workload management over diverse distributed computing environments are being challenged by the EMI roadmap: enabling seamless access to both HTC and HPC computing services, implementing a commonly agreed framework for the execution of parallel computations and supporting interoperability models between Grids and Clouds. Besides, a rigorous requirements collection process, involving the WLCG and various NGIs across Europe, assures that the EMI stack is always committed to serving actual needs. With this background, the gLite Workload Management System (WMS), the meta-scheduler service delivered by EMI, is augmenting its functionality and scheduling models according to the aforementioned project roadmap and the numerous requirements collected over the first project year. This paper is about present and future work of the EMI WMS, reporting on design changes, implementation choices and long-term vision.

1. Introduction

The European scientific research has greatly benefited from the increasing availability of computing and data infrastructures, now with unprecedented capabilities, for what concerns large scale distributed initiatives. After the necessary initial period of research and consolidation, that has taken place in the past several years, the growing usage of these resources now requires the transformation of the computing infrastructures into a professionally managed and standardized set of services. It is of strategic importance for the establishment of permanent, sustainable research infrastructures to lower the technological barriers and prevent resource owners from federating their resources. The European Middleware Initiative (EMI) \cite{1} project aims at delivering a consolidated set of middleware services based on the four major middleware providers in Europe: ARC \cite{2}, dCache \cite{3}, gLite \cite{4} and UNICORE \cite{5}. In EMI, the Workload Management System (WMS) is the reference Grid meta-scheduler. The WMS represents a stable, scalable and decentralised service that implements early-binding submission to several kinds of Computing Element, CREAM \cite{6}, OSG CE \cite{7}, A-REX \cite{8} and LCG-CE \cite{9}. In particular, the submission to CREAM has been optimised by implementing a scalable synchronisation mechanism to query for status and consequently take decisions. The EMI project
is contributing to the evolution of the WMS towards a sustainable service, that relies on a common framework composed of the EMI consolidated APIs.

2. Present situation
The gLite Workload Management System (WMS) provides a service responsible for the distribution and management of tasks across resources available on a Grid. These tasks, which basically consist in execution requests, are usually referred to as “jobs”. Access to other paradigms, such as Cloud or Volunteer Computing are being evaluated at the time of writing; they do not represent a major obstacle and basically require writing a connector, or adapting an existing one, that is able to map WMS jobs to the specific execution requests typical of each interface. The WMS comes with a well known and rich set of features, see [10], implemented with the so called 'early-binding’ model. This model mandates that a job is bound to a resource as soon as possible and, once the decision has been taken, the job is passed to the selected resource(s) for execution, where, very likely, it will end up in some queue. At the other extreme, late-binding foresees that the job is held by the WMS until a resource becomes available, at which point that resource is matched against the submitted jobs and the job that fits best is passed to the resource.

Besides only implementing early-binding submission, the WMS cannot even considered to be a full-featured VO framework, that is, a central system utilised by the VO to coordinate generation of input, workload management and data distribution for all its users. Other than mere workload management, in fact, VO frameworks were born to manage high-level workflows from their users and have direct control on how they are translated into grid jobs in terms of expected dimension of input and output, expected completion time, locality of data, etc. Frameworks are able to both use early or late binding as the underlying submission layer, even if the tendency is to use tools based on late-binding. Future evolutions of the WMS will eventually include both the ability to act as a VO framework and to implement late-binding through a standard protocol, something that will be described more in detail later in the paper. Nonetheless, even with the present design, there are still several use-cases that make the WMS usable in HEP. The Service Availability Monitoring implemented at CERN [11], for example, uses the WMS at the core of its submission engine. CMS analysis [12] is using the WMS in instances deployed at the italian INFN/CNAF and CERN in load balancing. Again, other minor HEP VOs, such as TheoPhys [13], are interested in using the WMS for its capabilities of handling MPI [14] jobs through recent extensions of its JDL that allow to pass configuration hints on to the batch system well in advance the job is run. In general, the WMS can be utilised as a submission system that abstracts low-level submission on behalf of VO frameworks. For what concerns the submission to the CREAM CE, in particular, the WMS is able to provide an efficient and scalable service through its internal component called ICE, responsible for monitoring the state of submitted jobs and for taking the appropriate actions when job status changes are detected (e.g. to trigger a possible resubmission upon detection of a failure) in an optimised and well tested way. For better interoperability with the Open Science Grid, the WMS will be able to submit to GRAM5 based CEs starting from the next EMI releases.

Here is a summarized view of some of the functionality provided by WMS, in parentheses the first release where the functionality was made available is indicated:

- Submission to LCG-2 CE, OSG CE through GRAM2, ARC CE through GRAM2 and EMI-ES (EMI-3)
- Scalable submission to CREAM through ICE (EMI-1)
- Support for submission and monitoring for the LCG, gLite and CREAM CEs (EMI-1)
- Support for Data management interfaces, such as DLI and StorageIndex (gLite)
- Support for RFC proxies (EMI-1)
3. Evolution towards sustainability
The Grid consists of a highly decentralized and distributed system glued together by certain agreements (implemented by common standards, protocols, services and so on). In early times, the design of workload management services was fundamentally meant to reflect this concept with a static approach, so as to rely on a series of global and coordinated Grid Services. This model included globally controlled systems for information publishing and retrieval, distributed policies for authentication and authorization at sites, global job tracking systems and so on. Over time, this design did not reveal to properly deal with the speed at which information varies in a Grid. This eventually led to the implementation of so called VO and pilot frameworks, centralised system where requests for computations from VO users are gathered, distributed and managed in a single entry point through late-binding. These systems are characterised by lack of dependencies on centralized systems and opportunistic scheduling.

The original design of the WMS, expecting to depend on services to measure dynamic information such as the gLite BDII [15], seems to be a little bit out-dated in this respect. The recent development driven by the EMI EU-project are making of the WMS both a more sustainable service, in terms of dependencies on a consolidated set of Grid APIs, and a more autonomous service, able to select the most suitable resources only according to a global service registry, such as the EMI Service Registry, and its own statistics, accessible through the extensibility of its JDL, on previous history [16].

One of the most important goals of the EMI project is achieving a unified and consolidated middleware distribution. In the past years, the grid computing services now part of the EMI project were separately implemented, mainly in the field of High Energy Physics, in the context of three different middleware stacks: ARC, gLite and UNICORE. To be able to converge to such a harmonized distribution, much of the effort during the first and second year was devoted to providing a common ground, through use of same protocols and agreements, in such a way that both end users and services are able to seamlessly interoperate throughout both existing and new Distributed Computing Infrastructures (DCI). The focus on the third year will basically move to consolidation, further work on the agreements previously taken and fulfilling new requirements from the user community that have come in the meanwhile.

The main points are as follows: a considerable amount of work during the second year was devoted to the implementation of the EMI Execution Service. The EMI Execution Service (EMI-ES) deals with the submission and management of computational tasks on a computing resource and covers important use cases from both high-throughput and high-performance computing.
Worth of mention is the ability to compute multi-scale workflows on both High Performance Computing (HPC) and High Throughput Computing (HTC) environments, described by the COMPCHEM community, that will be detailed in a separate paragraph below. The EMI-ES has always represented a significant part of the EMI vision, its fully interoperable and production-quality implementation become now of primary importance. Acknowledgments by official standardization bodies would also be welcome and can surely be pursued during the third project year, even if not being directly a primary goal. Once this interface is finalized, the harmonization process of the existing clients will finally take place, built on top of the EMI-ES, so as to achieve uniform and high-level access to computing resources without reducing the present functionality required by each DCI. That would add another reason for a VO framework to adopt the WMS as a unified submission service: other than GRAM and CREAM legacy interface, the WMS will be in fact also able to submit to resources exposing an EMI-ES interface.

As anticipated, the interaction with the common EMI service registry [18] will remarkably allow for easier interoperability, cooperation and uniformity in operation of DCIs.

Following is a list of development items that the WMS is working on in the EMI project.

- Submission to the EMI Execution Service [20]
- Integration with the EMI Service Registry (EMIR) and ability to rely only on static information provided by the Information System
- Integration with the EMI Common Authentication Library [21]
- Integration with the EMI Authorisation Service [22]
- Support for GLUE schema version 2.0 [23]
- Support for multi-many core jobs embedded in the JDL

4. A newly developed testing framework
An automated test-suite has been developed in order to automatically test a set of basic functionality in the WMS and in the WMS Command-Line Interface (WMS-CLI). The test-suite, developed in python, is divided into three packages. The first package (WMS-cli) is composed of some python methods, one for each command available in the WMS-cli (glite-wms-job-`). The purpose of the tests implemented by this package is to check every single command using all the possible combinations of meaningful options and checking if the service reacts as expected. The second package (WMS-service) is composed of some general python methods which check the most critical features of the WMS service. The third package (regression-tests) contains methods that perform individual regression tests of bugs which should have already been fixed in the WMS code.

Each python script contains a number of individual test cases. Each test run ends with a specific status, either success or fail. When a test case fails for any reasons, the test does not terminate, but continues to the next available test case. During the test execution a working directory is created which contains, among the others, three files useful for debugging. The first one is the WMSService-TS-test_id.log (for WMS-service package) or WMSCLI-TS-test_id.log (for WMS-cli package) which contains general debug information for the whole execution of the test. In this file all the useful information for each test case is logged, like test number and description, jobids, status of jobs, commands run and their status and output. The log level is configured by the user and can assume one of the following values WARNING, INFO and DEBUG (see below). Another file useful for debugging is errors.log : it contains information related to the errors which have occurred during the execution of the test, like the test number and its description, the failed operation and the failure reason. A third file (traceback-errors.log) contains traceback information for each failure. When a test case fails, the working directory is not cleaned, so that the user can inspect the debug files in order to determine the failure reason.
As the tests involve typical grid operations, like submitting a job to the WMS and waiting for its completion, re-submitting a job, cancelling a job, use delegation, perform proxy renewal operations and so on, a valid proxy is required in order to run the testsuite. There are three ways to set a valid proxy for the test suite:

- create a valid proxy before executing the test suite (i.e. the default location pointed to by the environment variable X509_USER_PROXY is considered)
- let the test-suite create the proxy by passing the passphrase to it in an interactive mode
- set a default key without password

Some tests (like delegation and proxy renewal tests) require to create limited proxies, so they only work with the second option. The configuration parameters that can be passed to the testsuite are divided into two categories: required and optional. The configuration parameters can be defined in a configuration file or passed as options to the command-line. The configuration file can contain both required and optional parameters, while the command-line can accept only required parameters.

5. Bridging HPC and HTC altogether

Computational researchers of several scientific communities make heavy usage of the resources of the accessible computing facilities. The advent of High Performance Computing (HPC) networks and large grids of off-the-shelf High Throughput Computing (HTC) elements can provide these scientists with increasing processing power. HPC and HTC platforms, however, pursue different approaches to massive computing. Namely:

- HPC networks (like PRACE [25] in Europe and TeraGrid [24] in the United States) offer a suitable distributed infrastructure for tightly coupled calculations requiring large memory sizes, MPI libraries, a high speed interconnection network, large and high throughput storage devices. The interconnection of the different supercomputers is mainly intended in this case for facilitating the job management and offering unified resources (like storage).
- HTC grids (like that of the European Grid Infrastructure EGI [26]) offer a highly cost-effective computational platform exploiting the concurrent elaboration of a huge number of small-middle sized computers most often of the rackable CPU cluster type. The typical job exploiting the advantage of this infrastructure consists of a huge amount of substantially uncoupled computational tasks that can be distributed on independent CPUs.

The HPC and HTC computing paradigms have developed separately, and sometimes even conflictingly, to enable two totally different approaches to concurrent computing, and, as a consequence, they are also managed by middleware quite different in nature. As a matter of fact, they target different classes of applications, numerical algorithms and computational methods, so resulting in different middleware designs. In many scientific fields, in fact, researchers would greatly benefit from combining the two HPC and HTC paradigms altogether, so as to enable the accurate modeling of real-like systems and virtual reality simulations based on multi-scale and multi-physics approaches. Despite that, little effort has been spent, up to date, to promote a seamless integration of the two infrastructures. COMPCHEM, in particular, is engaged in designing and implementing accurate realistic multiscale Molecular and Materials applications like the Grid Empowered Molecular Simulator (GEMS) [27], involving coordinated utilisation of HPC and HTC resources. This will allow to overcome the shortcomings of the present somewhat unsatisfactory situation whereby none of the available resources is suited for complex requests, in terms of the diversity of requested computations, such as the ones required by GEMS. What is lacking, in fact, is the coordination of different systems, such as HPC and HTC, to interoperate via a single workflow management system that properly isolates into
known patterns the user workflow and dispatches such building blocks so as to send each of them to the most suitable resource. This would allow a more efficient execution of complex applications. The implementation of submission through EMI-ES in the WMS, together with the common APIs developed in EMI for authentication, delegation and autorisation, will allow seamless submission to HPC and HTC, including resources from UNICORE and NorduGrid [28].

6. Future developments
The gLite WMS is designed and implemented to provide a dependable, robust and reliable service for efficient distribution and management of end-user requests for computation, storage, network, instruments and whatever resource may be shared across a production quality Grid. It comes with a fully-fledged set of added-value features to enhance low-level job submission. Thanks to the flexibility of a scalable, fault-tolerant and service-oriented architecture it has been deployed in a number of layouts and scenarios.

After several years of operation the WMS has reached sustained stability and a performance targeted at covering the current needs, coming in particular way from High Energy Physics and Bioinformatics. The ability to implement the functionality implemented by a so called VO framework, acting as a distributed batch system, together with the ability to provide a standard and exensible protocol for the submission of pilot jobs are being investigated at the time of writing. This will allow to make of the WMS a fully fledged workload management system for general purpose VO’s, relying on a common and consolidated set of Grid APIs, such as the ones being developed by the EU project EMI, comprising the four major Grid middleware providers in Europe. All the different views on security that each present VO framework holds about how delegation and proxy handling should be performed (without entering too much into details) will also be definitely sorted out by the utilisation of a well-defined and middleware service globally devoted to this task.

References
[1] http://www.eu-emi.eu/
[2] www.nordugrid.org/arc
[3] www.dcache.org
[4] http://en.wikipedia.org/wiki/GRID
[5] http://www.unicore.eu/
[6] Andreetto P, Borgia S A, Dorigo A, Gianelle A, Marzolla M, Mordacchini M, Sgaravatto M, Zanrando L et. al., CREAM: a simple, Grid-accessible, job management system for local computational resources, Computing in High Energy and Nuclear Physics (CHEP’06), T.I.F.R. Mumbai, India, February 13-17, 2006.
[7] https://www.opensciencegrid.org
[8] http://www.nordugrid.org/documents/arex_tech_doc.pdf
[9] https://twiki.cern.ch/twiki/bin/view/EGEE/LcgCE
[10] Cecchi M et al., The gLite Workload Management System, proceedings of GPC 2009 conference
[11] P. M. Rodrigues De Sousa Andrade, Service Availability Monitoring framework based on commodity software, Codispoti G, Mattia C, Fanfani A, Fanzago F, Farina F, Kavka C, Lacaprara S, Miccio V, Spiga D, Vaandering E, CRAB: A CMS Application for Distributed Analysis, IEEE Transactions on Nuclear Science 56:2850-2858 (2009)
[12] http://www.fis.unipr.it/dokuwiki/doku.php?id=grid:mpi-theophys
[13] http://en.wikipedia.org/wiki/Message_Passing_Interface
[14] http://glite.cern.ch/glite-BDII
[15] Cecchi M et al., Design and Evaluation in a Real Use-case of Closed-loop Scheduling Algorithms for the gLite Workload Management System,
[16] JDL Attributes Specification, https://edms.cern.ch/document/590869/1, EGEE-JRA1-TEC-590869-JDL-Attributes-v0-4
[17] Memon S, Field L, Marton I, Szegiti G, Discovering Infrastructure Services with EMI Registry (EMIR),
[19] Dvorak F, Kouril D, Krenek A, Matyska L, Mulac M, Pospisil J, Ruda M, Salvet Z, Sitera J, Skrabal J, Voci M, et. al., *Services for Tracking and Archival of Grid Job Information*, CGW05, Cracow - Poland, November 20 - 23, 2005 proceedings of CHEP 2012
[20] https://twiki.cern.ch/twiki/bin/view/EMI/EmiExecutionService
[21] https://twiki.cern.ch/twiki/bin/view/EMI/EmiJra1T4SecurityCommonAuthNLib
[22] https://twiki.cern.ch/twiki/bin/view/EGEE/AuthorizationFramework
[23] http://www.ogf.org/documents/GFD.147.pdf
[24] http://en.wikipedia.org/wiki/TeraGrid
[25] http://www.prace-project.eu/
[26] www.egi.eu
[27] Costantini A, Gervasi O, Manzalì C, Faginas Lago N, Rampino S, Laganà A, COMPCHEM: progress towards GEMS a Grid Empowered Molecular Simulator and beyond, *Journal of Grid Computing* 8(4), 571-586 (2010)
[28] http://www.nordugrid.org/