Deliverable 3.3

MAPPER Profile

Project acronym: MAPPER

Project full title: Multiscale Applications on European e-Infrastructures.

Grant agreement no.: 261507

Due-Date:	31 March 2012
Delivery:	30 March 2012
Lead Partner:	LMU
Dissemination Level:	Public
Status:	Final
Approved:	
Version:	1.0

DOCUMENT INFO

1.3.2012	M. Schiffers	initial
13.3.2012	Bartosz Bosak, Mariusz Mamonski, Eryk Ciepiela, Derek Groen, Michael Schiffers	Draft version for internal review
30.3.2012	M. Schiffers and internal reviewers	Final version

TABLE OF CONTENTS

Executive Summary

This document defines the MAPPER Basic Profile. It consists of a set of requirements, policies and standards that any single resource provider must implement in order to claim to be “MAPPER compliant”. The requirements were derived from two pilot MAPPER scenarios:

1. Loosely Coupled Application Scenario,
2. Tightly Coupled Application Scenario.

Both scenarios have in common the co-allocation of heterogeneous resources hosted by various providers. However, from a single service provider point of view the two use cases can be reduced to a simpler scenario: the MAPPER Multiscale Base Case (MBC). This document describes all the necessities that the service provider must fulfill in order to support the Multiscale Base Case directly, and thus, indirectly the Loosely Coupled and Tightly Coupled Application Scenarios. The last section of this deliverable is devoted to the lower level standard Distributed Resource Management Application API (DRMAA) [7][8]. In particular, how it is exploited by MAPPER components and how other technology providers can facilitate it in order to offer services compliant with the MAPPER profile.

1 Introduction

MAPPER, grant number 261507, is an infrastructure project for the development and deployment of multiscale applications on European e-infrastructures. As there is no doubt that multiscale applications will be of interest to the broader scientific community, it is of basic interest for the community to derive a common understanding of how a particular set of standards specifications may be composed in order to solve a basic use case in the context of multiscale applications. The single use case described in this document is the MAPPER Multiscale Base Case (MBC). MBC serves as a building block for more advanced multiscale application scenarios.

The MAPPER Profile consists of references to mandatory specifications, along with any clarifications of their contents, restrictions on their use, and references to any normative extensions to them. While it is envisioned that many systems will have capabilities above and beyond those described in this profile, this profile describes a *basic* set of capabilities that can be used as the core of interoperability testing between systems claiming MAPPER compliance.

The document is structured along particular aspects of a MAPPER Profile compliant system. The first aspect covers MAPPER job descriptions, the second deals with data staging, the third relates to security aspects.

It is worth noting that the MAPPER Profile is focused on describing the basic capabilities that must be supported by a compliant system. In many cases, the systems in question will support higher levels of functionality than described here, and many systems will support various extensions to the functionality described in the referenced specifications. It is not the goal of this profile to prohibit the use of such extensions, but to define a set of capabilities that can provide a basis for interoperability. As such, this profile may implicitly allow the use of various constructs, but not make any statement about the semantics of such use, and thus these constructs should not be used as the basis of any interoperability testing of MAPPER Profile compliant systems

2 MAPPER Multiscale Base Case (MBC)

The simplest MAPPER use case is a single submodel of a multiscale application running as a job on a single Grid Computing element (which may be a high performance computing (HPC) system). What distinguishes MBC from a pure vanilla batch job submission use case is that:

- the job must be submitted into an advance reservation context (a requirement of the MAPPER co-allocation protocol),
- the system must support parallel jobs as most of the multiscale applications require HPC resources.

The users may belong to several autonomous organizations but belonging to the same Virtual Organization (VO), an inherent Grid requirement.

2.1 Background

A typical multiscale application consists of (see the detailed descriptions in [1]):

- software modules simulating certain phenomena in certain time or space scale (scaleful). Usually these modules are computationally intensive and would require HPC resources. They are often (but not always) implemented as parallel programs.
- software modules that convert data from one scaleful module to another. Usually these modules do not have demanding computational requirements. However, to avoid communication overhead, they often require to be executed "close" to the scaleful modules they are connecting (they could even be implemented in the same process as the one for the scaleful modules).

The communication structures of multiscale applications may vary significantly between:

- a master-worker paradigm, e.g., a macro scale module (master) triggers the micro scale simulation of a part of its domain that requires more detailed attention. This type usually requires dynamic triggering of module execution. The number of module instances is usually dynamic.
- a peer to peer type of computation where all modules are executed concurrently and exchange data in a usually asynchronous fashion. During the course of their execution, applications often pass many synchronization points (the number can be static or dynamic). Therefore, this type often requires mechanisms for efficient communication.
- a pipe type communication where modules execute one after another.
- a hybrid communication with combinations of two or more types mentioned above. The initial condition module is connected to the rest of the simulation via "pipe", and then the rest of the simulation consists of modules that run concurrently.

Regarding the type of module executions one can distinguish between

- *stateless* modules: after they finish, they return results (in a form of returned state parameters or snapshot files) and they do not preserve any data from their computation;
- and *stateful* modules: after (often partial) computations they remember the state of calculations.

Multiscale applications can also be classified as *loosely* or *tightly* coupled. In loosely coupled simulations there is no loop in the coupling topology (this could be a pipe or direct acyclic graph scheme) and the modules are stateless. In tightly coupled simulations, there is a loop in the coupling topology and the modules are stateful. The two basic coupling types were covered by the two real application use-cases selected by the MAPPER consortium for the first year demonstration.

The next subsections summarize shortly the architecture of both the loosely coupled and the tightly coupled scenarios, as well as the communication patterns between selected components. For more details please consult the MAPPER deliverable D5.2 (Vertical Integration Plan) [2].

2.2 Interactions between MAPPER and Service Providers

The interactions between MAPPER components (or services) and services provided by respective Service Providers depend on the scenario specifics. A detailed plan for the scenarios is presented in [2]. The sequence diagrams in the following figures describe the

Loosely-coupled Application Scenario (Figure 1) and the Tightly-coupled Application Scenario (Figure 2).

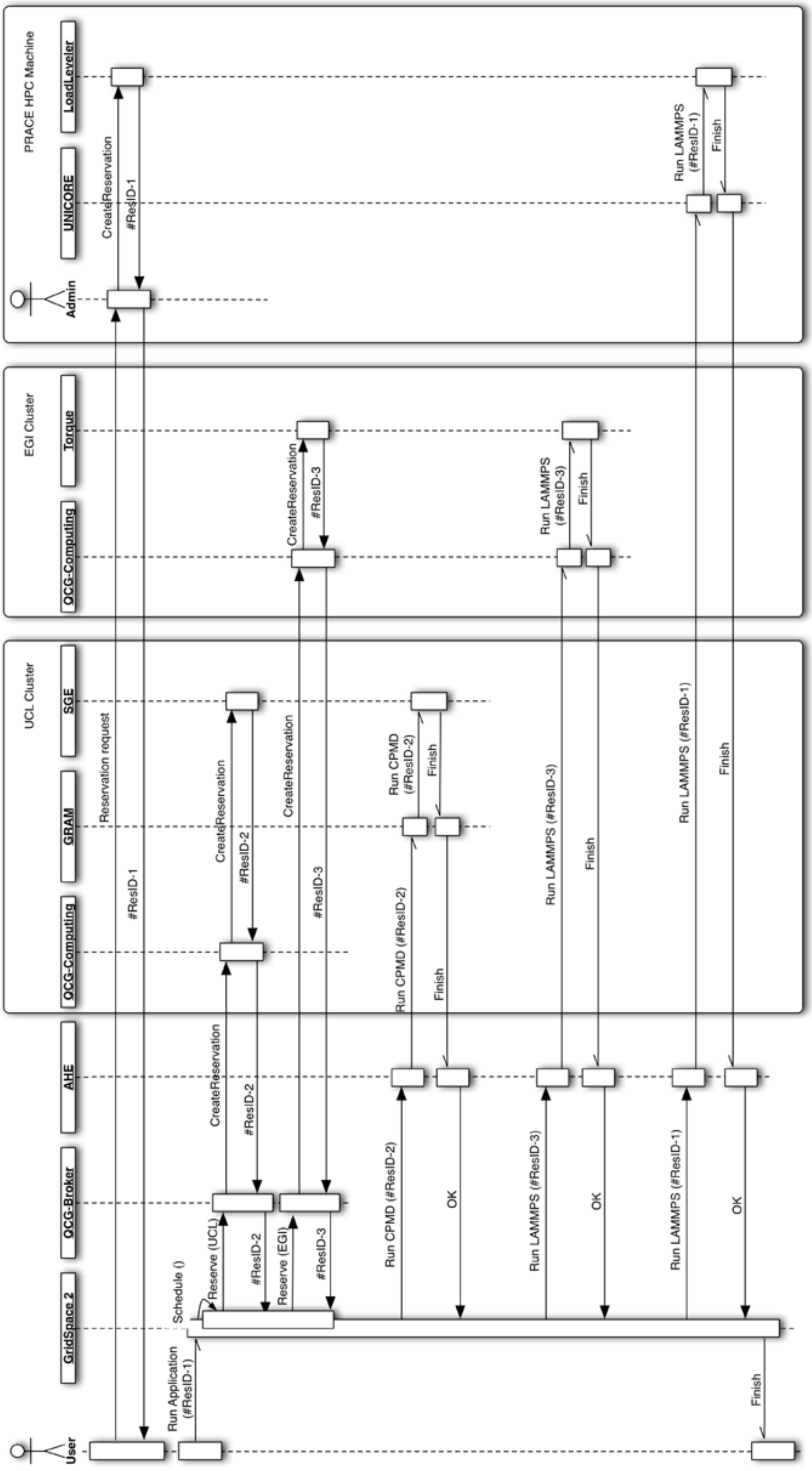

Figure 1: Loosely Coupled Application Scenario [2]

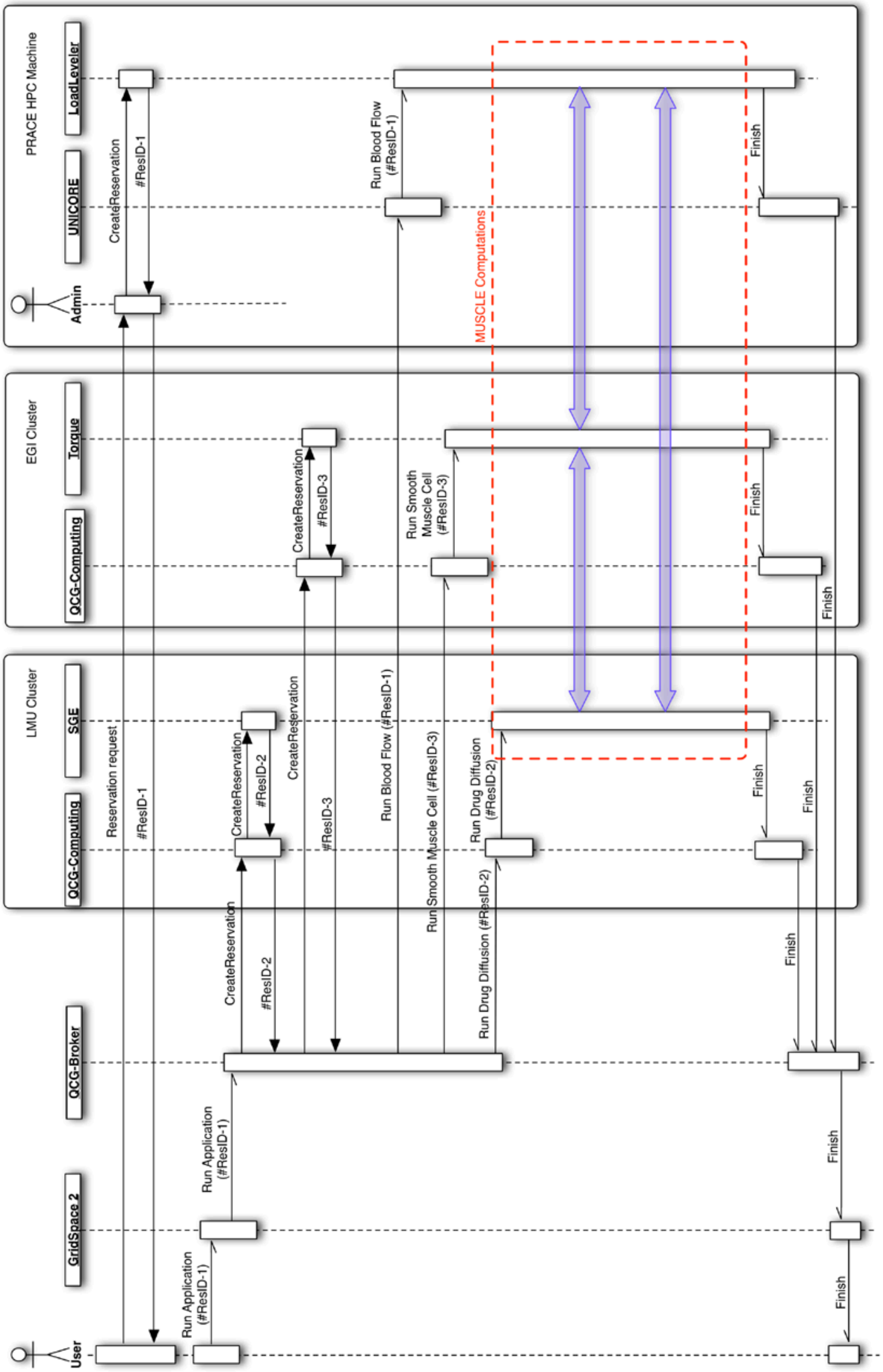

Figure 2: Tightly Coupled Application Scenario [2]

2.3 MAPPER Resource Descriptions

As we assume that applications are preinstalled, and often optimized, by the user for the target site we assume also that he or she knows the quantitative and qualitative properties of the system resources (e.g., by means of Service Provider documentation). Therefore, the resource descriptions of the target sites are beyond the scope of this profile.

2.4 MAPPER Fault Tolerance Model

If a job fails due to system problems then it must be resubmitted by the client (as a new job) as the job scheduler will not automatically rerun the job. Moreover, if persistent storage fails then all job information – past and present – is assumed lost.

2.5 Out-of-Band Aspects

Program provisioning is also considered to be out-of-band. That is, programs are assumed to be pre-installed – either directly on the system components or on something like a distributed file system that is accessible from the Grid nodes. The program absolute path must be registered so the other services can use abstract application names.

Creation and management of user security credentials are considered to be out-of-band. Users are assumed to possess the necessary credentials needed to submit both data and work to the Grid components. Note that this covers, among other scenarios, systems where all jobs are run using “anonymous” credentials and users must explicitly stage any private data they wish to use to common “pool” storage facilities.

The Grid's job scheduler is reachable at a well-known communications endpoint. Thus, there is no need for directory services beyond something like DNS. Management of the system resources (i.e. compute nodes) and services of the compute cluster is out-of-band and indeed opaque to users of the cluster. This is not to say that there isn’t a standard means of managing a computing Grid element; just that system management is not part of this MAPPER use case.

Also how the users acquire resources allocation (so called computational grants) is out of band for this specification.

2.6 Out-of-Scope Considerations

Which scheduling policies a scheduler supports is out-of-scope. Similarly, concepts such as quotas and other forms of SLAs are out-of-scope. From the point of view of a single Resource Provider jobs are completely independent of each other. The notion of dependencies among jobs is handled by higher level MAPPER components and are thus out-of-scope for this document.

A job consists of a single program running on a single Grid element, whether it is a sequential program, a single node application parallelized using OpenMP, or a multi-node MPI job. The process of creation of reservation is currently out-of-scope of the MAPPER Base Case. It might be a call to the system supporting advance reservation interface (e.g. QCG-Computing[1]) or a manual process involving human interactions with the system administrator (e.g., via mail or phone call – a classic out of bound channel). Here we only assume that reservation identifiers are known to the system in advance.

3 MAPPER Requirements

This section lists both the normative and the non-normative requirements that a single Resource Provider must conform to in order to support the MAPPER Base Case scenario. It is also clearly stated whether the particular requirement is mandatory or optional.

3.1 Basic Execution Service Requirements

The service provider must provide access to the hosted resources over the OGSA Basic Execution Service Version 1.0 compliant interface [5]. Moreover, the interface must also adhere to the HPC Basic Profile, Version 1.0 specification [4]. The MAPPER profile requires that the service implements at least the *BES-Factory* port type (i.e., the implementation of the *BES-Managament* and *BES-Activity* port types is optional). The *BES-Factory* port must, at least, implement the three (of total five) following operations:

- *CreateActivity* – an operation that creates a new activity (i.e. submit a job),
- *GetActivitiesStatuses* – an operation that provides the status of the previously submitted activity,
- *TerminateActivities* – an operation that requests cancellation of the previously submitted activity. It is left to the implementation whether this operation has synchronous or asynchronous semantics.

The service must be accessible over an HTTPS or HTTPG (HTTP over GSI [10]) endpoint.

As an optional extension of the base case the service provider may support:

- notifications of activity statuses over a WS-Notifications [11],
- advance reservation management web service interface.

[1] http://www.qoscosgrid.org/trac/qcg-computing

Unfortunately there is currently no normative extension of the BES interface for advance reservation management, however any technology provider wishing to provide such interface may derive from existing works [9].

3.2 Job Description Requirements

The *CreateActivity* operation of the *BES-Factory* port accepts activities described using the Job Submission Description Language Version (JSDL) 1.0 standard [6]. Here we provide a list of mandatory JSDL elements that must be *accepted* by the conformant system. By *accepted* we mean not only parsing the element but also applying all semantics as described in the JSDL specification, together with all the clarifications given in the HPC Basic Profile and in this specification. We distinguish normative JSDL elements from non-normative vendor extensions. List of normative JSDL document elements that must be accepted by the target system are listed below:

- *ApplicationName* – an abstract name of the application (e.g. MUSCLE) that is mapped to the physical absolute path by the underlying system,
- *JobIdentification/JobName* – an opaque name of the job,
- *HPCProfileApplication/Argument* – a vector of arguments passed directly to application,
- *HPCProfileApplication/Input* – a name of the application standard input file,
- *HPCProfileApplication/Output* – a name of the application standard output file,
- *HPCProfileApplication/Error* – a name of the application standard error file
- *HPCProfileApplication/Environment* – a vector of the key value pairs that denote environment variables that must be set before starting the application
- *HPCProfileApplication/WorkingDirectory* – an absolute path of the working directory where the application should be started,
- *Resources/TotalCPUCount* – a number of job slots (cores) allocated for the job,
- *Resources/IndividualCPUCount* – a number of job slots (cores) allocated for the job on single node.

Moreover, we request one additional non-normative extension of the JSDL document: the target system must be capable of submitting job into an advance reservation (how the reservation is created is out-of-scope for this document). Therefore, the system must accept, probably as a vendor extension of the *Resource* element, a local reservation identifier in the job description document.

3.3 Data Staging Requirements

Although we assume that the application is preinstalled (see Section 2.5) there is still a need for transferring input and output data of the application. For this purpose the profile requires that the gridFTP [12] service is available at the target site. We also require that the service has access to the file system that is available from cluster worker nodes, and thus accessible directly within a job.

3.4 Security Requirements

As both the Tightly and Loosely coupled scenarios involve additional services (e.g., the QCG-Broker or AHE) between the user and the infrastructure it implies that the the target site must accept delegated credentials. Therefore in the MAPPER profile we request that the target service must accept X.509 proxy certificates [13].

Moreover, for the convenience of the user, the target site should accept certificates issued by any of the Certificate Authorities that are member of EUGridPMA[2]. This prevents users from the necessity of presenting different credentials while accessing different resources.

3.5 Other requirements specific to the Tightly Coupled Application Scenario

The last, non-normative, requirement is related only the Tighly Coupled Application Scenario as it involves parallel computations spanning across multiple sites. The cross-cluster communication is facilitated with the help of the additional user-space daemon deployed on the frontend machine under the assumption that the sites follow the following firewall policy:

- there exists one incoming open port on the frontend/interactive node (the port must be accessible from the other clusters involved in cross-cluster simulation),
- the connectivity from worker nodes to the frontend/interactive node is not restricted.

4 How To Claim Profile Conformance

Claims of conformance to the MAPPER Base Case Profile can be made using the mechanisms described in WS-I Conformance Claim Attachment Mechanisms [14], when the

[2] www.eugridpma.org/

applicable profile requirements as described previously have been met. The conformance claim URI for the MBC Profile will be published as an update to this document. In any case, a claim of conformance must be made for the WS-I basic profile [15] and a successful pass of the MAPPER Test Suite [16] must be produced..

5 Distributed Resource Management Application API

5.1 DRMAA 1.0

The Distributed Resource Management Application API 1.0 is an Open Grid Forum standard for portable submission and management of batch jobs into Distributed Resource Management (DRM) systems [7], [8]. It offers a high-level interface, modelled after POSIX *fork/wait* routines, for anyone who needs portable interfaces for batch systems. Over the years DRMAA 1.0 gained many implementations. The interface is currently offered by the following systems: Grid Engine, Condor, Torque/PBS Pro, GridWay, Xgrid, Platform LSF, UNICORE, Kerrighed Cluster Framework, IBM Tivoli Workload Scheduler LoadLeveler and SLURM. The core of most of the implementations is written in the C language. However the DRMAA community developed also many bindings for high-level languages like Java, Python, PERL and Ruby. Finally, in 2007, the DRMAA 1.0 specification received the “full recommendation” status in the Open Grid Forum organization, thus confirming the maturity of the standard.

5.2 DRMAA 2.0

Despite the great success of the DRMAA 1.0 specification, the scope of 1.0 version of the standard might be seen too narrow for current HPC needs, especially if compared to modern batch systems. For this reason, a few years after releasing the DRMAA 1.0 standard, the DRMAA Working Group started works on a new version of the standard. All those efforts were crowned when the DRMAA 2.0 specification was officially released in February 2012. The DRMAA 2.0 API, comparing to the initial standard, offers much more features, namely: restartable sessions, parallel jobs support, machines monitoring capabilities and advance reservation management. All this, and even more, is available over coherent and extensible interfaces.

The MAPPER project was one of the contributors of the new DRMAA specification, as the unified access to the advance reservation mechanism could significantly simplify the implementation of the aforementioned Loosely Coupled and Tightly Coupled Application Scenario.

5.2.1 QCG-Computing

The QCG-Computing service for the job submission, control and management operations relies only on the DRMAA API interfaces. This approach distinguished QCG-Computing from similar services (e.g., the Globus GRAM) which usually use scripts that interact with batch systems via command line tools (comparing to native C APIs exploited by most of the DRMAA implementations). In the design phase of the QCG-Computing it was crucial to map all the JSDL elements, which are marked as mandatory by the HPC Basic Profile, to the corresponding DRMAA 1.0 job template attributes. The mapping used by the QCG-Computing service is presented in the table below:

JSDL element name	DRMAA 1.0 attribute name
JobName	DRMAA_JOB_NAME
Executable	DRMAA_REMOTE_COMMAND
Argument	DRMAA_V_ARGV
Environment	DRMAA_V_ENV
WorkingDirectory	DRMAA_WD
Input	DRMAA_INPUT_PATH
Output	DRMAA_OUTPUT_PATH
Error	DRMAA_ERROR_PATH

Values of the other JSDL elements (e.g., *TotalCpuCount*) are passed to the underlying batch system via the DRMAA_NATIVE_SPECIFICATION Job Template attribute. The translation to the native options is handled in the batch system specific fashion, by so-called JSDL Filter module (e.g., *pbs_jsdl_filter* for the Torque/PBS Pro systems).

It is planned to move from DRMAA 1.0 to the DRMAA 2.0 interface as soon as the first implementation of the new version of the standard would be publicly available. With DRMAA 2.0 the QCG-Computing service could map more JSDL attributes to the job templates attributes (e.g. *TotalCpuCount*) and exploit a new portable Advance Reservation API instead of native interfaces.

5.2.2 GridSpace

GridSpace relies on the DRMAA specification in two ways. Indirectly, by using the QCGExecutor to submit and manage jobs, and directly when using the SSHExecutor to connect to the sites directly. The former is explained in detail in the previous section. In the latter case, the SSHExecutor runs a generic DRMAA client on the head node of a site in order to run a job on the local resource management system. DRMAA conceals the batch system specifics and allows for using their capabilities in a uniform way. Thus, the same scripts can be used on the UCL Grid site *Mavrino* and on the Polish Grid site *Zeus*, despite the fact they run the Sun Grid Engine and the PBS Torque systems, respectively.

For tools like GridSpace DRMAA constitutes a good abstraction layer over local resource management systems. Making GridSpace relying on the DRMAA standard reduces vendor-specific code and defines integration points for existing and potential new local resource management system providers.

6 Intellectual Property Statement

MAPPER takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the MAPPER Project Office.

MAPPER invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to practice the recommendations expressed in this document. Please address the respective information to the MAPPER Project Office.

7 Disclaimer

This document and the information contained herein is provided on an “As Is” basis and MAPPER disclaims all warranties, express or implied, including but not limited to any warranty that the use of the information herein will not infringe any rights or any implied warranties of merchantability or fitness for a particular purpose.

8 References

[1] MAPPER Deliverable D8.1: Description of the Architectures and Interfaces; http://www.mapper-project.eu/documents/10155/23479/D8.1+-+Architecture+and+Interfaces.pdf, 2011

[2] MAPPER Deliverable D5.2: MAPPER vertical integration plan; http://www.mapper-project.eu/documents/10155/23586/D5.2-VerticalIntegrationPlan.pdf, 2011

[3] Robert Chumbley, Jacques Durand, Micah Hainline, Gilbert Pilz, Tom Rutt: Basic Profile Version 1.2, http://ws-i.org/Profiles/BasicProfile-1.2-2010-11-09.html, 2010

[4] Blair Dillaway, Marty Humphrey, Chris Smith, Marvin Theimer, Glenn Wasson: HPC Basic Profile, Version 1.0, http://www.ogf.org/documents/GFD.114.pdf, 2007

[5] I. Foster, A. Grimshaw, P. Lane, W. Lee,M. Morgan, S. Newhouse, S. Pickles, D. Pulsipher, C. Smith, M. Theimer: OGSA Basic Execution Service, Version 1.0, http://www.ogf.org/documents/GFD.108.pdf, 2008

[6] Fred Brisard, Michel Drescher, Donal Fellows, An Ly, Stephen McGough, Darren Pulsipher, Andreas Savva: Job Submission Description Language (JSDL) Specification, Version 1.0, http://www.gridforum.org/documents/GFD.56.pdf, 2005

[7] Hrabri Rajic, Roger Brobst, Waiman Chan, Fritz Ferstl, Jeff Gardiner, John P., Andreas Haas, Bill Nitzberg, Daniel Templeton, John Tollefsrud, Peter Tröger: Distributed Resource Management Application API Specification, Version 1.0, http://www.ogf.org/documents/GFD.133.pdf, 2008

[8] Tröger P., Brost R., Gruber D., Mamoński M., Templeton D: Distributed Resource Management Application API Specification, Version 2.0, http://www.ogf.org/documents/GFD.194.pdf, 2012

[9] Mamoński M.: Smoa Computing HPC Basic Profile Adoption – Experience Report, http://www.ogf.org/documents/GFD.179.pdf, 2011

[10] A. Konstantinov: The HTTP(S,G) and SOAP Server/Framework – Code and Usage Description, http://www.nordugrid.org/documents/HTTP_SOAP.pdf, 2009

[11] Steve Graham, David Hull, Bryan Murray: Web Services Base Notification 1.3, OASIS Standard, http://docs.oasis-open.org/wsn/wsn-ws_base_notification-1.3-spec-os.pdf, 2006

[12] I. Mandrichenko, W. Allcock, T. Perelmutov: GridFTP v2 Protocol Description, Report GFD-R-P.047, http://www.ogf.org/documents/GFD.47.pdf, 2005

[13] S. Tuecke, V. Welch, D. Engert, L. Pearlman, M. Thompson: Internet X.509 Public Key Infrastructure (PKI) Proxy Certificate Profile, RFC 3820, http://www.ietf.org/rfc/rfc3820.txt, 2004

[14] Mark Nottingham, Claus von Riegen: Conformance Claim Attachment Mechanisms Version 1.0, http://www.ws-i.org/Profiles/ConformanceClaims-1.0-2004-11-15.html, 2004

[15] Keith Ballinger, David Ehnebuske, Martin Gudgin, Mark Nottingham, Prasad Yendluri: WS-I Basic Profile Version 1.0, http://www.ws-i.org/profiles/BasicProfile-1.0-2004-04-16.html, 2004

[16] MAPPER Deliverable D3.4: Description of the MAPPER Test Suite; 2012

9 Abbreviations

AHE	Application Hosting Environment
DRMAA	Distributed Resource Management Application API
HPC	High Performance Computing
JSDL	Job Submission Description Language
MBC	Multiscale Base Case
MPI	Message Passing Interface
PBS	Portable Batch System
QCG	QosCosGrid
SLA	Service Level Agreement
VO	Virtual Organization