1. EARTH SCIENCES RESEARCH APPLICATION REQUIREMENTS AND RECOMMENDATIONS

The basis for describing the applications requirements is the joint list of use cases of the AWG (the DataGrid Applications Working Group), which combines the common requirements of HEP, Biosciences and Earth Observations applications. All requirements and use cases described there are valid for the Earth Science community. This section should be considered in addition to the AWG joint list of use cases [R6] and represents the main Earth Science recommendations.

The following section presents a discussion of the main ES recommendations arising from ES Grid application deployment experiences so far. This is followed by a set of tables summarizing the recommendations discussed.

1.1. Discussion of ES research requirements

1.1.1. VO MANAGEMENT

A VO database contains information about available VOs, groups within the VOs and the users belonging to the VOs.

In addition to defining one or more groups within a VO, it must be possible to define user roles. The VO database must allow the definition of named roles in terms of available VO groups.

It must be possible to assign to a user one or more groups and/or one or more roles.

A VO must have at least one group.

A member of a VO must belong to at least one group or have at least one role.

The VO database must be operated via an administrator interface available over the Internet using a simple web pages interface. The administrator interface must authenticate administration sessions using standard grid security mechanisms (i.e. via the user's proxy certificate).

The VO administrator interface must allow the definition of system administrators and VO administrators. Only system administrators can create or destroy VOs. One or more VO administrators must be assigned for each VO that is created. When adding a new VO to the database the system administrator must specify the details of the VO administrator(s). The administrator interface must ensure a backup (i.e. deputy) person is specified for each system and VO administrator.

A VO database service must provide information about VOs, groups, roles and members to clients (i.e. local grid security services) that request it. The service must be remotely accessible via standard web protocols and must use standard grid security mechanisms to carry out authentication and authorization of service requests. In order to ensure timely propagation of VO changes to the sites, clients which use the database information service must be able to register with the service to receive automatic notifications about VO database updates.

1.1.2. SECURITY

A clear and straightforward procedure must be established for a user to get a valid key pair signed by a CA. A well-established CA infrastructure with clear and verifiable procedures is vital for security.

A single, uniform, standard security model must be developed, applied and used by all grid components that need to implement security.

Security must be seamlessly integrated into all components that need it as a fundamental design consideration right from the start.

Individual grid components should not design or develop individual security solutions but should make use of the standard security components provided by EGEE.

The security solution must be fully compatible, conceptually and functionally consistent with the proxy certificate model used by GSI.

It must be possible, having fulfilled any prevailing security considerations, for the user to belong to any number and combination of VO(s)/group(s)/role(s).

A grid user should only need to possess a single certificate, independently of the VO(s)/group(s)/role(s) he/she is associated with.

The user certificate should only be used for the purposes of identifying the user and as a consequence there should be no specific information about VO(s), group(s) and role(s) of the user associated with the certificate.

It must be possible for the user to carry out the grid single sign on (proxy generation) procedure directly from the users normal machine, whether a desktop, workstation or laptop, whether running under Linux or Windows, without requiring the user to log on to an intermediate machine, and the user must never be required to store his/her private key on a remote machine (e.g. grid User Interface machine, etc.).

Any VO(s)/group(s)/role(s) authorization information, as may be required for a particular task in hand, should be specified by the user at the time the user performs grid "login" or single sign on, during the process of obtaining a valid proxy certificate.

During the single sign on process, the user must specify the required VO/group/role attributes to be associated to the proxy certificate that is created. A process running on the user's machine should contact each of the relevant VO database information services to gather the required authorization information as requested by the user. It must be possible to attach any number and combination of VO/group/role attributes to the proxy certificate, as long as any specific security constraints that may apply (e.g. conflicts between requested roles) can be resolved.

From the users point of view, the security mechanisms should appear as non intrusive, transparent and functionally and procedurally the same, regardless of the type of grid resources being accessed (e.g. resource broker, application executables, application environment, replica catalogue, replicated data, metadata, information system, storage resources).

A standard, uniform security access procedure should be applied whenever a proxy delegated process attempts to carry out an operation on a grid resource, regardless of the type of process and the type of resource. First, the authorized identity (or identities) delegated to the remote process is established by recovering the authorization attributes embedded in the proxy certificate. Next, the local permissions associated with the resource are checked. The local permissions consist of a table of recognized identities (e.g. VO/group/role) and the corresponding permitted operations (read/write/execute). The operation will only proceed if a positive match is obtained between the operation(s) requested and permissions granted to one of the authorized identities, as delegated via proxy certificate attributes to the requesting process.

Whenever the user process running on the grid creates any persistent resources or information, such as data on a storage element, or replica metadata information in a replica catalogue, etc. the user's default or currently set security attributes (e.g. read/write/execute permissions), or as delegated via proxy to the requesting process, shall be attached to the entity that is created. All subsequent accesses to that entity will be checked by the grid security middleware, using the attached permissions. It must be possible for the user to subsequently modify the assigned permissions later on, at any time.

1.1.3. DATA MANAGEMENT

The grid middleware must provide applications with both command-line commands and library embedded APIs to carry out all required data management functions. Where appropriate, (e.g. replica manager commands, information system commands), the same middleware must be available and must function the same way, whether running on a worked node or on a User Interface (i.e. job submission) machine.

All data management operations must follow standard security procedures as defined in the section on security requirements above. For instance, creating replicas, file transfers, inserting, updating and

retrieving replica catalogue information and metadata, must include proper security checks on the proxy-delegated permissions.

Users of replicated data must be able to have 100% confidence in the validity and integrity of the replicated data (it must not be necessary for the application to expressly check the integrity and validity of replicated data every time it is used).

Data replication must be available as an atomic operation, consisting of data transfer and catalogue metadata registration. Data transfer must not be started until the middleware has correctly verified that the proceeding registration can be successfully performed (i.e. the replica catalogue service is running and the user has the required permissions, etc.). Integrity of replicated data must be automatically verified by the data transfer procedure and subsequent access to the replicated data must be preceded by an automatic data integrity check, which can only be skipped by an explicit override option. Replica registration must not be attempted until data transfer has been successfully completed and data integrity has been verified.

The replica and data management systems must not expose any internal replica details to end users and applications. For instance, any internal ids, such as globally unique IDs, that are assigned to replicated data and used internally by the middleware to manage files, must be hidden from the user or application, unless the details are specifically requested. The application must be able to refer to replicated data at all times by the normal application/user defined logical filenames; it must not be mandatory to refer to data (whether logical file names or physical file names) using a different id that was assigned automatically by the middleware, which is probably going to be meaningless and devoid of any semantic application information. From the application/user point of view, details such as the globally unique ID are considered internal to the middleware and should be handled transparently.

Replica catalogue performance must be constant and must not be susceptible to degradation, either due to the number of entries in the catalogue or the request loading on the replica cataloguing service. Quality of service, in terms of performance and overheads (i.e. wall clock time consumed) associated with the insertion (registration), search and retrieval of replica information in the replica catalogue, must be quantified as absolute minimum, mean and maximum, guaranteed values. These QoS values must be published by the service in the grid information system and easily retrieved by the end user/application.

Both the replica catalogue, as well as the storage resource services, must be capable of handling (storing, searching, updating, retrieving) file numbers of the order of millions of replicas.

It must be possible to operate any number of independent replica and storage resource services per VO, for instance, each VO group may operate, if necessary, one or more individual replica catalogues and storage services.

An application must be able to create, populate, utilize and destroy replica catalogues on the fly, as required by the application, without the need for human administrator intervention, having performed the appropriate security and resource utilization quota checks.

Applications must be informed in advance when resource utilization (e.g. number of files, storage space) approaches the allowed limits imposed by the system. Actually reaching imposed limits must be handled in a controlled, predictable and organized way; the user/application must be properly informed. Both replica catalogues and storage resources must provide ways to inform users/applications about levels of utilization (e.g. percentage available, used).

The replica catalogue must offer mechanisms to associate user/application defined metadata values with individual replicated data files. It must be possible to define key/value tuples. All the standard, most frequently used data types must be supported, e.g. date, time, polygon, integer, alphanumeric, float. When inserting/updating replica information it must be possible to assign both the associated set of metadata keys, as well as the logical filename, by a single operation (i.e. it must not be necessary to call the API or issue the command line for each associated metadata key/value pair). The application must be able to refer to replicated file(s) either by using assigned logical names or using metadata key/value tuples. Given a logical file name, it must be possible to recover the full list of associated

key/value pairs in a single operation on the API or command line. Given a metadata search consisting of a set of target key/value pairs (e.g. as a SQL query), it must be possible to recover the complete list of logical file names that satisfy the query. It must be possible to specify metadata key values to be searched on as conditional expressions.

It should be possible for users to change ownership and/or access rights to files on the Grid, comparable to 'chmod' and 'chown' commands on Unix systems.

1.1.4. WORKLOAD MANAGEMENT

The main function required of the workload management system is automation of resource discovery, job submission (including staging of executables and auxiliary data from the submitting machine to the execution platform), monitoring of job status throughout the submission, and automated retrieval of job result files created on the execution platform, as specified by the job description. The possibility to cancel a submitted job and the possibility to communicate with the job via standard input and output channels are also necessary.

The workload management system includes a resource broker service that carries out automated matching of job requirements, as specified by the user/application, against available resources published in the grid information system. The resource broker uses JDL language to specify job requirements, including replicated data.

Often a job executing on the worker node needs to retrieve certain job parameter information that was specified by the application when composing the JDL script. It must be possible for jobs running on the worker nodes to retrieve parameters that were specified in the JDL, for instance, the full list of logical file names specified using the inputData attribute. This avoids the application job pre scheduler environment having to specify job parameter information twice, e.g. once in the JDL and then again for use by the programme to be executed on the worker node environment.

Since the grid system is by nature highly dynamic, and users usually take some time and experience to understand how to make best use out of the middleware, it will often not be possible to successfully carry out user or application requests, for a wide variety of reasons. Some failures may be due to error conditions and others may be may be due to conflicts in the formulation of JDL scripts, while still others may be due to service outages, malfunctions or misconfigurations. Whichever the case, components of the workload management system (indeed, the same applies most of the grid middleware components that have application interfaces) must provide clear, accurate and comprehensive information about error conditions, or any other kind of failure (for example, reasons for failing to find any matching resources). The error or failure status information returned by the middleware must be detailed and accurate enough to allow the user (or the application) to identify the reason for the failure and to take the appropriate corrective action.

The workload management system should not be a single point of failure. It should be a system containing redundancy to cope with system failures, transparent to the user.

The performance of the workload management system (i.e. response time) should be guaranteed within certain known QoS limits. The service should include mechanisms to protect from response time degradation under conditions of heavy load.

1.1.5. INFORMATION SYSTEM

The information system should provide a reliable and functional service for applications. This implies the information system not to be a single point of failure, i.e. it should be able to cope with system failures, transparent to the user.

In order to use the information system effectively, applications must first be able to obtain the schema used by the information system, so that they can use it to obtain, and make use of detailed information about available grid resources.

Next, applications need to obtain the addresses (i.e. URLs) of the available information system services. This information (including schema information) could be published in a top-level grid directory or grid information web pages. This same top level directory could also provide a running log about overall grid status and serviceability, known problems, outages, scheduled maintenance, useful sources of information, etc. This would be useful for users to locate the required information.

Details published in the grid information system should be protected by security controls just like any other grid resource. Application (or other client) requests to obtain details from the grid information system should first be checked to ensure the requesting entity is authorized to receive the requested details. For instance, details of grid services dedicated for use only by a particular VO should only be available to members of the same VO, and the grid system administrators VO. If a delegated process running on the grid requests information about a given class of resources (e.g. storage resources or replica services), it should only receive details about those resources that are available to the VO(s) which the process belongs to (in DataGrid a request for available Computing Elements returns information about all resources, including those that cannot be used by the VO the application belongs to).

The performance of the information system (i.e. request response time) should be guaranteed within certain known QoS limits. The service should include mechanisms to protect from response time degradation under conditions of heavy load, e.g. due to numerous, frequent updates to the information system by information producers at the same time as numerous, frequent requests by information consumers.

All available grid services should be published and discoverable in the grid information system, e.g. including resource brokers, replica catalogues, replica optimizers, metadata catalogues.

The information system should be able to detect wrongly configured sites. The information system should flag these systems as ‘unusable’, so the RB will not use them.

1.1.6. FABRIC MANAGEMENT

The procedure for a new site joining the grid must be clearly defined, well documented and published so that it is easily accessible to potential new sites. The grid site installation and configuration procedure must be simple, automatic and straightforward. As far as possible, the installation procedure must follow standard software installation procedures and conventions. There must be no assumption or requirement of any previous grid system knowledge or experience by the local site system administrators. Fast turnaround installation support by email and telephone must be available for local site system administrators.

It should not be necessary to wait until a site is properly certified before it can be registered in the production grid information system; a site should be automatically registered as part of the installation and configuration procedure. However, site status published in the grid information system should be properly set, according to the actual site status (i.e. whether fully certified, or undergoing installation and acceptance tests, or offline, down for maintenance, etc). The Resource Broker, or any other potential client services, should specifically check the published site status before using any resource.

The site installation and configuration procedure should include automatic launching of self-checking procedures. Once the automatic procedures are passed, the local system administrators should run specific acceptance testing and certification procedures to confirm the site integrity. The site (or resource) status published in the grid information system must be updated only after the certification has been successful.

1.1.7. MASS STORAGE MANAGEMENT

A Grid aware Mass Storage management system, or Storage Resource, should provide uniform user and local administrator interfaces for different types of mass storage system implementations. The Storage Resource should provide functionality that allows the local system administrator to control the way that the device is shared by grid users, i.e. chiefly who has access to which areas of the storage

system and how the available space is used. It should present the grid user or application with a standard grid storage resource interface that abstracts the details of the underlying system.

The basic functionality required by applications is the ability to store, retrieve and delete data files, whether replicas or master copies. In addition the application must be able to find out the amount of storage space available, and to reserve space for a determinate period of time.

Standard, fine grained grid security control and checking functionality, based on VO/group/role identities (as described in the section on security requirements) must be an integral, automatic part of the storage resource service, its operation should be transparent and non intrusive to the user or application use of the service.

The storage service should ensure a high degree of reliability and should perform automatic checks to ensure integrity of data at all times.

The storage resource may be either a stand-alone disk system or a front-end interface to an archive system, or both. In the case of a disk system and a front-end interface to an archive system, the storage resource should provide functionality to automatically manage the disk system as a cache for the archive system. This involves automatically managing which files are allowed to stay in the cache and which files should be moved to the archive storage. A file pinning mechanism must be provided in case the application has to ensure files are not moved while they are being used.

The storage resource service must continually monitor free space available and should never allow the system to become completely full and unusable as a result. The service should warn clients and system administrators well in advance if free space falls below a specified safe threshold, so that advance remedial action can be taken well before the problem becomes critical enough to impact on essential operational services.

As files are continually added to a storage resource, it must be possible to ensure a minimum amount of free space is available in the cache, by automatically moving unused files to the archive; this should be done on a least recently accessed basis. The storage resource middleware should record the times of file accesses; the least recently accessed files should be the first candidates to be moved to the archive.

When a request arrives to access a file that has been moved to the archive, the storage resource middleware should automatically handle the retrieval of the file back into the cache. Files that are created on the storage resource should be assigned a default pinning time to live, that can be specified by the application. After this time has elapsed they can be moved to a secondary storage area.

Since the replica catalogue will keep track of files stored on different storage resources, the storage resource middleware and the replica management middleware must be must be closely integrated. In particular, when asked to return a reference to a physical file location, the replica catalogue should first check the availability (i.e. serviceability) of the referenced storage resource, in order to avoid returning references to physical files that cannot be accessed because the associated store resource service is down (e.g. for maintenance or other hardware fault). If a replica catalogue entry contains a reference to a physical file on a storage element that cannot be accessed (i.e. because the storage resource service in not working, for whatever reason), the replica manager should consider the entry to be invalid and ignore it (it should not return the invalid physical reference to the application to handle the error).

The storage resource must be capable of handling millions of files per VO/group. It must be capable of handling file sizes anywhere between e.g. 10kb and 100Gb.

A single storage resource may operate one or more storage resource services, possibly on a per VO basis. In order to avoid degradation of performance, the number of services running at any time should be dimensioned dynamically according to levels of utilization (i.e. number of requests).

When requested by the application, the storage resource service should provide QoS estimates on transfer times to, or from one or more host locations specified by the application. The service may in turn make use of a data transfer optimization service in order to calculate the requested QoS estimates.

1.1.8. SYSTEM INTEGRATION

Since grid is all about intercollaboration and cooperation, grid components typically work together with other components in order to carry out the required operations. Therefore it is essential that developers take special care to clearly specify the interfaces between components that work side by side, e.g. resource broker and replica catalogue, replica catalogue and storage element, information provider and information user, etc. Often there is a mismatch between interfaces that needs to be clarified; usually this arises when it is not immediately clear which components should be responsible for certain functionality lying close to the borderlines of certain component interfaces.

Standard mechanisms and approach to fault tolerance, error handling and recovery should be pervasive throughout all middleware components. Services that run as background processes are often subject to failure, due to unexpected process termination without warning, entering an endless wait or sleep state, or getting stuck in a loop. To detect such occurrences, essential services should be monitored and restarted automatically when they fail.

A standard set of grid error conditions and associated error codes, comparable to standard HTTP response header codes, e.g. 404 page not found, could be very useful to characterize and classify known, frequent error conditions and failures, whether due to system or user failures (or both).

Middleware components that represent potential single points of failure should be avoided, e.g. by providing redundant backup services which clients can use as an alternative.

There should be recommendations for middleware developers to follow a consistent model/format for APIs and command line commands and options.

1.1.9. APPLICATION DEPLOYMENT

Standard procedures must be available for packaging and deploying application environments on all grid sites where a given VO or VO sub group is authorized to use processing resources. This procedure must be well documented and published so that application integrators can easily identify, obtain and use the procedure to deploy their application environments.

To allow for upgrades, it must be possible for applications to install several versions of the same environment, capable of being used side by side, simultaneously.

It is essential to have a well defined and simple mechanism to allow the deployed application to specify the unique id signature which identifies the application environment, that gets published, i.e. as part if the CE information, in the grid information system by the local information provider, which is the same runtime environment identifier value that will be specified by the application or user when composing the JDL script.

1.1.10. DOCUMENTATION AND USER SUPPORT

Middleware components created by different sources making up the overall grid middleware should conform to use a well-defined documentation model and format. The documentation should have clear version management.

A standard documentation hierarchy suite should be specified and adhered to by all grid middleware developers and service operators, e.g. installation guide, administration guide, user guide, troubleshooting guide, etc.

A consistent documentation model would assist system administrators and users to locate useful information, independent of the type of middleware component.

1.1.11. MISCELLANEOUS

An important point for the application is the availability on the grid of different languages and COTS. In order to port new applications, the following languages generally used in ES are needed IDL,

Matlab or clones Scilab or Octave, Fortran 77, Fortran 90, and C. The climate community uses NetCDF, BLAS/LAPACK, Python, Ferret…

Other difficulty, that occurs, is related to the version of software available on the editor server and the one required by EGEE. As an example, the version of Netscape required by EGEE is no more available on the Netscape server.

1.2. Summary of Recommendations for EGEE development

The requirements described in the previous sections are summarized here, using a similar format to the one used for the recommendations of the EDG Application Working Group. The priority is assigned from 1 (low) to 5 (high). Even if the average is low some applications (priority 5) cannot run if the functionality is not there. These recommendations should be considered in addition to the AWG recommendations and requirements presented in the two documents **Error! Reference source not found.Error! Reference source not found.**.

1.2.1. VO MANAGEMENT

VO management	Priority			
	EO	Climate	Solid Earth	Average
Definition of groups within a VO	5	3	5	4.3
Definition of roles within a VO	2	4	3	3
Assign to a user one or more groups and/or one or more roles	5	5	3	4.3
Web based VO database administrator interface	3	2	3	2.7
Administrator interface login via the user's certificate loaded in browser	3	2	3	2.7

1.2.2. SECURITY

Security	Priority			
	EO	Climate	Solid Earth	Average
Clear and straightforward procedure for obtaining a certificate and joining the Grid	4	4	5	4.3
Unified, common security protocol, standard for all services and resources	4	4	5	4.3
Standard security protocols seamlessly integrated into all components as fundamental design feature	4	4	5	4.3
From user perspective, security mechanisms should be non intrusive, transparent and should present similar functionality, independent of the type of grid resources being accessed	4	4	5	4.3
User may belong to any number and combination of VO(s)/group(s)/role(s)	4	4	4	4
User may possess a single certificate valid for all associated VO(s)/group(s)/role(s)	4	3	4	3.7
Private key stored only on users PC; proxy generation directly from the users PC	3	1	3	2.3

User able to specify the required VO/group/role attributes to be associated to the proxy certificate during single sign on	5	4	4	4.3

1.2.3. DATA MANAGEMENT

Data management	Priority			
	EO	Climate	Solid Earth	Average
Fast performance of replica catalogue operations, resistance to degradation under heavy loads, whether due to increased data or user numbers	5	5	4	4.7
QoS response times published by each replica manager service in the grid information system, retrievable by the end user/application	3	3	3	3
All functions, e.g. creating replicas, file transfers, inserting, updating and retrieving replica catalogue information and metadata, must include proper security checks on the proxy-delegated permissions	5	3	5	4.3
Mechanisms for establishing and operating QoS levels to guarantee validity and integrity of replicated data	3	2	3	2.7
Data replication (i.e. data transfer and catalogue metadata registration) must be performed as an atomic operation	4	4	3	3.7
Correct management of disk space quota checking integrated in the data transfer operations	5	5	5	5
Advance, pre-emptive warnings when available space falls below threshold levels	4	4	4	4
Automatic data integrity checks embedded in replica manager operations	4	4	4	4
Capability to handle (storing, searching, updating, retrieving) file numbers of the order of millions of files	4	3	3	3.3
An application must be able to create, populate, utilize and destroy replica catalogues on the fly, as required by the application, without the need for human administrator intervention, having performed the appropriate security and resource utilization quota checks	2	2	3	2.3
RC integrated support for user or application defined metadata associated to logical filenames, supporting wide range of available data types (e.g. date, time, polygon, integer, alphanumeric, float)	4	5	4	4.3
Register both the logical filename and the associated set of metadata keys (list of key=value tuples) in a single operation	3	4	3	3.3
Retrieve full set of metadata tuples associated to a LFN in a single operation	4	4	5	4.3
Capability to store millions of files per VO	4	3	3	3.3
Mirroring of metadata and data bases	3	3	3	3

1.2.4. WORKLOAD MANAGEMENT

Workload management	Priority			
	EO	Climate	Solid Earth	Average
Method for jobs running on the worker nodes to retrieve attributes and values that were specified in the JDL	3	3	2	2.7
Improved error messages	4	5	4	4.3
Provision and enforcement of job submission quotas to prevent overloading the RB	4	3	4	3.7
Monitoring of RB state and job submission timings with results dynamically published in GIS, to enable user/application to choose RB service based on current performance and loading	3	4	4	3.7
Mechanisms to protect from response time degradation under conditions of heavy loading	4	3	3	3.3

1.2.5. INFORMATION SYSTEM

Information system	Priority			
	EO	Climate	Solid Earth	Average
Improved reliability and fault tolerance, eliminate vulnerabilities due to single points of failure	4	4	4	4
Schema to include grid descriptor info to allow applications to determine the type of Grid environment e.g. production grid, development grid	3	2	4	3
Mechanisms to enforce security restrictions on published information, e.g. info about VO resources only available to members of the VO	3	2	3	2.7
Guaranteed response times within published QoS figures	2	2	2	2
Schema to include info on all available services, e.g. resource brokers, replica catalogues, replica optimisers, metadata catalogues, etc.	3	3	5	3.7
To allow clients to determine whether a service is usable or not, the schema should support inclusion of current serviceability status for all published services	4	4	4	4

1.2.6. FABRIC MANAGEMENT

Fabric management	Priority			
	EO	Climate	Solid Earth	Average
Fast, easy site installation and verification procedures for new sites joining the Grid	4	4	4	4
Mechanisms for scheduling maintenance and dealing with outages, forewarning users about service problems, downtime or unavailability	3	4	4	3.7

1.2.7. MASS STORAGE MANAGEMENT

Mass storage management	Priority			
	EO	Climate	Solid Earth	Average
Mechanisms for administrators to perform storage space allocation and management among VOs and users contending for the same storage resource	4	4	4	4
QoS mechanisms allowing applications to determine service level conditions related to: operation (commands) response times (e.g. for storing, retrieving, or deleting), space management (total, used, free and reserve space available, procedure to request more space), data integrity and privacy levels and guarantees	3	3	3	3
Advance warning when available space falls below reserve levels	4	4	4	4
Automated management of disk cache by file swapping to back end archive storage device	1	5	3	3

1.2.8. SYSTEM INTEGRATION

System Integration	Priority			
	EO	Climate	Solid Earth	Average
The grid middleware must provide applications with both command-line commands and library embedded APIs	4	4	5	4.3
Uniform configuration across the grid (i.e. available software and installed versions) for each type of resource or service	3	4	4	3.7
Standard mechanisms and approach to fault tolerance, error handling and recovery supplied to all middleware components.	4	4	4	4
Common standard (guidelines) for command line formats and APIs	4	4	5	4.3
Standardization of error codes, error messages and error handling procedures	4	4	5	4.3

1.2.9. APPLICATION DEPLOYMENT

Application deployment	Priority			
	EO	Climate	Solid Earth	Average
Standard procedures for packaging and deploying application environments on CEs	3	5	4	4

Ability to deploy and utilize different versions of the same application environment e.g. IDL v5.4, IDL v5.6	3	3	4	3.3
Fortran 90	4	5	5	4.7
Fortran 77	4	1	2	2.3
Matlab or clones: Scilab, Octave	3	1	2	2
MPI	2	3	4	3
BLAS/LAPACK or clones	1	5	0	2
netCDF	2	5	2	3

1.2.10. DOCUMENTATION AND USER SUPPORT

Documentation and user support	**Priority**			
	EO	**Climate**	**Solid Earth**	**Average**
Common standard for documentation, consistent documentation model, supported tools and formats across all middleware packages	4	4	5	4.3

1.2.11. MISCELLANEOUS

Miscellaneous	**Priority**			
	EO	**Climate**	**Solid Earth**	**Average**