



LCG

Baseline Services Group Report

Version 1.0

June 24, 2005

Edited: Ian Bird

Baseline Services Working Group:

ALICE: Latchezar Betev

ATLAS: Miguel Branco, Alessandro de Salvo

CMS: Peter Elmer, Stefano Lacaprara

LHCb: Philippe Charpentier, Andrei Tsaregorodtsev

ARDA: Julia Andreeva

Applications Area: Dirk Düllmann

gLite: Erwin Laure

Tier 1 representatives:

Steve Traylen (RAL), Flavia Donno (CNAF), Ruth Pordes (FNAL),
Razvan Popescu (BNL), Anders Wannanen (NDGF),

Secretary: Markus Schulz

Chair: Ian Bird



Contents

| | | |
|------|---|----|
| 1 | Introduction..... | 3 |
| 1.1 | Goals and Mandate..... | 3 |
| 1.2 | Group Membership..... | 4 |
| 1.3 | Baseline Services Classification..... | 4 |
| 2 | Baseline Services..... | 7 |
| 2.1 | Storage Element Services..... | 7 |
| 2.2 | Basic Data Transfer Tools..... | 11 |
| 2.3 | Reliable File Transfer Services..... | 11 |
| 2.4 | Grid Catalogue Services..... | 13 |
| 2.5 | Catalogue and Data Management Tools..... | 18 |
| 2.6 | Compute Resource Services..... | 18 |
| 2.7 | Workload Management..... | 19 |
| 2.8 | VO Agents..... | 20 |
| 2.9 | VO Management Service..... | 20 |
| 2.10 | Database Services..... | 20 |
| 2.11 | POSIX-like I/O Services..... | 21 |
| 2.12 | Application Software Installation Facilities..... | 21 |
| 2.13 | Job Monitoring Tools..... | 22 |
| 2.14 | Reliable Messaging..... | 22 |
| 2.15 | Information System..... | 22 |
| 3 | Interoperability..... | 24 |
| 4 | Future Work..... | 25 |
| 5 | References..... | 26 |



1 Introduction

The LCG architecture will consist of an agreed set of services and applications running on the grid infrastructures provided by the LCG partners. These infrastructures at the present consist of those provided by the Enabling Grids for E-science (EGEE) project in Europe, the Open Science Grid (OSG) in the U.S.A. and the Nordic Data Grid Facility (NDGF) project in the Nordic countries. The EGEE infrastructure brings together many of the national and regional grid programs into a single unified infrastructure. In addition, many of the LCG sites in the Asia-Pacific region run the EGEE/LCG-2 middleware stack and appear as an integral part of the EGEE infrastructure. At the time of writing (April 2005) each of these projects is running different middleware stacks, although there are many underlying commonalities. Some sites also present interfaces to two of the grid infrastructures.

The essential grid services should be provided to the LHC experiments by each of these infrastructures according to the needs of the experiments and by agreement between LCG, the sites, and the experiments as to how these services will be made available. The set of fundamental services are based on those agreed and described by the present report. Where a single unique implementation of these services is not possible, each infrastructure must provide an equivalent service according to an agreed set of functionalities, and conforming to the agreed set of interfaces.

This report presents the conclusions from the group on the set of baseline services discussed and understood as essential to support the basic computing models of the experiments for LHC startup. It does not try to reproduce the full range of the discussions – for that the agendas, minutes, and presentations in the working group are available for reference.

This report is not the final word, in several cases it is clear much more work still remains to be done, and the group proposes to continue – with a modified mandate to follow up on this work and to be a forum where many of these technical issues can be discussed.

1.1 Goals and Mandate

The mandated goals of the group were the following:

- To negotiate agreement between the experiments, the regional centres and the LCG project on the baseline services that:
 - Are essential to support the computing models of the experiment for the initial period of LHC running,
 - and that must be in operation by September 2006 in order to achieve that.
- The services concerned are those that supplement the basic services of operating system, local cluster management, compilers, etc., and which are not already covered by other LCG groups such as the *Tier-0/1 Networking Group* or the *3D Project*.
- The results of the group are needed as input to the LCG TDR, and as such the group should report by end April 2005.
- The group should



- Define services with targets for functionality & scalability/performance metrics.
- Propose services that are feasible within next 12 months, i.e. for post SC4 (May 2006), and propose fall-back solutions where this is not considered feasible
- When the report is available the project must negotiate, where necessary, work programmes with the software providers.
- One of the explicit goals is to expose the experiment plans and ideas, and to understand for each service where the responsibility to provide it lies.

The timescales relevant for this work are essentially two - around the service challenges:

- SC3 (July 2005) to validate and test new and existing components
- SC4 (February 2006) to have available all missing tools and services
- September 2006: to have the production service in place.

1.2 Group Membership

The members of the group were the following:

| | |
|--------------------------|--|
| ALICE | Latchezar Betev |
| ATLAS | Miguel Branco, Alessandro de Salvo |
| CMS | Peter Elmer, Stefano Lacaprara |
| LHCb | Philippe Charpentier, Andrei Tsaregorodtsev |
| ARDA | Julia Andreeva |
| Applications Area | Dirk Düllmann |
| gLite | Erwin Laure |
| Tier 1 | Flavia Donno (Italy), Steve Traylen (UK), |
| representatives | Anders Waananen (NDGF), Razvan Popescu, Ruth Pordes (US) |
| Secretary | Markus Schulz |
| Chair | Ian Bird |

Additional technical experts were invited as needed for the discussions, and sub-groups were set up for specific work such as that of file transfer services and SRM functionality.

1.3 Baseline Services Classification

The following services are the current understanding of essential baseline services that need to be provided.

- 1) Storage Element
 - Interfaces: SRM, gridFTP, POSIX-I/O, authentication/ authorization/ accounting
- 2) Basic data transfer tools
 - gridftp and srmCopy
- 3) Reliable file transfer service
 - Wrapping basic gridFTP or srmCopy



- Providing overall scheduling, control, monitoring as well as reliability
- 4) Catalogue services
 - Interfaces: POOL, DLI/SI, POSIX-I/O
 - including a reliable asynchronous update service
- 5) Catalogue and data management tools
- 6) Compute Element
 - Defined set of interfaces to local resources,
- 7) Workload Management
 - Overall WLMS essential for some experiments, but not all
- 8) VO agents
- 9) VOMS
 - VO management system providing extended proxies describing roles, groups, and sub-groups
- 10) Database services
 - For catalogues, VOMS, FTS, as well as experiment-specific components
- 11) Posix-like I/O service
- 12) Application software installation
- 13) Job Monitoring tools
- 14) Reliable messaging/information transfer service
- 15) Information system

1.3.1 Priorities

In order to clarify the priorities for each of these defined services and how these priorities relate to each experiments' needs, in the table below the priorities of each service for each experiment is summarized according to the following scheme:

Priority A: High priority and mandatory;

Priority B: Standard solutions are required, but experiments could select different implementations;

Priority C: Desirable to have a common solution, but not essential

| <i>Service</i> | <i>ALICE</i> | <i>ATLAS</i> | <i>CMS</i> | <i>LHCb</i> |
|--|--------------|--------------|------------|-------------|
| <i>Storage Element</i> | A | A | A | A |
| <i>Basic transfer tools</i> | A | A | A | A |
| <i>Reliable file transfer service</i> | A | A | A/B | A |
| <i>Catalogue services</i> | B | B | B | B |
| <i>Catalogue and data management tools</i> | C | C | C | C |
| <i>Compute Element</i> | A | A | A | A |
| <i>Workload Management</i> | B/C | A | A | C |
| <i>VO agents</i> | A | A | A | A |
| <i>VOMS</i> | A | A | A | A |
| <i>Database services</i> | A | A | A | A |
| <i>Posix-I/O</i> | C | C | C | C |
| <i>Application software installation</i> | C | C | C | C |
| <i>Job monitoring tools</i> | C | C | C | C |
| <i>Reliable messaging service</i> | C | C | C | C |
| <i>Information system</i> | A | A | A | A |





2 Baseline Services

Below we describe for each of the agreed Baseline Services the current understanding of what the service should provide, the experiments' plans for use, existing implementations and issues.

2.1 Storage Element Services

A Storage Element (SE) is a logical entity that provides the following services and interfaces:

- Mass storage system, either disk pool or disk cache front-end backed by a tape system. Mass storage management systems currently in use include Castor, Enstore-dCache, HPSS and Tivoli for tape/disk systems, and dCache, LCG-dpm, and DRM for disk-only systems.
- SRM interface to provide a common way to access the MSS no matter what the implementation of the MSS. The Storage Resource Manager (SRM) defines a set of functions and services that a storage system provides in an MSS-implementation independent way. The Baseline Services working group has defined a set of SRM functionality that is *required* by all LCG sites. This set is based on SRM v1.1 with additional functionality (such as space reservation and pinning) from SRM v2.1. Existing SRM implementations currently deployed include Castor-SRM, dCache-SRM, DRM/HRM from LBNL, and the LCG dpm.
- gridFTP service to provide data transfer in and out of the SE to and from the grid. This is the essential basic mechanism by which data is imported to and exported from the SE. The implementation of this service must scale to the bandwidth required. Normally the gridftp transfer will be invoked indirectly via the File Transfer Service or through srmcopy.
- Local POSIX-like input/output facilities to the local site providing application access to the data on the SE. Currently this is available through rfiio, dCap, and their gsi-enabled versions, aioid, rootd, according to the implementation. It is also anticipated that xrootd will be available as an alternative method. Various mechanisms for hiding this complexity also exist, including the Grid File Access Library in LCG-2, and the gLiteIO service in gLite. Both of these mechanisms also include connections to the grid file catalogues to enable an application to open a file based on LFN or guid.
- Authentication, authorization and audit/accounting facilities. The SE should provide and respect ACLs for files and datasets that it owns, with access control based on the use of extended X509 proxy certificates with a user DN and attributes based on VOMS roles and groups. It is essential that a SE provide sufficient information to allow tracing of all activities for an agreed historical period, permitting audit on the activities. It should also provide information and statistics on the use of the storage resources, according to schema and policies to be defined.

A site may provide multiple SEs providing different qualities of storage. For example it may be considered convenient to provide an SE for data intended to remain for extended periods and a separate SE for data that is transient - needed



only for the lifetime of a job or set of jobs. Large sites with MSS-based SEs may also deploy disk-only SEs for such a purpose or for general use.

Since, most applications will not communicate with the storage systems directly, but will use ROOT (either via POOL or directly), it is clear that these applications must also be enabled to work with SRM interfaces. It is anticipated that ROOT will include the ability to talk to SRMs.

2.1.1 SRM functionality

The group has agreed on the following sets of required functionality from all SRM implementations. There are two timescales involved:

1. SC3 (July 2005). On this timescale the existing implementations of SRM v1.1 are sufficient.
2. SC4 (February 2006). For SC4 we require the full set of recommended functionality to be available. We label this the "LCG-required SRM" functionality, which all SRMs deployed at LCG sites are *required* to support. Of course, SRM implementations that offer more are acceptable, but the required set *must* be provided. For SC4 SRM interfaces are required to all storage elements.

2.1.1.1 Description of basic SRM functions from the SRM sub-group:

The various SRM specifications and draft specifications can be found linked to the Baseline Services group web page (<http://cern.ch/lcg/PEB/BS>) for reference.

- Type of files:
 - Volatile: a temporary and often sharable copy of an MSS resident file. The file, if not pinned, can be removed by the Garbage Collector if space is needed.
 - Durable: The file cannot be removed automatically. A user may assign this type for a new file if he/she does not know yet if the file should be copied to MSS. If the disk is full and space is needed, the local implementation may decide to copy the file to MSS or to send a mail to a VO admin. Such manual interventions are generally unwelcome.
 - Permanent: the system cannot remove the file.

The default file type is "Volatile".

- Type of space: The same 3 categories of space are defined corresponding to the 3 file types. A site may decide to offer different Quality Of Service for the 3 categories, for example better disk hardware for "Permanent" than for "Volatile". Normally a file of a certain type resides in a space (container) of the same type. But if the file type of one file is changed from "Volatile" to "Permanent", the site has the freedom to move the file to the "Permanent" space container or to keep it "Permanent" in the "Volatile" space container.
- Space reservation: In SRM v1.1, space reservation is done on file by file basis, the user does not know in advance if the Storage Element will be able to store all the files of a (multi-file) request. In SRM v2.1, an administrator or a user can reserve space in advance. The reservation has a lifetime associated with it. The user is given back a space token that he/she will provide in the following



PrepareToGet/PrepareToPut requests. When the space reserved has been exhausted, the next requests will fail with "No user space". In SRM v3.0 (draft), the notion of "streaming" mode is introduced: when space is exhausted, new requests will not fail but simply wait for space released by the user.

- Global space Release There are 2 modes:
 - Default: Permanent files are kept; Volatile files are kept until the Pin time expires.
 - Force: all the files (even Permanent or pinned) are removed.
- UpdateSpace: to change the amount of space reserved (the size of the container) and/or the space lifetime.
- CompactSpace: reclaim the space occupied by the released files (but the released files are not necessarily removed from disk) and update space size to current usage.
- Permission functions: In SRM v2.1, permissions (very similar to Posix ACLs) may be associated with each directory or file.
- Directory functions: to create/remove directories, delete files, rename directories or files. The (recursive) listing of directories is also possible but this operation can be very expensive. Directories in a managed storage system are namespace implementations.
- Data transfer functions: This is a very misleading name, as these functions do not normally move any data but prepare the access to data. The only exception is "Copy" which moves data between 2 Storage Elements (not between a Worker Node and a Storage Element). Almost all these functions can handle a set of files and in SRM v2 report a status / error message for each file
 - PrepareToGet is equivalent to stagein and make sure that a given file is available on disk. It may recall a file from a local MSS but does not get a file from a remote site.
 - PrepareToPut is equivalent to stageout and just reserve space for a file (on the Storage Element contacted).
 - PrepareToGet/PrepareToPut/Copy return a request token to be used in the following methods.
 - getRequestStatus (SRM v1) has been split for SRM v2 and v3 into 3 different methods StatusOfGetRequest, StatusOfPutRequest,
 - StatusOfCopyRequest. In the current v2 specification, there is no global request status, but only individual file statuses.
 - PutDone marks a new file as complete. At this point, the file can be copied to MSS.
 - ExtendFileLifeTime is equivalent to pin and associates a lifetime with a file. Several jobs/requests may give a different lifetime for a given file. The actual lifetime is the highest value.
 - ReleaseFile is equivalent to unpin. The actual lifetime is recomputed from the remaining pin values.
 - setFileStatus (SRM v1) has now disappeared.
 - GetRequestSummary gets the summary of a given request: how many files are ready, how many files are in progress and how many are still queued.



2.1.1.2 LCG-required SRM functionality

- SRM v1.1, together with the following capabilities from SRM v2.1:
- Pin/unpin of files with expiration time, for possible update,
- Space reservation and management
 - Ability to reserve space, release space, and update reservations.
- The need for “Volatile” and “Permanent” space was agreed, the “Durable” space type was not required. However, some existing instances of SRM do provide “Durable” space for certain applications.
- Basic directory functionality (ls etc) are requested. However, it was noted that some of these operations (e.g. ls on the full storage system) might result in responses that are difficult to handle.
- The storage system should respect permissions on directories based on VOMS roles and groups. No experiment has asked for file ownership by individuals.
- The abilities to abort, suspend and resume requests are requested but with low priority,
- The “srmgetprotocols” method was seen as useful. This function is also provided already in the information system.
- Relative path required with respect to a VO base directory

2.1.2 Implementations available

The currently existing implementations of SRM include:

- CASTOR (CERN), srm v1.1
- dCache (DESY+FNAL), srm v1.1 + extensions (= 2.1?)
- LCG-DPM (LCG), srm v1.1 + srm v2.1
- DRM (LBNL), srm v1.1

Of course, it is vital that all the SRM implementations interoperate seamlessly with each other and appear the same to applications and grid services. To this end an SRM test-suite will be used to validate SRM implementations against the LCG-agreed set of functionality and behaviour.

2.1.3 Experiment commitments and plans

In the discussions all of the experiments expressed strong statements that having SRM interfaces to storage was essential, and should be provided to all storage elements and storage systems. Although the SRM interface is the strategic choice for the grid interface for all storage systems, the transfer systems are expected to also support other protocols (specifically gridFTP) at smaller sites in the short term before SRM implementations are available everywhere.

The “LCG-set” of functionality described above, was agreed in the discussions with representatives from all experiments, and later confirmed by all. This functional set is now the subject of discussion and agreement between the experiments and the developers of the various SRM implementations, with the goal of having the agreed set of functionality in place by the end of 2005.

The need to be able to communicate with mass storage systems in order to communicate multiple requests and to allow the MSS to handle priorities, and to



optimize tape access was deemed as essential by the storage system managers, and recognized by the experiments. The use of the `srcopy` method provides this ability, and would be the preferred method of initiating file transfers, rather than calling `gridFTP` (or `globus-url-copy`) directly. This should be taken into account by the file transfer services layer.

2.2 Basic Data Transfer Tools

It is crucial that the basic underlying transfer mechanisms be made absolutely reliable. These are recognized as being `gridFTP` and `srcopy`. Note that `srcopy` in itself is not a transfer mechanism, but it in turn uses `gridFTP` to perform transfers. However, since it is likely that `srcopy` will become the interface in the majority of cases to the transfers it should be reliable.

`GridFTP` exists in the current production version from Globus toolkit 2. This will be replaced by the new version coming from Globus with GT4. This will be tested as a replacement for the GT2 version.

`GridFTP` is also the component that each site must ensure is implemented in a reliable and scalable service to ensure that the required transfer bandwidth can be achieved. Considerable planning and care in implementing this service must be taken to ensure that it is highly reliable, and that the site is able to load-balance the service across many physical machines, and that individual machine failures do not disrupt the service. The implementation should permit the service to be scaled to the required performance levels.

2.3 Reliable File Transfer Services

Basic level data transfer is provided by `gridFTP`. This may be invoked directly via the `globus-url-copy` command, or equivalent APIs (which of course do not know about the SRM) or through the `srcopy` command which provides 3rd-party copy between SRM systems. However, for reliable data transfer it is expected that an additional service above `srcopy` or `gridFTP` will be used. This is generically referred to as a reliable file transfer service (rfts). A specific implementation of this – this `gLite FTS` has been suggested by the Baseline Services Working group as a prototype implementation of such a service. The service itself is installed at the Tier 0 (for Tier0-Tier 1 transfers) and at the Tier 1s (for Tier 1 – Tier 2 transfers). It can also be used for 3rd-party transfers between sites that provide an SE. No service needs be installed at the remote site apart from the basic SE services described above. However, tools are available to allow the remote site to manage the transfer service.

For sites or grid infrastructures that wish to provide alternative implementations of such a service, it was agreed that the interfaces and functionality of the FTS will be taken as the current interface.

It was clear that the need for such a service above basic `srcopy` or `gridFTP` is needed to permit:

- Scheduling of requests, above what `srcopy` is able to do. This is particularly important for example, for large Tier 1 sites that need to be able to schedule and prioritize between different VOs competing for access to transfer bandwidth.



- Providing a mechanism for allowing interactions with other services such as updating grid catalogues. This function should not be coupled to the SRM or storage system directly.

2.3.1 Implementations available

The discussion was focused upon the implementation provided by gLite (the gLite FTS), which has an architecture based on that of the CMS PhEDex system. The implementation provides for asynchronous agents to provide the interaction with external components (such as experiment catalogues), will provide the ability to schedule bulk file movement, and can use SRM or gridFTP end-points for the transfers. Obviously for efficient bulk transfers, the SRM interface is needed.

The other existing implementation of a reliable transfer service is the Globus RFT. In its existing implementation this can only talk to gridFTP and not SRM, however it does allow for restarting transfers part way through a file which the FTS does not.

2.3.2 Experiment commitments and plans

Summary of issues discussed

The experiments all require the underlying base storage and file transfer infrastructure (gridFTP and SRM) to be available at high priority and provide an extremely robust and reliable service.

All experiments see the need for a more reliable transfer layer in the longer term, but the perceived urgencies differed due to differences in their current systems, some of which have this functionality.

The gLite FTS seems to satisfy the current requirements of the experiments and for all to integrate their current systems with this would require modest effort.

The ability to interact with other components such as the catalogues from the file transfer service is necessary, and is allowed for in the gLite implementation.

The difference between the SRM ability to copy files (srmcopy) and a file transfer service caused some discussion. Currently each has its own abilities, which may evolve. The main point is that the SRM (via srmcopy) can optimize access to the local mass storage system, staging all tapes for a given request close together for example, while a file transfer service can provide a view of the global state and load of the network and storage systems when scheduling transfers.

In order that other implementations of file transfer services could be provided, it will be important to agree on the interfaces. It was proposed that the client tools provided by the gLite FTS be the starting point for such an interface.

- **ALICE:**
 - Regard the file transfer service layer as a service that underlies data placement. They have previously used FTD (in AliEn) with aiod as transfer protocol for this, and are currently testing xrootd.
 - Expect gLite FTS to be tested with other data management services in SC3 - ALICE will participate.



- Expect implementation to allow for experiment-specific choices of higher level components like file catalogues
- **ATLAS:**
 - Don Quixote implements a file transfer service similar to the gLite FTS and works across 3 grid flavours
 - Accept current gLite FTS interface (with current FIFO request queue limitation). Willing to test it as soon as possible.
 - They would expect that their interface would be for Don Quixote to feed requests into FTS queue.
 - If these tests are acceptable, would want to integrate experiment catalogue interactions into the FTS, hopefully in time for SC3.
- **CMS:**
 - Currently PhedEx is used to transfer to CMS sites (including Tier 2 sites), and this satisfies CMS needs for production and data challenges
 - Highest priority is to have lowest layer (gridftp, SRM), and other local infrastructure available and production quality. Remaining errors handled by PhedEx
 - Work on a reliable file transfer service should not detract from this, but integrating as service under PhedEx is not regarded as a considerable effort (ideas from PhedEx were used in the design and implementation of gLite fTS with the explicit aim of allowing this integration)
- **LHCb:**
 - Have a service with a similar architecture, but with request stores at every site
 - Would integrate with the gLite FTS by writing agents for VO specific actions (eg catalogues),
 - Central request store is acceptable for now, having them at Tier 1s would allow scaling. This is the suggested deployment model.
 - LHCb would like to use this service in September for data created in service challenge, would like resources as soon as possible for integration and creation of agents

File placement services, which would provide a layer above a reliable file transfer service (providing routing and implementing replication policies), are currently seen as an experiment responsibility. In future such a service may become part of the basic infrastructure layer.

2.4 Grid Catalogue Services

The experiment models for locating datasets and files vary somewhat between the different experiments, but all rely on grid catalogues with a common set of features. These features include:

- Experiment-dependent information is kept in catalogues provided by the experiment (bookkeeping databases, experiment metadata, etc)
- All experiments have some form of collection (with varying names - datasets, collections). CMS go further and define "fileblocks" as (~TB) units of data management. CMS datasets point to sets of files within the fileblocks.
- The grid file catalogues are used for more than just data files, e.g. log files, etc.



- File Catalogues provide the mapping of Logical file names to GUID and Storage locations (SURL). In the ALICE file catalogue the storage location is kept as the Storage Index. The SURL mapping is part of the SE local catalogue.
- Hierarchical namespace (directory structure)
- Access control
 - At directory level in the catalogue
 - Directories in the catalogue for all users
 - Well defined set of roles (admin, production, etc)
- Support bulk-operations: Important: Dump entire catalog, support full catalogue scans so that external components (spiders) can read it
- Client-side monitoring of user activities – for inspecting the log files for example
- Interfaces are required to:
 - POOL
 - Workload Management Systems (e.g. Data Location Interface /Storage Index interfaces)
 - Posix-like I/O service

The deployment models also vary between the experiments, and are described in detail elsewhere in this document. The important points to note here are that each experiment expects a central catalogue which provides lookup ability to determine the location of replicas of datasets or files. This central catalogue may be supported by read-only copies of it regularly and frequently replicated locally or to a certain set of sites. There is however in all cases on a single master copy that receives all updates and from which the replicas are generated. Obviously this must be based on a very reliable database service.

ALICE, ATLAS and CMS also anticipate having local catalogues located at each Storage Element to provide the mapping for files stored in that SE. In this case the central catalogue need only provide the mapping to the site, the local catalogue at the site providing the full mapping to the local file handle by which the application can physically access the file. In the other cases where there is no such local catalogue this mapping must be kept in the central catalogue for all files.

The central catalogues must also provide an interface to the various workload management systems. These interfaces provide the location of Storage Elements that contain a file (or dataset) (specified by GUID or by logical file name) which the workload management system can use to determine which set of sites contain the data that the job needs. This interface should be based on the StorageIndex of gLite or the Data Location Interface of LCG/CMS. Both of these are very similar in function. Any catalogue providing these interfaces could be immediately usable by for example the Resource Broker or other similar workload managers.

The catalogues are required to provide authenticated and authorized access based on a set of roles, groups and sub-groups. The user will present an extended proxy-certificate, generated by the VOMS system. The catalogue implementations should provide access control at the directory level, and respect ACLs specified by either the user creating the entry or by the experiment catalogue administrator.



It is expected that a common set of command line catalogue management utilities be provided by all implementations of the catalogues. These will be based on the catalogue-manipulation tools in the lcg-utils set with various implementations for the different catalogues, but using the same set of commands and functionalities.

Summary of Experiment Catalogue needs and deployment models:

ALICE:

- Single central instance of an experiment file catalogue (the AliEn file catalogue).
- Replication of this catalogue at several geographical locations to improve performance is being tested.

LHCb:

- Central file catalogue in addition to experiment run bookkeeping catalogue
- Do not require local catalogues at each site
- Would like a few replicated read-only copies of the central catalogue at some Tier 1 sites for scalability and robustness

ATLAS:

- Will have a single experiment-specific central dataset catalogue, mapping datasets to files. The ATLAS model has no central catalogue of files or replicas.
- Will have catalogues at each site for files located at that site, to provide the local mapping (LFN → SURL). This is their only baseline requirement. They expect that the local catalogue will be a grid-infrastructure specific implementation.

CMS:

- Central dataset catalogue provided by the experiment
- Local site catalogues (as for ATLAS), to provide the local mapping. However, CMS did express a preference for a simple mapping function to derive the SURL from the LFN at a site rather than having a catalogue if it was possible.

No experiment currently sees the need for a distributed catalogue system (as in the full Globus/EDG RLS/RLI model). However, the need for replication of catalogues is clear. This issue is addressed through the 3D project, as are ideas for caching of catalogue data although this latter is clearly not seen as a baseline requirement.

Although not specifically a catalogue issue, it was clear that the storage systems are expected to respect a single set of ACLs in identical ways no matter how the access is done – whether the use is local, a grid user, uses Kerberos authentication, etc. There is also a clear distinction between security and access control in the catalogue and access control to the actual files. It was strongly felt that the access to storage must be managed at the storage service.

2.4.1 Catalogue Mappings

The operations that the experiment-specific and grid file catalogues need to support are the following:

- a) Query experiment metadata to determine a Logical File Name (LFN). This LFN could refer to a collection of files also.



- b) Resolve the LFN to guid before job submission, for example in the job creation, or job splitting process
- c) Resolve a guid or LFN to a site or SE in the workload management system to determine which sites have the data
- d) Resolve a guid or LFN to an SURL (physical location and means to access the file) from a job on a worker node, or from the WMS to create a catalogue fragment
- e) Register a new file (SURL) as a guid and LFN

The basic ways in which each of the experiments do these mappings are described below:

ALICE:

ALICE uses the AliEn file catalogue for all of these operations. The catalogue provides 3 interfaces for cases b), c), d) above. In addition ALICE require to be able to register new files in the central AliEn catalogue synchronously from a job on a worker node. This requires outbound connectivity from the worker nodes.

LHCb:

The LHCb bookkeeping database is used for a). Currently the resolution of guid/LFN to site/SE (case c) is done using the AliEn file catalogue, although the LFC is being tested as a replacement. For d) LHCb send an experiment-specific XML file with the job to the worker node to provide resolution of the physical file location.

LHCb have a central bookkeeping service that receives updates to the central catalogue with the provenance of new files created by the job. This update can be asynchronous with the job. File catalog updates are foreseen to be synchronous from the transfer agents (after successful file transfer). A simple relation between LFN and SURL is being envisaged in order to facilitate d) above (e.g. LFN as relative path from the VO base storage location in SRM).

CMS:

CMS will provide their own services to determine LFNs and Fileblocks from the experiment metadata, and to map Fileblock locations to sites. A LFN can refer to a file or a Fileblock. The LFN can be used by the WLM system to determine sites that provide the file or fileblock. The CMS job configuration file sent with the job in the input sandbox contains the LFN (or LFN-equivalents). The job matching stage uses Fileblock locations, but the job itself contains the list of actual files that the job needs. Once the job is at the site on the worker node, a site-local query resolves the LFN→SURL for each file.

The CMS Dataset Bookkeeping System is CMS-specific and is provided and maintained by the experiment. The Data Location Service is expected to be implemented as a grid service using a standard grid component.

Catalogue updates are performed using the job output sandbox. They could also make use of a reliable update service. It was noted that if the output file should move to a remote SE via the FTS then the FTS must provide the mechanism to do the catalogue update.



ATLAS:

ATLAS will use a central dataset catalogue to map datasets to LFNs and sites. This is an experiment-specific catalogue although they expect to reuse the POOL FC interface to map from datasets to files. The WLM will use the LFN to provide a list of sites containing the LFNs. ATLAS foresees having local catalogues at all sites which will provide the local mapping LFN/guid → SURL.

Catalogue updates are anticipated to be performed from a VO-owned service, or from a general reliable update service.

General points

- There is no clear requirement for lookup to a central (remote) catalogue by jobs on a worker node, jobs either arrive at the site with the necessary information, or use a local catalogue to resolve file names. However, in some analysis use-cases this might be required.
- But - ALICE does require outbound connectivity from worker nodes or an appropriate tunneling or proxy mechanism for *synchronous* catalogue updates
- ATLAS, CMS, LHCb, all have asynchronous catalogue update models. A reliable generic mechanism/service to do this should be provided. This could potentially be based on the proposed Reliable Replication Service (RRS) (ref xxx).
- A table was constructed to illustrate the similarities and differences in the mappings and needs of the various experiments. This table can be seen at <http://cern.ch/lcg/PEB/BS/baseline-cats.html>.

2.4.2 Catalogue Implementations and experiment usage

The existing grid file catalogues that provide all or some of the requirements described here are the following:

- LFC (LCG File Catalogue): provides all the interfaces described here: POOL, implements the DLI, and can be used together with GFAL.
- FireMan (gLite file catalogue): also provides all the interfaces described here: POOL, implements the Storage Index (and soon DLI also), and works with the gLite I/O service.
- Globus RLS: is now integrated with POOL. Does not (???) implement the DLI or Storage Index interfaces. Posix I/O ???

ALICE: will use the AliEn FC, but is testing both LFC and Fireman. The AliEn FC provides the mapping interfaces required by ALICE. It does not interface to POOL as this is not required by ALICE. The AliEn FC implements the Storage Index.

LHCb: Uses an old version of the AliEn FC, but notes performance issues that are fixed in the new AliEn FC, will evaluate LFC now and will test Fireman. The LHCb model allows the parallel use of different catalogue implementations for direct comparison of performance and scalability.

ATLAS: Will use an ATLAS provided catalogue as the central dataset catalogue.. Expect that the local site catalogues will be provided by the local grid infrastructure middleware. These local site catalogues must implement the POOL file catalogue interface. In the EGEE sites this will be LFC or Fireman, in the US likely to be the Globus RLS. In NDGF a prototype use of Fireman is under test.



CMS: Plan on a CMS implemented catalogue as the central catalogues, but LFC or Fireman could also fulfill this need. Local site catalogues will be implemented in the same way as for ATLAS – by the local grid infrastructures.

Note that, from the point of view of the sites it is extremely important that for example the local catalogue implementations for the different experiments be the same at a given site. The overhead of maintaining multiple catalogue flavours at a given site will be unacceptable. A VO could maintain and run a local catalogue within the VO agent mechanism.

In fact this is true for all services run by a site, experiments requiring several different implementations of a service will cause unnecessary work to support them.

2.5 Catalogue and Data Management Tools

Currently the LCG-2 distribution provides a complete set of tools for replica management and catalogue interaction and manipulation. gLite provides a similar set of tools. In addition POOL provides tools for POOL back-end catalogue manipulation. The result of the discussion of the group was that as far as possible these should be converged into a single set of common tools to be provided by LCG. It is clear that not all tools are relevant for all implementations of catalogues or related services. A first study was done by the gLite data management team to compare the functionalities provided by the lcg-utils tools and the glite-xxx tools. This comparison may be seen at <http://egee-jra1-dm.web.cern.ch/egee-jra1-dm/lcg-utils-glite.htm>.

2.6 Compute Resource Services

The Compute Element (CE) is the set of services that provide access to a local batch system running on a compute farm, or possibly to computing resources local to the CE host. Typically the CE provides access to a set of job queues within the batch system. How these queues are set up and configured is the responsibility of the site and is not discussed here. Note that there is an outstanding issue (or set of issues) related to how the local batch schedulers communicate information to grid workload managers. No fairsharing batch queue can respond to queries such as “how many slots are available to a VO”. A LRMS would like to have a reasonably sized queue to work effectively, whilst grid schedulers tend to prefer to submit to lightly loaded sites.

A CE is expected to provide the following functions and interfaces:

- A mechanism by which jobs may be submitted to the local batch system. This is implemented typically at present by the Globus gatekeeper in LCG-2 and Open Science Grid. Nordugrid (the ARC middleware) uses a different mechanism.
- Provision of information that describes the resources available at a site and the current state of those resources according to the GLUE schema. In particular this information should be published through the grid information system and associated information providers. It should also be available through a polling interface. With the introduction of new CE implementations we would expect that the GLUE schema, and evolutions of it, should be maintained as the common description of such information.



- Publication of accounting information, in an agreed schema, and at agreed intervals. Presently the schema used in both LCG-2 and OSG follows the GGF accounting schema. It is expected that this be maintained and evolved as a common schema for this purpose.
- A mechanism by which users or grid operators can query the status of jobs submitted to that site. This has been implemented in LCG-2.4.0 in the R-GMA based user level job monitoring.
- The Compute Element and associated local batch systems must provide authentication and authorization mechanisms based on the VOMS model. How that is implemented in terms of mapping grid user DNs to local users and groups, how roles and sub-groups are implemented, may be through different mechanisms in different grid infrastructures. However, the basic requirement is clear - the user presents an extended X509 proxy certificate, which may include a set of roles, groups, and sub-groups for which he is authorized, and the CE/batch system should respect those through appropriate mappings locally.

Several experiments expressed the need to be able to dynamically adjust the priorities between jobs of different users within the VO at a site. For example, to have the ability to say that a certain user has a high priority for a certain time. This could require a mechanism to dynamically alter the local batch system fairsharing within the VO, or by allowing the VO to manage its queue. However, it should be made clear that the requirement is to be able to dynamically (re)define priorities within the VO.

The CE should expose a standard interface for all the above functionality. It should accept actions from any sources providing the required level of security. In particular VO agents should be able to interact directly with CEs. CEs should have inbound connectivity.

It was pointed out that this ability was being pursued in the gLite CE together with Condor and Globus, where a clean separation between VO services implementing VO policies and resource services providing the necessary monitoring, auditing, and security context as required by resource providers, is being aimed at. This is at a preliminary stage.

It is anticipated that a new CE from gLite, based on Condor-C, will also be deployed and evaluated as a possible replacement for the existing Globus GRAM-based CEs within LCG-2 and Open Science Grid.

2.7 Workload Management

Various mechanisms are currently available to provide workflow and workload management. These may be at the application level or may be provided by the grid infrastructure as services to the applications. The general feature of these services is that they provide a mechanism through which the application can express its resource requirements, and the service will determine a site that fulfils those requirements and submit the job to that site.



It is anticipated that on the timescale of 2006-2007 there will be different implementations of such services available, for example the LCG-2 Resource Broker, and the Condor-G mechanism used by some applications in OSG, and new implementations such as that coming from gLite implementing both push and pull models of job submission.

The area of job workflow and workload management is one where there are expected to be continuing evolutions over the next few years, and these implementations will surely evolve and mature.

In order to work with the grid file catalogues specified here, any workload management system must use either the Data Location Interface (DLI) or the Storage Index (SI) or preferably both. Similarly any catalogue that provides these simple interfaces can be used with a WLMS (e.g. the gLite Resource Broker) to determine sites that have replicas of data needed for scheduling a job.

2.8 VO Agents

The LHC experiments require a mechanism to allow them to run long-lived agents at a site. These agents will perform activities on behalf of the experiment and its applications, such as submitting jobs to a CE, monitoring those jobs, scheduling file transfers, or scheduling database updates. No such general service currently exists, but solutions will be prototyped. Currently such actions are performed by experiment software running in the batch system, but this is not a good mechanism in the longer term as it could be seen as a misuse of the batch system. It is better to provide a generic solution which is accepted by the sites, but which provides the facilities needed by the applications. The deployment of the VO agents will be the responsibility of the VO. They may require additional local resources such as disk space or local databases (e.g. MySQL).

Further discussion on this subject and presentation of more detailed needs for these agents by the experiments can be seen in the Service Challenge 3 preparation workshop at: <http://agenda.cern.ch/fullAgenda.php?ida=a051784>.

2.9 VO Management Service

The VOMS software will be deployed to manage the membership of the virtual organizations. It will provide a service to generate extended proxy certificates for registered users which contain information about their authorized use of resources for that VO.

All of the grid infrastructure providers participating in LCG either have or intend to deploy a VOMS service. The mechanisms for implementing the mapping to users and groups, or to provide access control at a site may vary (potentially from site to site) depending on the local policies and requirements.

2.10 Database Services

Reliable database services are required at the Tier 0 and Tier 1 sites, and may be required at some or all of the Tier 2 sites depending on experiment configuration and need. These services provide the database backend for the grid file catalogues as



either central services located at CERN or local catalogues at the Tier 1 and Tier 2 sites. Reliable database services are also required for experiment-specific applications such as the experiment metadata and data location catalogues, the conditions databases and other application-specific uses. It is expected that these services will be based on scalable and reliable hardware using Oracle at the Tier 0, Tier 1 and large Tier 2 sites, and perhaps using MySQL on smaller sites. Where central database services are provided, replicas of those databases may be needed at other sites. The mechanism for this replication is that described by the 3D project (ref xxx). The 3D project will also provide the coordination for the set up of the database services with the database teams at the sites.

2.11 POSIX-like I/O Services

The LHC experiment applications require the ability to perform POSIX-like I/O operations on files (open, read, write, etc.). Many of these applications will perform such operations through intermediate libraries such as POOL and ROOT. In addition, other solutions are being deployed to allow such operations directly from the application. The LCG Grid File Access Library, the gLite IO service, and xrootd in Alien are examples of different implementations of such a service.

It is anticipated that all applications and libraries that provide this facility will communicate with grid file catalogues (local or remote), and if necessary the SRM interface of the SE in order that the file access can be done via the file LFN or guid. Thus these libraries will hide this complexity from the user.

It is not expected that remote file I/O to applications from other sites will be needed in the short term, although the mechanisms described above could provide it. Rather data should be moved to the local storage element before access, or new files should be written locally and subsequently copied remotely.

2.11.1 Existing implementations

- LCG-2 Grid File Access Library (GFAL). Provides file operations using LFN or guid, interfaces to the LFC and RLS catalogues, negotiates with an SRM and provides local i/o through rfio, dcap currently. Implemented as client libraries.
- gLiteIO. Implemented as a service. All I/O traffic goes via the service machine. Integrated with the Fireman catalogue, SRM, and can use rfio or dcap to access data in the SE, although traffic to the application uses the gliteIO protocol based on aiop. The security model assumes that the gliteIO service owns the files on the SE, access control lists are maintained in the Fireman catalogue.
- xrootd: This is the solution under evaluation in ALICE.

2.12 Application Software Installation Facilities

Currently each grid site provides an area of disk space, generally on a network file system, where VOs can install application software. Tools are provided in LCG-2, or by the experiments themselves to install software into these areas, and to later validate that installation. Generally, write access to these areas is limited to the experiment software manager. These tools will continue to be provided, and will be



further developed to provide the functionalities required by the experiments. In addition a tool is provided to publish the installed software in the information system and make use of it during job matching.

2.13 Job Monitoring Tools

The ability to monitor and trace jobs submitted to the grid is an essential functionality. There are some partial solutions available in the current systems (e.g. the LCG-2 Workload Management system provides a comprehensive logging and bookkeeping database), however, it they are far from being full solutions. Effort must be put into continuing to develop these basic tools, and to provide the users with the appropriate mechanisms through which jobs can be traced and monitored.

Monitoring at the level of jobs could be achieved by appropriate instrumentation of the script actually running the job (job wrapper) or the job script itself. As this monitoring may be VO-dependent, it would make use of monitoring agents.

Progress has been made on this since the discussion. In LCG-2 we now publish the RB information the current detailed status for every job in R-GMA. Users can find out information like CPU time, wall clock time, memory usage and WN for every job. Additional info is published for the FTP transfers. Simple command line interfaces to R-GMA will be provided.

The ARC middleware already provides job monitoring facilities.

In addition to user level monitoring, there is also a need for monitoring at the level of the VO, such that VO managers can determine the use made by users in the VO of resources, data access, etc. This requirement includes accounting data.

2.14 Reliable Messaging

This is noted here as a placeholder. Whilst not discussed, it became clear that there is scope for a reliable messaging system - the ability to send a message between applications, services, users, in a reliable (asynchronous) manner. Such a service would have a wide range of uses. Currently implemented by several experiments independently, a common trustworthy service would be of value. It should preferable make use of existing open source or public domain tools, and not be written from scratch.

2.15 Information System

This, although not discussed as a baseline service, is nevertheless a crucial part of the system. Normally the information published is accessed only indirectly by user applications via their interactions with other services. It is important however, that the information system and the associated information schema are adequate to represent the full scope of detail needed to describe services, their parameters, and their existence.

The GLUE schema is proposed as the standard schema for the information system, and an update to the schema in Q4 2005 will provide hopefully commonality between EGEE, OSG, and NDGF who have all agreed to collaborate on its



development. The goal of the updated schema is to address problems and lack of flexibility in the existing schema.



3 Interoperability

This report has presented the basic essential services that must be provided to the LHC experiments by all grid implementations. The majority of these deal with the basic interfaces from the grid services to the local computing and storage fabrics, and the mechanisms by which to interact with those fabrics. It is clear that these must be provided in such a way that the application should not have to be concerned with which grid infrastructure it is running on.

At the basic level of the CE and SE, both LCG-2 and OSG use the same middleware and implementations, both being based on the Virtual Data Toolkit. In addition, both use the same schema for describing these services, and have agreed to collaborate in ensuring that these continue to be compatible, preferably by agreeing to use a common implementation of the information system and information providers. Common work is also in hand on other basic services such as VOMS and its management interfaces. In addition, both EGEE and OSG projects are defining activities to ensure that interoperability remain a visible and essential component of the systems.

The ARC middleware used in the Nordic Data Grid Facility will provide the agreed baseline services, and already provides some of them. The ARC middleware intends to support community-agreed interfaces, such as JSDL for job description, and intends to implement agreed standards for authenticated access to CEs.

The Open Science Grid Software Stack provides baseline services on the US infrastructure based on the VDT. The current release provides CE, SE, VOMS and Information Services that interoperate with LCG-2. Interoperability of monitoring, accounting, VO access as well as in the roadmap for additional services are part of the future work.

The information system GLUE schema will be revisited and updated in the light of current experience and is being committed to by all of the grid infrastructures providing services to the LHC experiments.

With the basic services being defined in a clearer way through this working group, it is to be hoped that the existing grid infrastructures will be able to adapt themselves to be able to provide these essential services in a transparent way to the applications.



4 Future Work

It is clear that in many places the discussions were rather cursory, or uncovered other technical issues. These need to be explored and understood, and hopefully agreed.

Specific services where the group has identified more work is needed to understand and specify requirements, are the following:

- VO-specific agent service machines, and the requirements of the services
- Experiment (and other) application software installation mechanisms
- Monitoring tools
- Reliable messaging service
- VO policy management – priorities, access control, etc.
- Information service and developments

It was also apparent that this group fulfilled a missing need – that of providing a forum for in-depth technical discussions between the experiment experts, the grid middleware developers (although not all middleware projects were represented), the sites who need to run the services and the deployment team.

In addition, it would be appropriate to keep the group in place to follow up on the development, provision, and performance of the services described here. It could become the forum where further developments and changes are negotiated and proposed.

Two distinct functions can be defined:

1. Continuation of the present group to:
 - Continue the unfinished work of the group as discussed above, and address still open issues,
 - Produce agreed milestones, metrics, and interface definitions for the agreed services,
 - Follow up and monitor progress towards these goals,
2. Technical working forum, to address and agree issues, such as:
 - Interoperability between the different grid middleware stacks (EGEE, OSG, ARC)
 - Compatibility and coordination with the Applications Area projects,
 - Address and resolve technical issues (such as security models, storage access control, etc., etc.)
 - Agree on and monitor the development, release, and deployment of services

We therefore propose to maintain the group, with an appropriate mandate, and with perhaps a slightly wider membership to foster such technical discussions and to propose solutions to them, and to act as a forum for exposing the evolution of the ensemble of the LCG services. In addition we propose to the PEB the setting up of a second group to pursue the technical issues, which can have a broader and less formal membership. It is essential that representation of the sites in strong in the first group and that middleware developers (from all development projects) are present in the second group.



5 References

Mailing list:

- project-lcg-baseline-services@cern.ch

Web site:

- <http://cern.ch/lcg/peb/BS>

Agendas: (under PEB):

- <http://agenda.cern.ch/displayLevel.php?fid=31132>