Arun Jagatheesan, San Diego Supercomputer Center
January 16, 2006

The GGF Grid File System Architecture Workbook

Status of This Memo

This is an informational document to the Grid community on the design issues, requirements and status of the standard Grid File System Architecture. This is the version 1.0 of this document and its distribution is unlimited. The latest version of this document would be available at the Gridforge website.

Copyright Notice

Abstract

This is a working document that summarizes the discussions at the Grid File System working group. This document provides brief introductory information on: Why we need a Grid File System; What are the vendor/user expectations on such a system; What are the design issues and requirements that are being considered while designing the standard architecture for Grid File System. The Grid File System (GFS) enables the plug-n-play of data storage resources from different organizations and different vendor implementations into a logical resource namespace. This document is based on the information exchanged at the GFS-WG (F2F, Weekly Telecom, Mailing list). The content expressed in this document is expected to provide significant input to the standard recommendation document on GGF Grid File System Architecture. Any comments on this document could be discussed at gfs-wg@ggf.org. Latest version of this document could be obtained from Gridforge at the GGF Website [https://forge.gridforum.org/projects/gfs-wg]

Contents

1. Introduction

1.1 Background

A Grid is a coordinated distributed computing infrastructure formed by combining heterogeneous resources from autonomous administrative domains, as an ensemble to support large-scale, resource-intensive, and distributed applications. The resources in a Grid are shared via standard mechanisms based on local site policies along with a coordinated global grid policy. The combined distributed heterogeneous resources from the administrative domains create a logical resource namespace that could be used as a utility based on local and global grid policies.

The autonomous administrative domains that take part in a grid could be either from different organizations (forming an inter-organizational grid or virtual organization), or from the same organization (forming an intra-organizational grid or enterprise grid). The resources shared in a grid could possibly include computational resources, information (data), storage, services etc., Grid users and applications require a logical namespace for using and managing distributed data and other shared resources in the grid.

The Grid File System Working Group (GFS-WG) [https://forge.gridforum.org/projects/gfs-wg] in the Global Grid Forum (GGF) [GGF] aims to provide recommendations that lead to standard specifications of Grid File System Directory Services (Resource Namespace) and Architecture of Grid File System Services.

1.2 Purpose and Status of this document

This document is aimed at organizing the requirements and design issues that need to be considered in the proposed standard recommendation on the architecture of the Grid File System Services. This is a public information document produced by the GGF Grid File System Working Group, which is part of the Data Area in GGF.

The evolution of this document is an open processes influenced by the discussions in the e-mailing lists and at the GFS-WG sessions. This document serves as the Workbook for the GFS-Architecture design.

1.3 Intended Audience

This document is intended for a diverse audience. Expected readers include: People making technical decisions about Grid technologies; Designers and developers of Grid File System Services; Architects of data grid tools and products; and others interested in distributed information storage management standards.

1.4 Document Organization

In Section 2 of this workbook (document), we provide some uses of GFS and why we need it in enterprise architecture. Vendor expectations we have gathered are presented in Section 3. Our design goals are provided in Section 4. The current design issues are summarized in section 5.

1.5 Terminology Used

This section prepares the reader by introducing some well-known terms and their interpretation as per this GFS document.

1.5.1 Autonomous Administrative Domain

An autonomous administrative domain or sometimes referred as a “grid node” is a grid entity that:

- Manages one or more grid resources independently
- Can make its own policies
- Might abide by a superior or global grid policy

Each grid contains one or more autonomous administrative domains with distributed resources that are usually heterogeneous. An autonomous administrative domain, for example could be an organization, or sub-organizational business unit, or a department in a university.

1.5.2 Service (or Web Service)

"A Web service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically WSDL). Other systems interact with the Web service in a manner prescribed by its description using SOAP messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards." [*http://www.w3.org/TR/ws-arch/*]

1.5.3 Data Grid

A data-grid is a distributed infrastructure formed by combining heterogeneous data storage resources from autonomous administrative domains. Applications view data grid as a *federated location independent, logical resource namespace that is dynamically spread across multiple administrative domains. Datagrids support shared collections and shared resources between the autonomous administrative domains.* Data grids can be considered as a superset of the Grid File System and provide additional capabilities and functionalities in managing an inter/intra organizational IT infrastructure.

Additionally a glossary of other terms and concepts used in this document is also available towards the end of this document. **We strongly encourage you to read the glossary before you read the rest of this workbook document**.

2. What and Why GFS?

A Grid File System (GFS) is a software system that provides a federated logical resource namespace of heterogeneous distributed infrastructure comprising of data and digital representations of shared resources amongst the autonomous participants in a grid infrastructure. This federated logical resource namespace consists of human-readable persistent identifiers to represent logical organization of data, storage, services and other digital entities shared in a grid. A Grid File System is any software system that supports the standard GFS interfaces provided by the GFS-WG.

The value addition GFS will provide for enterprises and data centers include (but not limited to):

- GFS logical resource namespace spans multiple organizational sub-units. This enables ease of data sharing and data storage infrastructure management. GFS facilitates the creation of a data grid infrastructure, which allows multiple sub-units of an enterprise to share the data storage resources as a utility based on their business policies.

(Application: Inter/intra-organizational collaboration of sub-units, Enterprise Mergers)

- GFS logical resource namespace can include multiple heterogeneous file system resources and replicated data with a single naming structure. This allows ease of Information Lifecycle Management (ILM) for online and offline data

(Application: Management of backups and replication, data load balancing, data management based on storage resource type)

- Logical resources allow provisioning of resources based on demand. Since, the resources are logical, more logical instance of a physical resource could be created based on the business requirements. Data can be moved or replicated based on their requirements on availability, access patterns, compliance requirements.

(Application: Resource utilization based on data-requirements during the lifecycle of data)

- Multi-vendor supported file system resource management standard interfaces

(Application: No vendor lock-in for enterprise customers for data storage management solutions)

It is anticipated that the GFS becomes a fundamental standard over which other data grid standards could be built for inter/intra organizational, multi-vendor supported file system resource management.

3. Vendor/Implementer/User Expectations on GFS

In this section, we briefly describe the expectations on the GFS standards as expressed by different vendors and users. It must be mentioned that not all these expectations might be satisfied in the final GFS standards.

- Standard Interface
    - Well defined interface
    - Simple interface for clients
    - Multi-vendor Participation
- Plug-n-play of resources
    - Storage resources could be added or removed dynamically inter-intra organizational storage pool
    - Management of context of the resources
- Federation of logical resource namespace
    - Unifies heterogeneous distributed data sources
    - Inter/Intra organizational
    - Online and Offline data
- Inter/Intra Organizational
    - Dynamic usage by users from different organizations
    - Avoid cross-registration of users at sites
- Enables ILM on demand
    - Background data management in an enterprise could be automated easily using a single logical namespace
    - GFS namespace spreads across both online, near-online and offline data that could be used to seamlessly move only selected components of data with affecting the applications/users view
- Object based Storage
    - Define standard remote operations that can be performed on the stored objects (digital objects).
- Management of the context
    - Managing the context or metadata about data storage with respect to storage location, access controls, compliance etc has to be made simpler
- Support existing client protocols
    - Don't introduce new clients

4. Design Goals for GFS

The overall design goals of GFS Architecture are provided in this section.

4.1 *Facilitate utility and grid computing models*

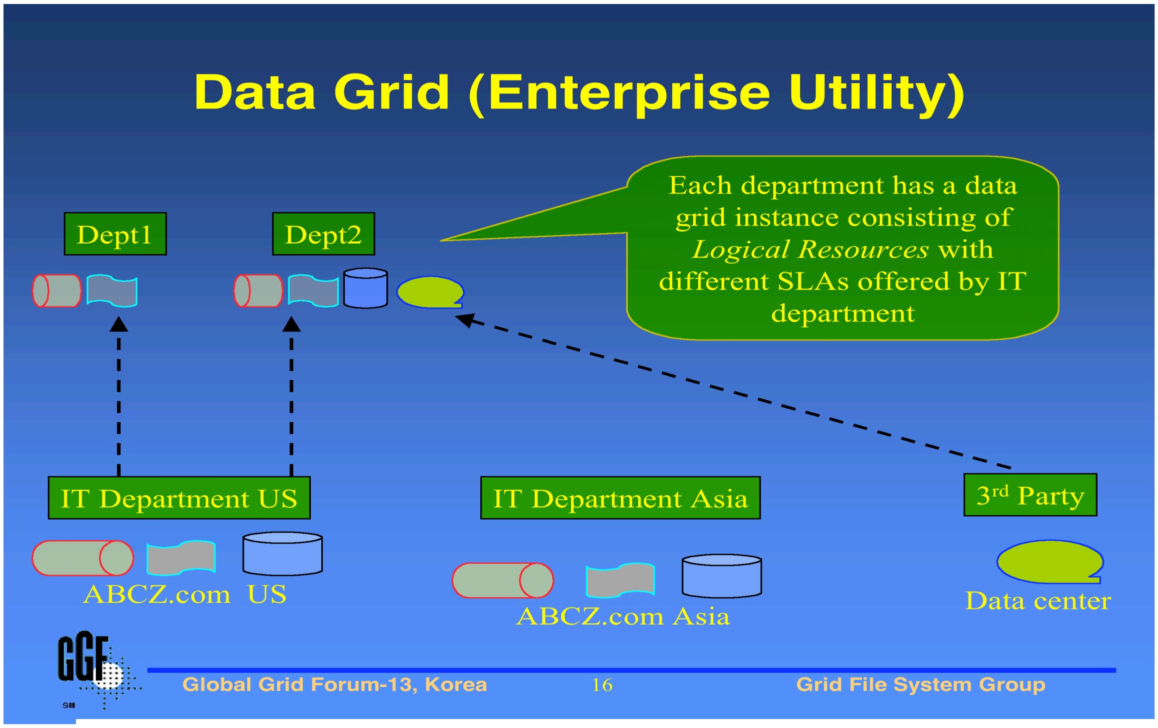


Figure 1: Data Grid Infrastructure as an enterprise utility of logical resources

In the grid/utility-computing model, a data grid infrastructure has to act as an inter/intra organizational utility that can support multiple qualities of service agreements based on its user's requirements and the available physical data storage resources. For example, let us consider an enterprise ABCZ with multiple departments that are served by a single IT department (Figure 1). ABCZ's IT staff could provide different data grid instances to its departments. Each data grid instance provided to each department is composed of multiple heterogeneous data storage resources that could satisfy different Service Level Agreement (SLA) based on each department's business requirements and policies. However, from the IT department's perspective, all the data grid instances are mapped to the different physical resources that might be managed by the IT department or by the data centers that support the ABCZ enterprise. Some of these logical resources could be shared between multiple departments. The proposed GFS does not create the entire data grid infrastructure discussed in the ABCZ example above. However, GFS standards become a foundation for the data grid standard to emerge. GFS standard interfaces provide a logical resource namespace for heterogeneous data storage resources that might be distributed in multiple grid nodes.

Again, in the grid/utility computing model, a data grid infrastructure has to allow coordinated sharing of data storage resources amongst inter/intra organizational grid nodes. The data storage resources that could be shared amongst the grid nodes include: data content (raw bytes), the meta-data and the physical resources used to store the data. Applying this scenario to our previous example of the ABCZ enterprise (Figure 2), this would relate to business usecases of sharing data and/or resources between the internal departments of ABCZ with each other or with external partners of ABCZ who might be in other countries. Each department might have its own logical view (data organization) of the data storage resources that is independent of any physical storage organization. A new collaboration would mean aggregation of a subset of the logical data resource views from the participants. A GFS interface provides the basic interface to create and manage the aggregated logical view of data resources from all the participants. A data grid instance would provide additional functionalities to manage the constraints associated with maintaining consistency of the physical resources that are presented in the aggregated logical view.

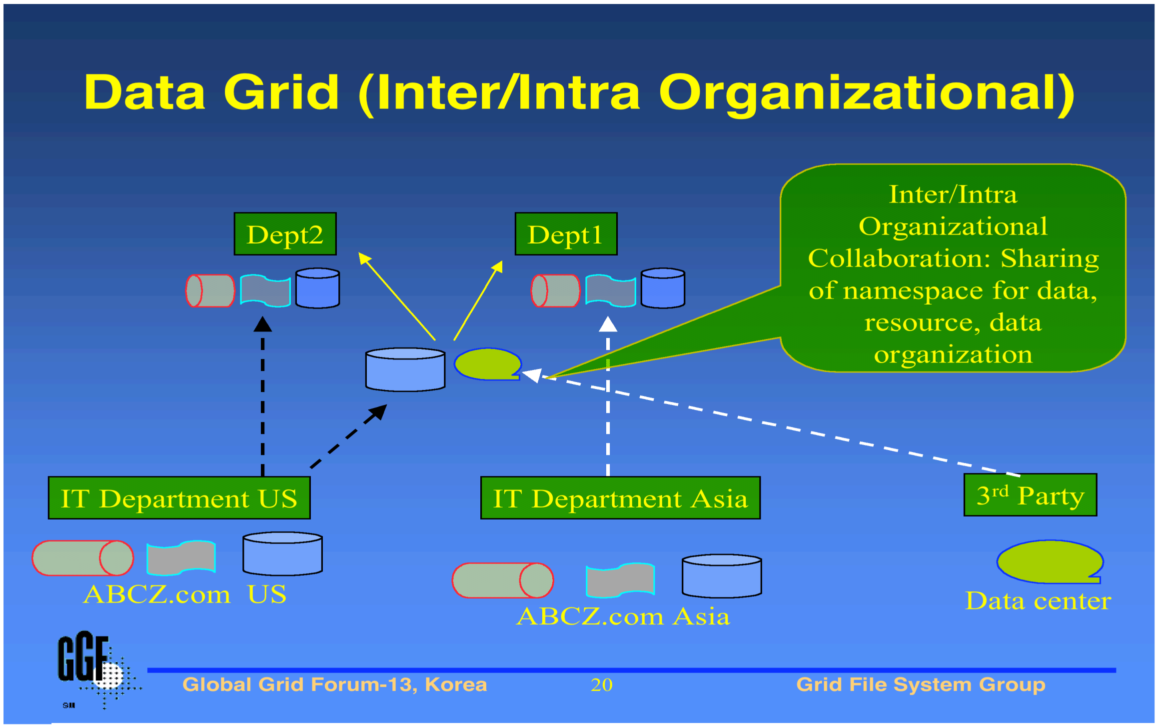


Figure 2: Data Grid Infrastructure for coordinated sharing of logical resources

4.2 Location and Infrastructure Independent Inter/Intra organizational Namespace

The logical resource namespace (including data and storage) that is rendered by GFS is dynamic and reflects the changes when autonomous domains that provide the resources join or leave the GFS. The namespace rendered is independent of location and infrastructure details allowing the data to be replicated/migrated to any file system resource at any location in the data grid. Even if the data is physically moved or migrated, the user applications still see the same logical resource namespace and are transparent to physical changes in data (even if the data is physically migrated from one storage system to another in a different organization based on the business demands or ILM processes).

The GFS namespace as shown in Figure 3, is logical and could be different from the underlying physical (file) system resources namespace. Each data entry in the GFS namespace represents a *resource endpoint* that would be mapped to a file (or any digital entity) in the underlying *Grid Resource Providers* (represented as "*GRP" in Figure 3).* Grid Resource Providers are the data storage resources that are logically mapped in the GFS namespace. GFS maintains the mapping between each *logical entry* in its logical resource namespace and the corresponding *resource endpoint* including the protocol and authentication mechanism needed to access the physical data in the GRP. The GFS-WG standards might not provide implementation details on how this mapping or state information will be stored by a GFS implementation. Vendors/Implementers can store this catalog information using LDAP or any database.

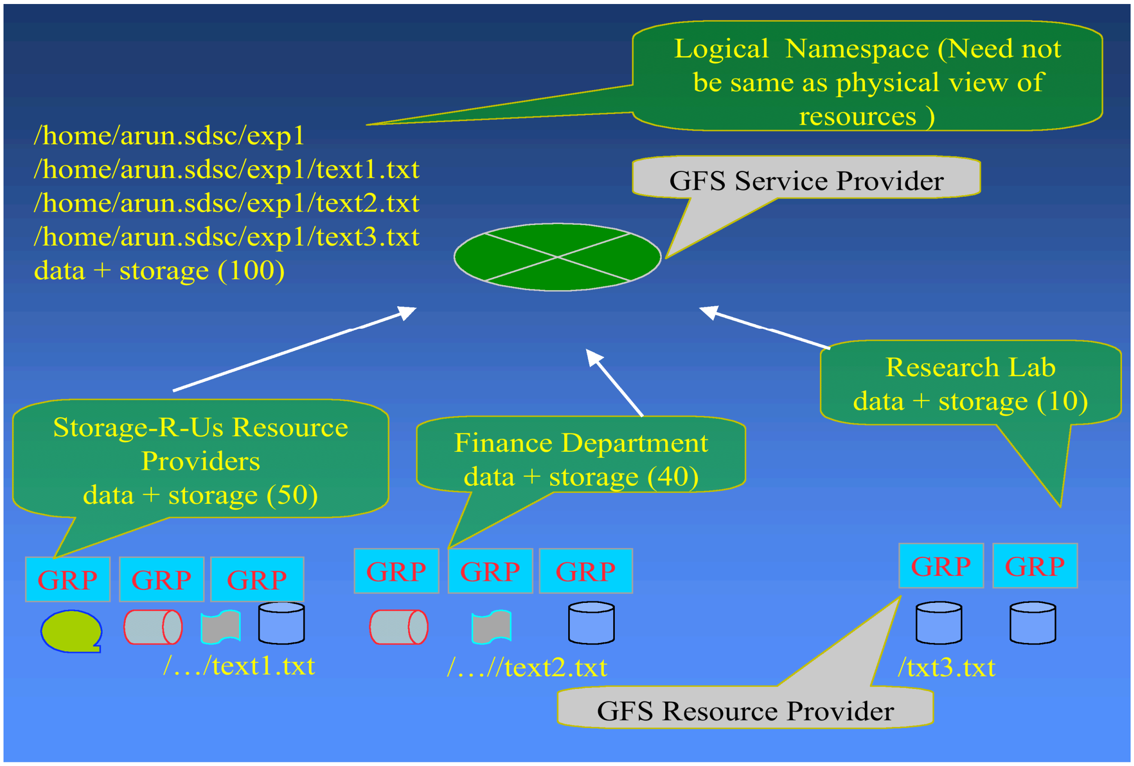


Figure 3: Grid File System - Infrastructure Independent Namespace

4.3 Bulk and asynchronous operations

The (file) system resources that take part in the GFS from different domains will be distributed across multiple administrative domains and data centers separated by a WAN. The control messages that need to be sent from each of the file system resources to the GFS have to be designed to work well in a WAN (Internet) environment. Bulk and asynchronous operations must be provided.

4.4 Independent of physical data access/transport, authentication mechanisms

The underlying file system resources used in the GFS could have their own transport, access, authentication/authorization protocols, which might be standardized or vendor specific. Each of the *data endpoints* have their own transport and own authentication mechanism.

In addition to the regular file system resources, some of the underlying resources in GFS could also be HTTP, FTP, or GridFTP servers with their files named using URLs. In the GFS namespace, they are represented using logical name, which hides the protocol and location. An additional requirement is to use GFS namespace as the aggregated view of all the unstructured and semi-structured data in the Grid.

5. Design issues that need to be discussed and resolved for the GFS Architecture

In this section, we introduce the major issues that need to be resolved before we can standardize the architecture and the standard interface for the GFS.

5.1 Criteria for choosing an option to solve an Issue

The issues mentioned here are actually WG design decisions that have different possible solutions. The GFS-WG needs to be making these decisions as a community for the use and proliferation of these standards. The GFS-WG needs to make sure that the proposed GFS-Standards are:

- A value addition to data storage products that support the proposed standards
- "Community-acceptable" and feasible for implementation by the participants of our GFS Working Group
- Appropriate solutions for the emerging requirements of very large-scale inter/inter-organizational data storage management in distributed enterprises or distributed data centers (rather than LAN based file storage systems for a single organizational unit).

5.2 Outstanding issues/concepts requiring attention

The following are the outstanding issues. (*We will keep updating this list and the options based on the discussions at the GFS-WG.)*

1. Logical resource namespace for grids and traditional file system namespace
2. Data-ownership and access control models
3. Use of logical storage resources to support collective operations (load leveling)
4. Work with existing standards without loosing advanced functionalities
5. Vendor interoperability without loosing advanced functionalities
6. If/can we take advantage of the GGF-Data area stack
7. Usage of grid and web Services. How can they be taken advantage off?
8. Grid File System Standard Interface
9. Basic data types

For the rest of this section, we discuss each issue outlined above and the different possible options or the solutions. Each solution provided has it advantages and disadvantages and we try (or need) to mention all these here in this section. The reader is encouraged/requested to provide comments or suggestions to the GFS mailing list regarding this Architecture Workbook.

5.2.1 Logical resource namespace for grids and traditional file system namespace

The GFS logical resource namespace could be a super set that is made of entries from different traditional file system namespace. Hence, the GFS logical resource namespace could have different users, different access controls, etc from the traditional file system resources. Even though, the hierarchical view of directories in GFS might be similar to the traditional file systems, the operations that could be performed (like migration, replication, etc.,) and the presence of multiple domains differentiates the GFS namespace from the traditional file system namespace.

The GFS Namespace will include digital entities from the traditional file system namespace. However, additional metadata or state information is present in the GFS Namespace that differential the GFS Namespace from the traditional file system namespace. This additional state information includes the information about replicas, mapping for each digital entity from the GFS namespace to the physical (or traditional file system) resources, access control information on the GFS namespace (which might differ from the traditional file system ACL), virtual organization information etc. In addition, the operations that are performed on the GFS namespace would differ from that of the traditional file system namespace. Hence, the GFS interface is different from the traditional file systems interface.

[*Agreed Resolution:* The GFS_WG has agreed that even though GFS and Traditional FS might present a similar hierarchical view, they are different and need different methodologies]

5.2.2 Intra-domain user visibility and associated data ownership/access model

In our example scenario described in section 4.2 (refer Fig.3), the location-independent logical resource namespace created by GFS could spread across all file storage system resources from different domains. This would enable users from one domain to access a file system resource transparently within another domain. A design goal is to enable such access without requiring the creation of a user account in the second domain, while preserving the ability to control access and identify file ownership.

The following are some possible options:

- *Cross-registrations of users*: Information about users are exchanged and cross-registered amongst all the participating domains. This is administratively difficult and not practically possible in some cases – but makes it easier for the GFS. This also allows the flexibility to provide granular access controls.
- *Organization as user:* Each domain knows only about other domains (but not their internal users). All users from an external domain are mapped to a user account based on their home domain or organization. This would require each domain to have a single account for each other domain in the data grid.
- *Template Roles:* Each domain has some pre-defined user-ids to which external users would be mapped. This would allow flexibility for the host domain to manage the external-user ids as well as create more customized user-id for their external users.
- *Shared collection ownership of the files.* The shared collections are assigned a local account in the remote domains (say *collection manager* account which runs the remote GFS servers). The collection manager owns or has access to the remote files in the remote storage system. Users authenticate themselves to the GFS and can have different access controls in the GFS. The GFS uses the *collection manager* account to access the data in the remote storage system.

Additionally, other related questions need to be discussed:

- Can data from one domain be placed physically in a file system resource from another domain? Who owns the data in such a scenario?
- Who owns the physical data that might be shared or cached as replicas in another domiain? Can we use the "*Collection Manager"* account as mentioned above?
- Do we need different Access controls from the traditional file systems? Who can replicate what data? Who can migrate from one physical data store to another? Who can create logical hierarchy or virtual directories?
- How is intra-domain user authentication/access control handled?

5.2.3 Logical Storage Resources

Can the storage resources from different organizations be categorized into different logical resource pools based on their QoS and used in the GFS namespace. This will allow virtualization and "plug-and-play" of the heterogeneous inter/intra organizational resources. We will have to analyze the advantages and disadvantages of this approach with respect to the inter/intra organizational (virtual organization) scenario, and provide a model for this if feasible. Logical resources can be used to implement collective operations such as fault tolerance, load leveling, and replication.

5.2.4 Work with existing standard interfaces without loosing advanced functionalities

As explained in section 5.2.1 above, the GFS interface has to be different from the traditional file systems interface to provide the advanced capabilities required in grid applications. However, NFS and CIFS are well known traditional file systems today. It will be advantageous for GFS to use the similar interface as that of NFS and CIFS. This will encourage easy adoption by users. Can there be a mapping provided from the operations in the well-known NFS/CIFS interfaces to the operations in the GFS Client interface? Even if the advanced operations or additional metadata of GFS might not be available from these traditional file system interfaces, the users can still use GFS indirectly using their well-known interface.

5.2.5 Vendor Interoperability without loosing advanced functionalities

The autonomous administrative domains that participate in a GFS namespace could use different vendor implementations to participate in the inter/intra organizational collaboration. The vendor implementations need follow the same standards that result in GFS interoperability amongst users. If we use a common or minimal denominator, the feature set available might not be that impressive. At the same time if we are more aggressive and require more features in the GFS standards, vendors might not find it easy to implement the entire feature set list (some of which might not be needed by their customers). To solve this problem, multiple levels of GFS-compatibility could be introduced. Vendors can choose to be compatible with a higher-level (or version of) GFS standard with more features and QoS or they can just begin with a lower level compatibility, which provides the basic capabilities for their users.

Vendors with existing data storage products should be able to easily support the basic GFS interface. A software interface at each storage repository (or system) might be provided that maps the GFS operations to the vendor product. This approach has been demonstrated in academic implementations to work across a very wide range of data storage systems (including databases, NFS, CIFS, tape, etc.,) without any modification to existing vendor technology, and can be tuned to support shared collections that are distributed across wide area networks.

We will have to provide a matrix of different levels of GFS-compatibility and the corresponding features set and QoS, the customers can expect for each of these levels.

5.2.6 If and how we take advantage of the GGF-Data area stack

The GGF Data Area has other working groups including:

- DAIS-WG (Data Access and Integration Services)
- GridFTP-WG (Grid File Transfer)
- INFOD-WG (Information Dissemination)
- OREP-WG (OGSA Replication Services)
- DFDL-WG (Data Format Description Language)

Also outside the data area,

- Grid Security Area: https://forge.gridforum.org/projects/sec (has related working groups)

We need to make sure if/how we relate to this groups and how we can take advantage of their specifications if any. This will make sure the GFS standard is used as a grid technology with any other related products or services seamlessly. We need to analyze on what standards we need to keep in mind while presenting our recommendation to the GGF.

5.2.7 Grid/Web Services and Traditional Clients

One of the aspects of Grid Computing is to provide resources as services. Once grid based utility computing becomes mainstream, enterprises will manage most of their IT infrastructure resources as services that can be offered to their internal or external users. IT infrastructure could be updated based on the user demand or the load on existing resources. The end users will be transparent to these changes and the utility model used by their IT team.

Applying this model to file system resources requires GFS to provide a service-oriented interface that could be used by Enterprise IT Infrastructure Management applications. In addition, grid applications would require a WS-Resource Framework compliant client interface that can be used to manage files and treat the Grid File System as well as any data (including files) as just a resource in the WS-Resource Framework.

On the other hand, most existing enterprise applications and existing vendor software investments prefer using non-client (or traditional CIFS/NFS client based interfaces). A service-oriented architecture at the file-level to manage the raw data might have significant compromises in performance penalties for the flexibility.

5.2.8 Grid File System Standard Service Interface
Grid users and developers would like to see a single interface that represents the entire GFS. This single interface could provide functionalities that are provided as Grid File System.

- Extension of RNS (Resource Namespace Services)
- RLSS (Replica Location and Selection Service)
- MDS (Meta Data Service)
- Resource Manager (Logical Resource with respect of File Systems only)
- File I/O
- User Access control management

All these functionalities represented in the GFS Service Interface would be from a GFS perspective. The implementer of this service can either build this whole interface as a single software stack (for efficiency purposes) or assemble the different components from the other service interfaces from (RNS, OREP-WG, Grid-FTP) etc.,

[*Resolution: Most of this was discussed and decided at the GGF12. Refer the* Appendix-1 *for the picture drawn at GGF12 in Brussels]*

5.2.9 Basic data types

The basic data types used in GFS namespace and their XML Schema need to be defined.

- We need to define the schema for basic data types needed like: Collection or Virtual directory and its listing
- Dataset or File
- Resource
- Access Permission

These have to be consistent with the requirements of GFS. For example, we might have complex requirements to list a collection that might have millions of files and the GFS server needs a way to batch or split the listing in those cases to multiple sets and relay the information to the GFS clients. The data type to list a collection must allow these types of requirements. The GFS implementation needs to define the size of collections that may be shared. Traditional file systems manage millions of files. Should the GFS implementation be designed to handle billions of files?

In addition, we need to find the applicability of any existing or emerging industry standards like DMTF CIM Schema [http://www.dmtf.org/standards/cim/] that might have schema definitions for systems similar to GFS or data grids.

6. Other standard bodies or Organizations

The GFS-WG effort has to work with other similar industrial or standardization efforts so that each effort benefits the other.

6.1 Enterprise Grid Alliance (EGA)
http://www.gridalliance.org/en/index.asp

6.2 Distributed Management Task Force (DMTF)
http://www.dmtf.org/home

6.3 The Internet Engineering Task Force (IETF)
http://www.ietf.org/

6.4 Storage Networking Industry Association (SNIA)
http://www.snia.org/home

6.5 Related Efforts

6.5.1 NFSv4 (IETF)
http://www.nfsv4.org/

6.5.2 CIM (DMTF)
http://www.dmtf.org/standards/cim/

7. Existing implementations

The GFS standardization effort could examine or request information from existing data grid implementations that manage hundreds of millions of files, distributed across up to a hundred resources, holding hundreds of terabytes of data, and spanning continents in scale. These systems provide excellent test beds for assessing performance, validating implementation strategies, and defining user requirements. The systems include:

- High-energy physics community with multiple applications (LHC, BaBar, Belle) trying different technology (Globus, EGEE, SRB, OSG)
- Worldwide Universities Network (federated data grids)
- Australian Partnership for Advanced Computing (APAC)

8. Summary

GFS standardization is a slow, but steady process. Once some critical issues are resolved and early adopters develop initial prototypes, community acceptance that would result in compliance by more vendors would be observed. For now, we need more "engineering management" and support from vendors with more engineers who can participate regularly in the WG and build reference implementations that align with their data storage solutions.

Editor Information

This workbook is intended to have contents provided by the members of the GFS-WG. This initial version of this workbook was written/compiled by Arun swaran Jagatheesan (arun@sdsc.edu) of the San Diego Supercomputer Center. Your suggestions and participation to make this workbook more useful are welcome. This document can be discussed at the GGF Grid File System Working Group (gfs-wg@ggf.org).

Contributor Information

The information provided here is compiled from the discussions at the Grid File System Working Group sessions and Teleconference calls. People from different organizations have contributed to the thoughts expressed here. Some of the contributors (in alphabetical order) include: Ted Anderson (IBM Almaden), Cameron Bahar (Storage Machines), Arun Jagatheesan (SDSC), Leo Luan (IBM Almaden), Reagan Moore (SDSC), Manuel Pereira (IBM Almaden), Osamu Tatebe (AIST), Ken Wood (Hitachi Data Systems), Jane Xu (IBM Storage Software Architecture), Alan Yoder (Network Appliance)

Glossary

This glossary provides an introduction some of the terms, concepts and jargons used by this community. The objective of this glossary is not be provide definitions but a detailed introduction to these terms and prepare the reader.

- Grid Computing (concept)

The definition for Grid Computing keeps on evolving. In this document we describe a “grid” as a coordinated distributed computing infrastructure formed by combining heterogeneous resources from autonomous administrative domains, as an ensemble to support large-scale, resource-intensive, and distributed applications.

- Data Grid (infrastructure)

A data-grid is a *logical unified view,* of a distributed infrastructure that consists of heterogeneous data storage resources from autonomous administrative domains. Data storage middleware create a federated location independent, logical infrastructure namespace that is dynamically spread across multiple administrative domains. Datagrids support shared collections and shared resources between the autonomous administrative domains.

- Data Grid Management System (middleware)

The *data storage middleware* used to create data grid is called “Data Grid Management System (DGMS)”. DGMS consists of multiple servers (software) that run on top of storage systems to create the data grid infrastructure. DGMS would consist of vendor specific additional data storage capabilities to manage the data grid. DGMS is a superset of support and possibly extend the standard Grid File System interface. DGMS might also have other interfaces or access mechanisms apart from the standard GFS interfaces.

- Logical Resource Namespace

Namespace as mentioned here implies a hierarchical organization of names representing digital entities. An example of a hierarchical organization is the traditional file system, where the digital entities are organized recursively as directories and files. In addition to files, the names could represent any digital entity including but not limited to data, service/process, storage or computer resource etc. These digital entities could be owned by different domains that participate in the data grid. The digital entities in this logical resource namespace could have different names and different organization from the underlying names used by the real resources that participate in the data grid. This WG is mostly concerned with the file-oriented view or the organization of unstructured data and the operations associated with it in this logical resource namespace. Other resources (apart from unstructured data) could also be present in this logical resource namespace. In some sections of the document, the logical resource namespace is reference as “GFS Namespace” to further stress on the “file system like” logical organization of unstructured data resources.

- Grid File System (interface)

A Grid File System (GFS) is a software system that supports a *suite of standard interfaces* provided by the GGF GFS-WG. A GFS is an attribute-based logical resource namespace of digital entities that can span across multiple autonomous administrative domains. The GFS logical resource namespace provides a traditional file system like view that allows plug-n-play and organization of heterogeneous data storage resources, inter/intra organizational data, computational nodes and compute processes.

- Storage Virtualization (concept)

Storage Virtualization is an *abstract concept* of bringing together different heterogeneous physical resources into one or more logical volumes that help in data storage management of an enterprise. This concept can be implemented in multiple ways and often vendors use this term from the perspective of their products and existing technologies that are available.

Virtualization has real advantages for organization in managing very large and heterogeneous storage resources.

- Data Virtualization (concept)

This term (or jargon) is assumed to have roots from the data grid community who want to take the abstraction of storage virtualization to a higher level. Data Virtualization is an *abstract concept* of bringing together different heterogeneous physical data and physical storage resources into one or more logical views so that distributed and replicated data appear as a single logical data source in a single data management system. The solution is to have logical data namespace on logical volumes of storage allowing very high levels of flexibility for distributed computing and migration of data storage resources. The physical data and resource names are logical and can be physically changed or moved/migrated without affecting the applications.

- Information Lifecycle Management (concept)

Information Lifecycle Management (ILM) refers to the concept of dynamic and cost-effective management of data placement and data retention to realize the requirements and business policies of an enterprise. ILM solutions include the right combination of hardware and software that migrate data autonomously to use different types of storage resources with respect to the long-term operational cost and the current value of the data. Unlike traditional Hierarchical Storage Management (HSM) solutions, which usually used the data freshness as the attribute in determining data placement, ILM solutions use data value and business policies to determine data placement and retention.

- Shared Collections (concept)

Shared collections denote the concept of having a persistent unique identifier to collectively identify a set of digital entities that may be physically distributed and owned by one or more autonomous administrative domains. In simple terms, this involves having a logical directory in which multiple physically distributed digital items from different organizations are present. Shared Collections enable emerging business requirements in virtual organizations to manage unstructured data from inter/intra organizational collaborations within or outside an enterprise. It must be mentioned that like most of data storage requirements which were observed only a few initially, shared collections are a requirement that is currently felt mostly in academic and very large IT data storage enterprises where sharing of unstructured data is critical to the business. It is not confirmed if this requirement will be seen in other businesses also.

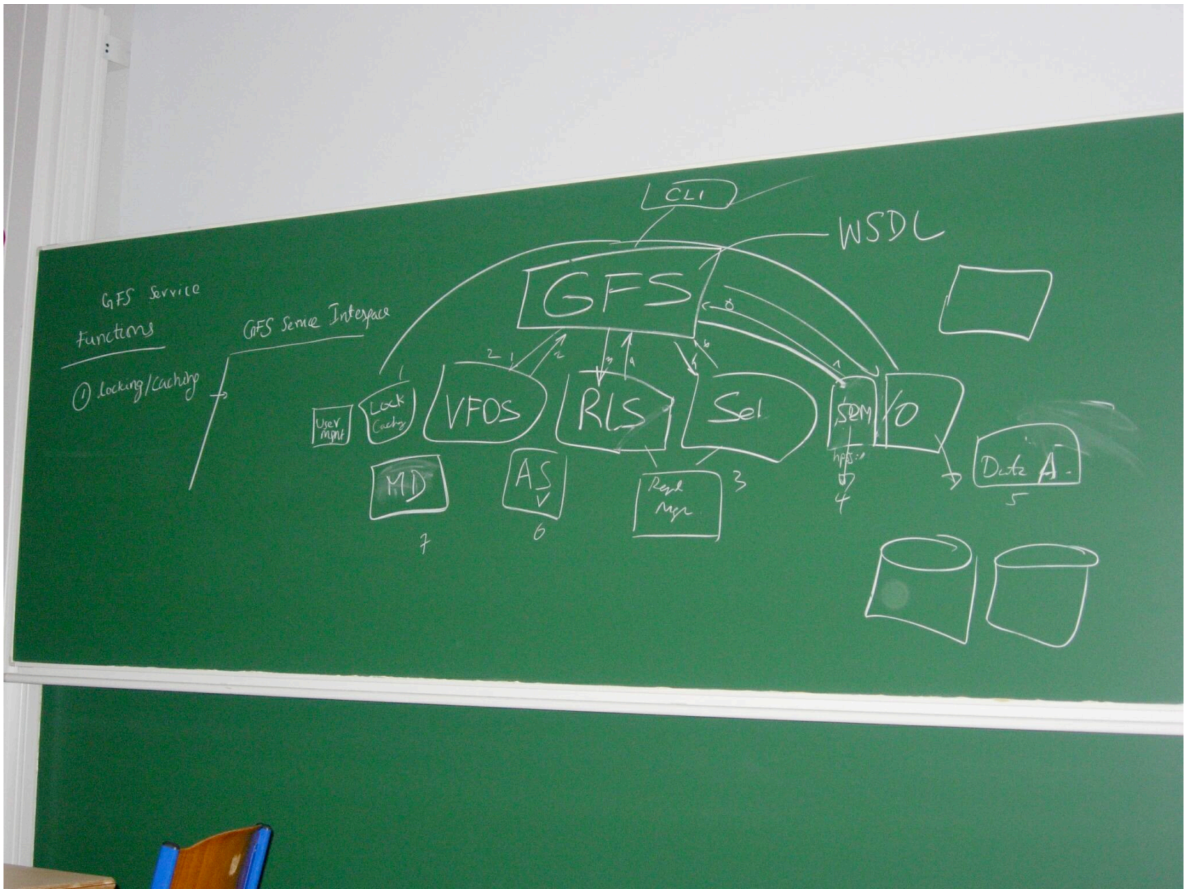
- Logical Resource (concept)

This concept involves representation of the one or more physical resources into one or more logical resources. Each logical resource is a logical view as though it were a physical resource. The users of the logical resource are provided an abstraction of using a physical resource dedicated to them.

Intellectual Property Statement

Appendix

1. GFS Blackboard Picture: The following diagram represents the need to have a single interface for GFS with the following core functionalities. It was drawn (collaboratively) by the members at the GFS-WG session at GGF12 in Brussels.


CLI
WSDL
GFS
GFS Service functions
GFS Service Interface
① Locking/Caching
VFOS
RLS
MD
AS