A Transparent Grid Filesystem

February 26, 2006

Brian Coghlan (coghlan@cs.tcd.ie), Geoff Quigley (gquigle@cs.tcd.ie), Soha Maad (soha.maad@cs.tcd.ie), John Ryan (john.p.ryan@cs.tcd.ie), Eamonn Kenny (Eamonn.Kenny@cs.tcd.ie), and David O'Callaghan (david.ocallaghan@cs.tcd.ie)

Affiliation: Department of Computer Science, Trinity College Dublin

Keywords
Grid Filesystem, Data management, Interoperability

Abstract

Here we argue that the existence of a transparent grid filesystem greatly simplifies the user's experience of grid middleware, particularly regarding data management. We focus on the EGEE grid environment, but we assert that the arguments are substantively true for arbitrary middlewares.

We describe a prototype of a transparent grid filesystem, *gridFS*, with basic but relatively complete functionality provided by four abstract engines: directory, discovery, data movement, and consistency engines. This grid filesystem can be deployed on every node of EGEE, Globus, or any other existing middleware and even on user's workstations. The grid filesystem is specifically intended to support interoperability between arbitrary middlewares.

The data movement engine consists of the basic client and server sides of a filesystem. Its functionality is focussed on accessing various types of data storages, and operation at the block rather than at file level.

The server side of the data movement engine exploits an enhanced version of the GridSite [26] module for the Apache webserver [27]. GridSite includes very desirable features including authentication and GACL [15] authorisation at each directory level, and supports convenient graphical editing of these permissions. Each directory contains a XML file that defines the permissions, conditioned by host, VO or person, based on the Globus Security Infrastructure (GSI) [28].

The client side of the data movement engine uses block-level cacheing to minimise traffic and support consistency. It supports a user-specific grid filesystem view. It uses HTTPS to communicate with the server-side, so it can traverse firewalls as easily as any browser can. By operating at block level, it can fractionally read or write remote files, only transferring the blocks in question.

Consistency semantics define the outcome of multiple accesses to a single file. An example case where inconsistency may arise is when several grid users access

the same filesystem and attempt a simultaneous write to a single file. To avoid such a case several consistency models can be adopted, but initially the engine enforces consistency only via write-through and write-back coherency policies.

The directory engine allows the creation and validation of the user view of the grid filesystem namespace. As such, a user can create their own logical view of grid data as a tree of filesystems. The directory engine implements many of the concepts of the Resource Namespace Service (RNS) provided by the Global Grid Forum Grid File System working group. RNS embodies a three-tier naming architecture, where, for example, the first level can represent the user's view of the namespace, the second can represent a globally consistent logical namespace (that may not be very readable by humans), while the third level can represent actual device addresses.

The directory engine is implemented using the Relational Grid Monitoring Architecture (R-GMA) [32, 33, 34] to store published namespace information. Directory engine producers publish namespace information into the database by using SQL INSERT statements, and its consumers retrieve that information from the database by using SQL SELECT statements.

The discovery engine [31] allows metadata to be published and queried as needed to support grid and inter-grid activity. The discovery engine assumes that every machine on the grid is able to export some directories according to given permissions. Like the directory engine, it depends on R-GMA to store published metadata. Discovery engine producers publish metadata into the database, and its consumers retrieve metadata from that database.

We provide example use cases that demonstrate the functionality of the grid filesystem. The way in which files are made available to an application will depend on the application and the middleware. For example, it may be appropriate that they be transferred via sandboxes, or they may belong to a third party, or their location may be in some sense unknown, etc. Nonetheless, the generic sequence of file operations that may be applied to the files is:

1. metadata create & publish

2. discovery, then namespace create & publish, or vice-versa

3. open/read/write/lock/.../close

4. namespace unpublish & delete

5. metadata unpublish & delete

Generally only a subset of these operations will be applied to any one file.

Five use cases are considered, where each involves a different subset of these actions within job submission and execution using the EGEE grid middleware. The use cases are:

Use Case 1 Local files that are passed to the job via sandboxes, i.e. passed by value (names plus contents)

Use Case 2 Files with known paths that are passed to the job via file catalogs, where the file catalog handles are passed to the job as arguments, i.e. passed by reference (names only)

Use Case 3 Files with known paths that are passed to the job as arguments, i.e. passed by reference (names only)

Use Case 4 Files with incomplete paths, with programmatic discovery during job execution

Use Case 5 Files with incomplete paths, discovered prior to submission, where the complete paths are passed to the job by reference

These use cases will highlight the simplicity that the use of the grid filesystem brings to job submission and execution. Performance measurements will also be presented for each use case, as well as general benchmarking results.

Key features that we have integrated into the filesystem are simplicity, portability, interoperability, and universal accessibility. The filesystem is constructed of four simple engines. Although it relies on UNIX technology, it is expected that it will be portable across a wide variety of UNIX family derivatives. Its native file processing was specifically designed to support interoperability across arbitrary middlewares. Reliance on a HTTPS transport layer should yield as widespread accessibility as is available to browsers. It is expected that this new filesystem will encourage further innovation to assist the increasing data management activity at grid and inter-grid levels.

1 References

[**1**] Enabling Grids For E-sciencE, http://www.eu-egee.org/

[**2**] Globus Project, http://globus.org

[**3**] GridPP Project, http://www.gridpp.ac.uk/

[**4**] Grid Data Farm, http://datafarm.apgrid.org/

[**5**] Peter Kunszt, "EGEE gLite Users Guide Overview of Glite Data Management", EGEE-TECH-570643-v1.0, 20th March, 2005, https://edms.cern.ch/document/570643.

[**6**] ELFI File System, http://www.egrid.it/sw/elfi/index_html

[**7**] INFN, http://www.infn.it/indexen.php

[**8**] LHC Computing Grid, http://lcg.web.cern.ch/LCG/

[**9**] FUSE, Filesystem in User space, http://fuse.sourceforge.net/

[**10**] The Linux Virtual Filesystem Layer, http://www.cse.unsw.edu.au/˜neilb/oss/linux-commentary/vfs.html

[**11**] Globus Toolkit, http://www.globus.org/toolkit/

[**12**] A. McNab, "SlashGrid - a framework for Grid-aware filesystems", Storage Element Workshop, CERN, 29th January 2002.

[**13**] Slashgrid, http://www.gridsite.org/slashgrid/

[**14**] CODA, http://www.coda.cs.cmu.edu/

[15] GACL, http://www.gridpp.ac.uk/authz/gacl/

[16] 0. Tatebe, N. Soda, Y. Morita, S. Matsuoka, S. Sekiguchi, "Gfarm v2: A Grid file system that supports high-performance distributed and parallel data computing," Proceedings of the 2004 Computing in High Energy and Nuclear Physics (CHEP04), Interlaken, Switzerland, September, 2004.

[17] M. Pereira, O. Tatebe, L. Luan, T. Anderson, J. Xu, "Resource Namespace Service Specification", GFS-WG, November, 2005.

[18] K. Czajkowski, F. D. Ferguson, I. Foster, J. Frey, S. Graham, I. Sedukhin, D. Snelling, S. Tuecke, W. Vambenepe, "The WS-Resource Framework", Version 1.0, March, 2004. http://www.oasis-open.org/committees/download.php/6796/ws-wsrf.pdf.

[19] GGF Grid File System Working Group (GFS-WG), http://phase.hpcc.jp/ggf/gfs-rg/

[20] "The GGF Grid File System Architecture Workbook", Version: 0.54, 3rd August, 2005.

[21] Global File System, http://www.deisa.org/organisation/global_filesystems.php

[22] TERAGRID, http://www.teragrid.org/

[23] DEISA, http://www.deisa.org/

[24] S. Maad, B. Coghlan, G. Pierantoni, E. Kenny, J. Ryan, R. Watson, "Adapting the Development Model of the Grid Anatomy to meet the needs of various Application Domains", Cracow Grid Workshop (CGW'05), Cracow, Poland, November, 2005.

[25] G. Pierantoni, O. Lyttleton, D. O'Callaghan, G. Quigley, E. Kenny, B. Coghlan, "Multi-Grid and Multi-VO Job Submission based on a Unified Computational Model", Cracow Grid Workshop (CGW'05), Cracow, Poland, November, 2005.

[26] GridSite, http://www.gridsite.org/

[27] Apache web server, http://httpd.apache.org/

[28] Grid Security Infrastructure, GSI, http://www.globus.org/security/overview.html

[29] S. Maad, B. Coghlan, J. Ryan, E. Kenny, R. Watson, G. Pierantoni, The Horizon of the Grid For E-Government, Proceedings of the eGovernment Workshop, ISBN 1-902316-46-0, Brunel University, United Kingdom, September 2005.

[30] M. Bar, "Linux File Systems", McGraw-Hill Companies; Book & CD edition, 27th July, 2001, ISBN: 0072129557

[31] S. Maad, B. Coghlan, G. Quigley, J. Ryan, G. Pierantoni, E. Kenny, Discovery of Grid Files: Towards a Complete Grid File System Functionality, submitted to The First International Conference on Grid and Pervasive Computing (GPC'2006), Tunghai University, Taichung, Taiwan, May 3-5, 2006

[**32**] S. Fisher, "Relational Model for Information and Monitoring", GGF, 2001, http://www-didc.lbl.gov/GGFPERF/GMA-WG/papers/GWD-GP-7-1.pdf

[**33**] B. Coghlan, A. Djaoui, S. Fisher, J. Magowan, M. Oevers, "Time, Information Services and the Grid", Proc.BNCOD 2001 - Advances in Database Systems, edited by O'Neill, K.D., and Read, B.J., RAL-CONF-2001-003, Oxford, July, 2001.

[**34**] R-GMA, http://www.r-gma.org/

[**35**] A. Chervenak, I. Foster, C. Kesselman, C. Salisbury, S. Tuecke., "The Data Grid: Towards an Architecture for the Distributed Management and Analysis of Large Scientific Datasets", Journal of Network and Computer Applications, 23:187-200, 2001 (based on conference publication from Proceedings of NetStore Conference 1999).

[**36**] D. Cameron, J. Casey, L. Guy, P. Kunszt, S. Lemaitre, G. McCance, H. Stockinger1, K. Stockinger, G. Andronico, W. Bell, I. Ben-Akiva, D. Bosio, R. Chytracek, A. Domenici, F. Donno, W. Hoschek, E. Laure, L. Lucio, P. Millar, L. Salconi, B. Segal, M. Silander, "Data Management Services in the European DataGrid Project", Proc.UK e-Science All Hands Conference, August, 2004.

[**37**] The Globus Resource Specification Language (RSL) Specification 1.0, http://www-fp.globus.org/gram/rsl_spec1.html

[**38**] The EDG Job Description Language (JDL), http://server11.infn.it/workload-grid/docs/DataGrid-01-TEN-0142-0_2.pdf

[**39**] The RFIO protocol, http://doc.in2p3.fr/doc/public/products/rfio/rfio.html