MAPFS-DAI, an extension of OGSA-DAI based on a parallel file system

Alberto Sánchez [a], María S. Pérez [a], Konstantinos Karasavvas [b], Pilar Herrero [a], Antonio Pérez [a]

[a] *Department of Computer Architecture and Technology, Technical University of Madrid, Madrid, Spain, {ascampos, mperez, pherrero, aperez}@fi.upm.es*

[b] *National e-Science Centre, e-Science Institute, Edinburgh, UK, kostas@nesc.ac.uk*

Abstract

Since current applications demand access to a huge volume of data, new and sophisticated I/O systems are required. Several mass storage systems have been developed from different institutions to access their own data repositories. These systems expose native interfaces not interoperable among them. In order to deal with this requirement, OGSA-DAI has emerged to provide a uniform access to data sources. However, this initiative is not focused on the performance of the I/O operations. This paper describes MAPFS-DAI, an approach which tries to combine both ideas: performance and interoperability.

Key words: Data access on grids, OGSA-DAI, MAPFS-Grid, Interoperability, Performance.

1 Introduction

A large number of applications must be able to "consume" huge volumes of data in order to improve their behaviour. Therefore, data management and access are taking a relevant role in current applications and systems.

Although the efficiency of accessing to data is a key factor, addressed by innovative I/O systems, which will be analysed later on, the interoperability between different data sources and the need of "standard" interfaces to access data are causing more and more interest on current solutions. Particularly, in grid environments this demand has originated the proliferation of specifications such as WS-DAI (*Web Services Data Access and Integration*), whose

main goal is to define a *"... specification for a collection of generic data interfaces that can be extended to support specific kinds of data resources ..."* [4].

OGSA-DAI [22] is intented to provide a reference implementation of WS-DAI. OGSA-DAI project has as main goal to provide a uniform access to data resources, being compliant with OGSA [15]. OGSA-DAI has been released on GT3, Axis, the OMII_1 infrastructure and on GT4, based on different specifications, such as OGSI, WS-I, WS-I+ [6] and WSRF [29] respectively.

In spite of all the benefits given by OGSA-DAI, its performance can be enhanced. In fact, in [5] it is stated that *"We expect to invest significant effort in engineering good performance..."*.

In order to tackle the performance and the uniformity of the I/O access, this paper presents MAPFS-DAI, an OGSA-DAI compliant infrastructure, which is based on a parallel I/O system for grids, namely MAPFS-Grid. The outline of this paper is as follows. Section 2 describes the problem which MAPFS-DAI tries to solve and presents related work. Section 3 shows MAPFS-DAI as a standard and optimised interface to data resources in a grid. Section 4 describes the results obtained for evaluating MAPFS-DAI. Finally, section 5 summarizes our conclusions and suggests future work.

2 Problem Statement and Related Work

In short, the problem stated by this paper is an extension of OGSA-DAI, in order to provide two features:

(1) Facilitating uniform access to data resources by means of a service-oriented architecture. With this aim, OGSA-DAI [22] was chosen, since it will provide a reference implementation of WS-DAI.
(2) Improving the performance provided by OGSA-DAI, through the use of a parallel I/O system suitable for grid environments, that is, MAPFS-Grid [33].

Both approaches are combined in order to produce MAPFS-DAI. The feasibility of this combination is guaranteed because of the following common characteristics between OGSA-DAI and MAPFS-Grid:

(1) They both are OGSA compliant [15]. OGSA-DAI supports WS-I, WS-I+ and WSRF [29]. MAPFS-Grid supports WSRF. For this reason, we will use WSRF as the common grid specification for MAPFS-DAI.
(2) OGSA-DAI can be extended, adding new functionalities and activities. Moreover, OGSA-DAI provides a flexible way to add new data resources.

Although OGSA-DAI is mainly intended for relational or XML databases, users can provide additional functionalities. MAPFS-DAI gives support for accessing flat files in a efficient fashion. OGSA-DAI allows files to be accessed on grids, but up to this point, they are focused on file formats such as OMIM, SWISSPROT and EMBL, and not on performance.

(3) Both OGSA-DAI and MAPFS-Grid rely on the factory pattern for the creation of grid services associated to data.

Before analysing our proposal in more depth, it is important to describe some of the most relevant approaches on data grids. This description is made in the next section.

2.1 Data Grids

Data Grids [13,8,20] are grids where the access to distributed data resources and their management are treated as first-class entities along with processing operations. Data Grids, therefore primarily deal with providing services and infrastructure for distributed data-intensive applications. The fundamental features of Data Grids are provision of a secure, high-performance transfer protocol for transferring large datasets and a scalable replication mechanism for ensuring distribution of data on-demand. In order to get the maximum benefits of the infrastructure, some requirements are needed:

- ability to search through numerous available datasets for those that are required;
- ability to select suitable computational resources to perform data analysis;
- ability to manage access permissions;
- intelligent resource allocation and scheduling.

Some studies have investigated and surveyed Data Grids. Bunn and Newman provide in [11] a survey of projects in High Energy Physics, while Qin and Jiang [37] produce a compilation that concentrates more on the consistent technologies. Moore and Merzky [27] identify functional requirements and components of a persistent archival system. In the same way thousand of projects are currently being developed for different application purposes and objectives, such as create and maintain a data movement and analysis infrastructure for users [23,2], create a uniform common/integrated/scalable grid infrastructure that provides computational and storage facilities for scientific research [14,19,36,16,9], access diverse observation and simulation archives through integrated mechanisms [21,7] or even to foster collaboration in biomedical science through sharing of data [10].

In all these projects data sources are generally mass storage systems from which data is transferred as files or datasets to other repositories. Thus, both

storage and data management play an important role in the applications of grid technology.

The earliest applications were complex analysis, often traversing large data sets. Examples of this include protein folding, semiconductor manufacturing, energy exploration research and analysis of recorded DNA sequences. The datasets for all these applications are normally static in nature.

Data management of the static data sets is handled via common data storage techniques such as file systems. The problem to be solved is how to take the data stored in files, typically large in size and move the data to the nodes in the grid that require the data in order to perform the designed tasks. Most common methods used today are File Transfer Protocol (FTP), GridFTP [18], as a protocol engine for data management within the grid, and distributed file systems. Since these address the movement of static data sets, Di Stefano [41] considers them as the first level of Data Grid. They do no address data management issues such as updates, transactions, or integration with external systems. The first example was found in Chervenak et al. [13] where two basic services (storage systems and data management) were introduced in data grid. For Moore at al. [28], data grid provides a uniform name space across the underlying store systems, and is also used to support projects as diverse as digital libraries (National Library of Medicine Visible Embryo project), federation of multiple astronomy sky surveys (NSF National Virtual Observatory project), and integration of distributed data sets (Long Term Ecological Reserve).

The use of Grid Computing has been expanded to also support dynamic data sets (such as those associated to business applications). New data management techniques and infrastructures were required for addressing these kinds of behavioural properties. This second level of Data Grid [41] takes into account both static and dynamic data sets and addresses data management issues associated to the management of this data in the grid, supplying among other things methods of access, management, synchronization, and for transactions. Some examples of the engines that support this level of data grids include JavaSpaces and projects such as OceanStore and OpenMP. JavaSpaces [17] combines persistent stores (tuples) with single operations, creates a shared memory environment that not only exchanges information between distributed processes but also coordinates operations and tasks with each other. JavaSpaces exhibit some of the properties of a distributed management system such as: shared access of data across a network of machines, persistence and transactions. OceanStore [38], a global persistent data store provides a consistent, highly-available, and durable storage utility atop an infrastructure comprised of untrusted servers. OceanStore caches data promiscuously; any server may create a local replica of any data object. These local replicas provide faster access and robustness to network partitions. They also reduce network congestion by localizing access traffic. OpenMP [30] has existed in the industry for

several years. It addresses multithreaded applications as well as the management of interactive data across clusters. Although OpenMP was not designed for grid computing, it provides the splitting of large processing loops into smaller bits of work in a multithreaded, multiprocessor environment. It also creates a distributed memory space to eliminate the use of traditional network communication methods and middleware to allow the threads to share data.

Data sources such as relational databases would become more prominent in future Data Grids. The challenge is to extend the existing Grid mechanisms such as replication, data transfer and scheduling to work with these new data sources. Work in this regard is already being done by projects such as OGSA-DAI [24,25], as we mentioned previously.

The storage Resource Managers (SRM) interface provides a standard uniform management interface to these heterogeneous storage systems, providing a common interface to data grids, abstracting the peculiarities of each particular Mass Storage System. Storage Resource Managers (SRMs) are middleware components whose function is to provide dynamic space allocation and file management on shared storage components on the Grid [40,39]. SRM interface could be used to access the different storage system such as CASTOR ("CERN Advanced STORage manager") [12], a scalable and distributed hierarchical storage management system developed at CERN in 1999. Other mass storage systems which provide a SRM interface are HPSS, Enstore, JASMine, dCache and SE.

Applications have to work between different systems, such as copying data from an archival system located in one organization to a disk drive located in another, having therefore to work across multiple interfaces. The existence of a standard, transparent, uniform, consistent and centralised interface that supports querying, transfer and archiving of data among different systems would simplify the development of Data Grid applications [31].

3 Proposed Approach

Some of the approaches mentioned in the previous section address several aspects related to data grids. Nevertheless, as far as we know, there is not any work which deals with the combination of access uniformity and high performance. This section shows our proposal, whose main aim is the integration of the OGSA-DAI philosophy and our system, MAPFS-Grid, for providing uniform access.

MAPFS-Grid [33] provides a grid-like interface to a parallel file system based on clusters, that is, MAPFS [32]. MAPFS (*Multi Agent Parallel File System*)

has been developed at the Universidad Politécnica de Madrid in 2003. The main contribution of MAPFS is the conceptual use of agents to provide applications with new properties, with the aim of increasing their adaptation to dynamic and complex environments. MAPFS is based on a multiagent architecture that offers features such as data acquisition, caching, prefetching and use of hints [34]. MAPFS is intended to be used in a cluster of workstations, transfering in parallel among all the cluster nodes. On the other hand, MAPFS-Grid allows heterogeneous servers connected by means of a wide-area network to be used as data repositories, by storing data in a parallel way through all the clusters and individual nodes which make up the grid.

The problem of this proposal is that MAPFS-Grid offers a *native* interface, which does not provide interoperability with other I/O systems. This feature is against the principles of a grid environment. For this reason we propose MAPFS-DAI as a bridge between the interoperability and the performance optimization.

3.1 MAPFS-DAI Architecture

According to our goals, MAPFS-DAI architecture must be an extension of the OGSA-DAI architecture [3]. This architecture is divided into four layers (see Figure 1):

(1) Data Layer, which consists of the data resources. The data resources exposed by MAPFS-DAI are flat and unformatted files. Besides, OGSA-DAI gives support to other kind of data resources, such as relational and XML databases.
(2) Business Logic Layer, which is composed of the data service resources and accessors. Concretely, it is necessary to specify: (i) a suitable Data Service Resource, which is named *File Data Service Resource*, associated to flat and unformatted files, and (ii) A new accessor, whose main goal is to control access to the underlying data resource, that is, files. We will name our accessor as *MAPFS-DAI accessor*. The MAPFS-DAI accessor allows activities to access data resources. Activities are the operations performed by data service resources. Currently in MAPFS-DAI, we have 2 activities, one for reading (*FileAccessActivity*) and another one for writing (*FileWritingActivity*), which are compliant with the File Activities defined by OGSA-DAI.
(3) Presentation Layer, which provides the web service interfaces to data services. In the case of MAPFS-DAI, WSRF is used.
(4) Client Layer, which includes client application and client toolkit. This last component makes easier the development of client applications by providing useful and simple tools to create the documents exchanged

between the client and server: perform and response documents. Both documents must fulfill the requirements specified by the service schema. As long as the service schema is supported, it is possible to modify the accessor to access a different storage system, without changing the client application.

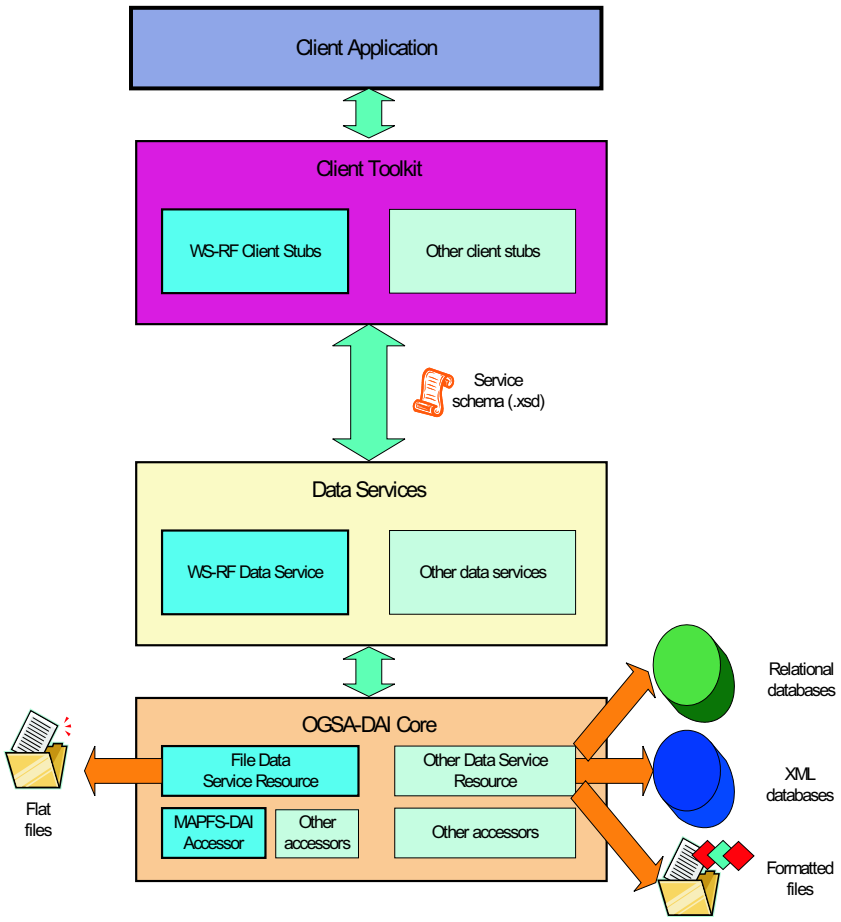

Fig. 1. MAPFS-DAI within the OGSA-DAI Architecture

3.2 Double level of parallelism

By providing a uniform access to our system, the performance is drastically reduced, as we will see in Section 4. For alleviating this decrease, we propose to take advantage of two levels of parallelism, in the same way that MAPFS-Grid does [35]:

(1) The high level provides parallelism among the set of storage elements.
(2) The low level provides parallelism among the set of nodes of each cluster, in the case of using MAPFS-DAI. For performing this action, MAPFS is used.

These two levels are depicted in Figure 2. The interoperability between MAPFS-DAI and other OGSA-DAI compliant systems is also shown. This is an important feature, since every storage element that exhibits the OGSA-DAI interface can be used together with MAPFS-DAI elements. This is the case of the element 1 in Figure 2. New OGSA-DAI compliant elements could be created from mass storage systems (MSSs), such as CASTOR, and incorporated to this scenario. Due to the interoperability provided by OGSA-DAI, different storage systems could be accessed in parallel.

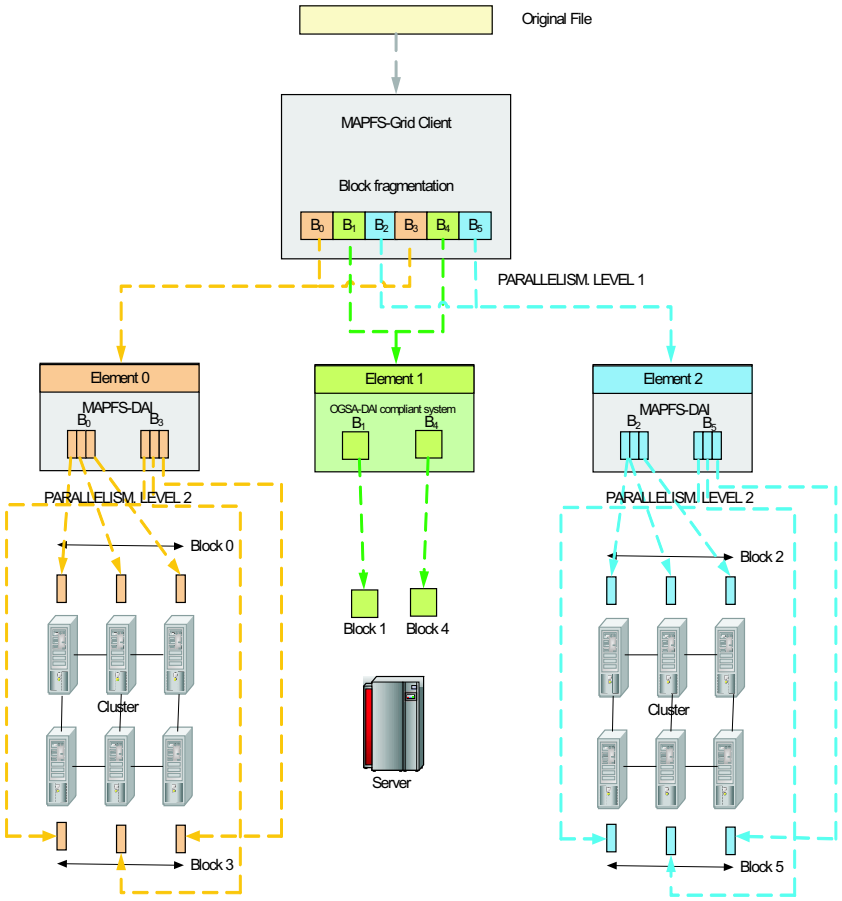

Fig. 2. Double parallelism by using MAPFS-DAI storage elements

4 Performance Analysis

This section shows the results obtained by evaluating our system (MAPFS-DAI) and the comparison of this system against other kind of systems, which allow us to extract some interesting conclusions that assert our previous proposals. Through this analysis, our aim is proving the performance benefits obtained due to the parallel use of a standard and uniform access provided by OGSA-DAI.

In order to validate our proposal, it is necessary to evaluate its performance. Experiments were run in two different clusters. The first one is composed of two Intel Pentium IV 3GHz nodes, with 512 MB of RAM memory, connected by a Gigabit network. The second one has seven Intel Xeon 2.40GHz nodes, with 1GB of RAM memory interconnected by means of a Gigabit network. Both are connected by means of a wide-area network.

Our experiment consists of accessing a file stored in the system, through a common interface (OGSA-DAI), which assists the access and integration of data, located in disparate data sources.

In the first approach we have integrated the MAPFS file system inside OGSA-DAI as an accessor. In this sense it is possible to access the data stored in MAPFS by using the fileAccess interface provided by OGSA-DAI. As it is shown in Figure 3 the total time taken to finish the activity is smaller in MAPFS-DAI than the basic implementation of fileAccess in OGSA-DAI. This is because of the fact that the MAPFS-DAI file access is made in a parallel fashion by using all nodes from the second cluster, whereas OGSA-DAI access is using only one server (the master node of the cluster).

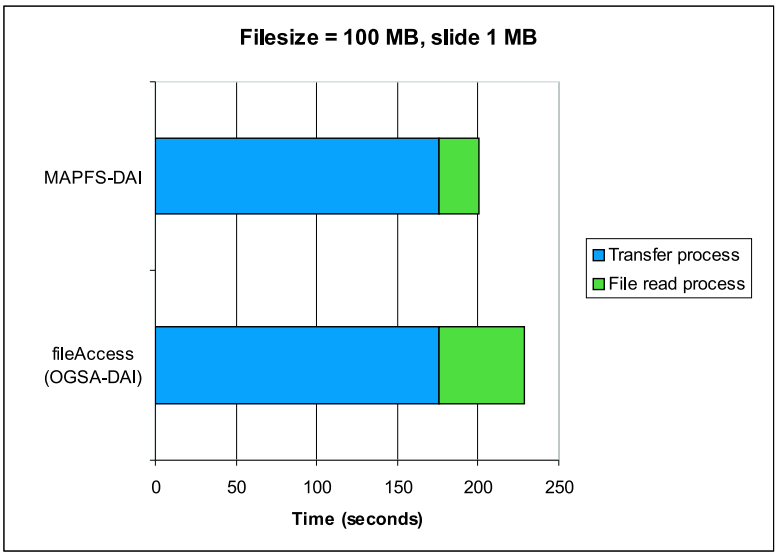

Fig. 3. Comparison between the FileAccess method implemented in OGSA-DAI and MAPFS-DAI to access a 100 MB filesize

However, MAPFS-DAI does not provide a performance-full solution. Although it is not comparable with GridFTP due to their different applications [1], if these results are analysed in relation to the solutions provided by GridFTP and MAPFS-Grid, as is described in [35], an interesting conclusion could be reached: most time is consumed by performing the file transfer because that is based on SOAP, used by OGSA-DAI. In fact, OGSA-DAI interacts with

[1] GridFTP only transfers complete files

data service resources via a document-oriented interface, by sending a perform document. The data service performs the actions described in the document and composes a response document, which is sent back to the client. More or less 85% of the time is used for the SOAP transfer. This means that the problem is not an I/O problem, but a protocol problem (e.g., use of SOAP). In order to enhance the overall system performance, one possibility would be to improve these transport mechanism.

For improving the performance, we have built a parallel client that can use all the different file accessors, because OGSA-DAI is a common interface, as explained in section 3.2. Thus, all the systems or accessors implemented that use the *fileAccess* interface could be used to access data in a parallel way.

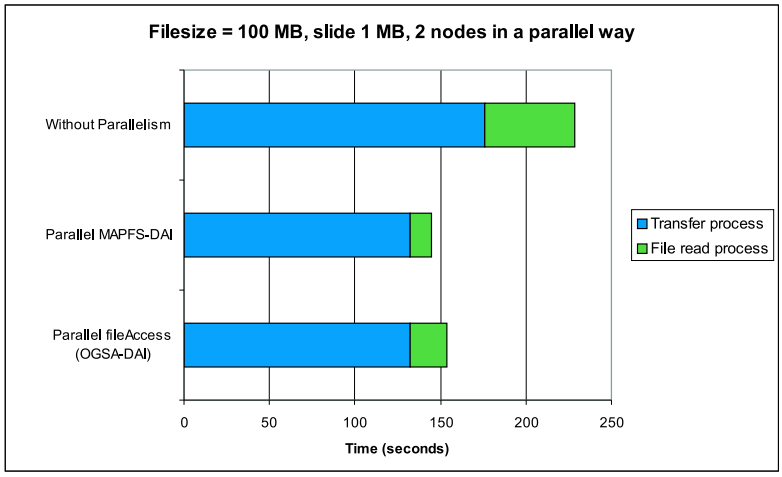

Fig. 4. Comparison between a simple access and parallel access by using the fileAcces method implemented in OGSA-DAI and MAPFS-DAI to access a 100 MB filesize

Figure 4 shows that the total time required to read a file is reduced. The performance improvement is obvious, since both clusters are being used in a parallel fashion through MAPFS-DAI, by taking advantage of both levels of parallelism. Thus, it is possible to reduce the access time limited by the network bandwidth by means of this double parallelism. In the case of a parallel access to OGSA-DAI services, it is possible to take advantage only of the higher level of paralellism.

Furthermore, it is advisable to take into account the fact that different kinds of systems (such as MAPFS-DAI, fileAccess based on OGSA-DAI, CASTOR, and so on), could be used together, in the case of providing the same interface. This feature should improve the performance if we use more servers and could increase the fault tolerance if data replication is implemented within the system.

5 Conclusions and Future Work

This paper describes MAPFS-DAI as a trade-off between uniform access interfaces to data resources and performance enhancement for accessing these resources, by means of the use of double parallelism.

The cost of using a standard interface is high, as we have seen in the performance analysis. Nevertheless, the interoperability of MAPFS-DAI with other OGSA-DAI I/O systems provides the possibility of using more I/O systems in parallel, which alleviates this performance lost. Furthermore, most of the time spent in I/O operations is lost in the data transfer through the network. This problem could be solved by reimplementing the transfer stage of the system according, for instance, to the way in which this problem is treated in GridFTP. This constitutes part of our future research work.

Additionally, we consider how important is to create a new line inside of the WS-DAI working group related to enhanced access to flat files. One of the possible aspects which should be tackled within this line would be the definition of semantics close to the POSIX standard. In OGSA-DAI, the FileAccess activity corresponds to one operation. For MAPFS-DAI, it would be more interesting to define only one activity related to the access to files and within this activity a set of POSIX-like operations (open, read, write, close and so on). This would improve the system performance because, in the current solution, it is required to open and close the file every time a read or write operation is performed. By using the POSIX semantics, a session is established and the file is opened and closed once, independent of the number of I/O operations. This requirement has been investigated in the WS-DAIF specification (Web Services Data Access and Integration - The File Realisation): *"It would be relatively straightforward to devise a bespoke XML schema to express the POSIX commands and parameters, but that is not the function of this specification"*.

References

[1] *13th International Symposium on High-Performance Distributed Computing (HPDC-13 2004), 4-6 June 2004, Honolulu, Hawaii, USA*. IEEE Computer Society, 2004.

[2] Bill Allcock, Ian Foster, Veronika Nefedova, Ann Chervenak, Ewa Deelman, Carl Kesselman, Jason Lee, Alex Sim, Arie Shoshani, Bob Drach, and Dean Williams. High-performance remote access to climate simulation data: a challenge problem for data grid technologies. In *Supercomputing '01: Proceedings of the 2001 ACM/IEEE conference on Supercomputing*, pages 46–46, New York, NY, USA, 2001. ACM Press.

[3] M. Antonioletti, M. Atkinson, R. Baxter, A. Borley, N. Chue Hong, P. Dantressangle, A. Hume, M. Jackson, A. Krause, S. Laws, P. Parson, N. Paton, J. Schopf, T. Sugden, P. Watson, and D. Vyvyan. OGSA-DAI status and benchmarks. In *Proceeedings of UK e-Science All Hands Meeting*, 2005.

[4] M. Antonioletti, M. Atkinson, S. Laws, S. Malaika, N. Paton, D. Pearson, and G. Riccardi. Web services data access and integration (WS-DAI). *13th Global Grid Forum*, 2005.

[5] M. Atkinson, K. Karasavvas, M. Antonioletti, R. Baxter, A. Borley, N. Chue Hong, A. Hume, M. Jackson, A. Krause, S. Laws, N. Paton, J. Schopf, T. Sugden, K. Tourlas, and P. Watson. A new architecture for OGSA-DAI. *AHM*, 2005.

[6] Malcolm P. Atkinson, David De Roure, Alistair N. Dunlop, Geoffrey Fox, Peter Henderson, Anthony J. G. Hey, Norman W. Paton, Steven Newhouse, Savas Parastatidis, Anne E. Trefethen, Paul Watson, and Jim Webber. Web service grids: an evolutionary approach. *Concurrency - Practice and Experience*, 17(2-4):377–389, 2005.

[7] Australian Virtual Observatory, accessed dec 2005 [online]. available: *http://www.aus-vo.org*.

[8] P. Avery. Data grids: a new computational infrastructure for data-intensive science. *Philosophical Transactions of the Royal Society of London Series a-Mathematical Physical and Engineering Sciences*, 360(1795):1191–1209, 2002.

[9] BioGrid japan, accessed dec 2005 [online]. available: *http://www.biogrid.jp*.

[10] Biomedical Informatics Research Network (BIRN), accessed dec 2005 [online]. available: *http://www.nbirn.net*.

[11] J. Bunn and H. Newman. Data-intensive grids for high-energy physics. *Grid Computing: Making the Global Infrastructure a Reality*, 2003.

[12] The Castor Project, *http://www.castor.org*.

[13] A. Chervenak, I. Foster, C. Kesselman, C. Salisbury, and S. Tuecke. The data grid: towards an architecture for the distributed management and analysis of large scientific datasets. *Network and Computer Applications*, 23:187–200, 2001.

[14] Enabling Grids for E-sciencE (EGEE), accessed dec 2005 [online]. available: *http://public.eu-egee.org*.

[15] I. Foster et al. The Open Grid Services Architecture, version 1.0. 2005.

[16] Ian T. Foster et al. The grid2003 production grid: Principles and practice. In *HPDC* [1], pages 236–245.

[17] David Gelernter and Arthur J. Bernstein. Distributed communication via global buffer. In *PODC '82: Proceedings of the first ACM SIGACT-SIGOPS symposium on Principles of distributed computing*, pages 10–18, New York, NY, USA, 1982. ACM Press.

[18] GridFTP: Universal Data Transfer for the grid, *http://www.globus.org/datagrid/gridftp.html*.

[19] Grid Physics Network (GriPhyN), accessed dec 2005 [online]. available: *http://www.griphyn.org*.

[20] W. Hoschek, F. J. Jaen-Martinez, A. Samar, H. Stockinger, and K. Stockinger. Data management in an international data grid project. In *Proceedings of the First IEEE/ACM International Workshop on Grid Computing(GRID 2000)*, December 2000.

[21] International Virtual Observatory Alliance, accessed dec 2005 [online]. available: *http://www.ivoa.net*.

[22] Konstantinos Karasavvas, Mario Antonioletti, Malcolm P. Atkinson, Neil P. Chue Hong, Tom Sugden, Alastair C. Hume, Mike Jackson 0003, Amrey Krause, and Charaka Palansuriya. Introduction to ogsa-dai services. In Pilar Herrero, María S. Pérez, and Víctor Robles, editors, *SAG*, volume 3458 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 2004.

[23] Lhc computing grid, accessed dec 2005 [online]. available: *http://lcg.web.cern.ch/LCG*.

[24] James Magowan. A view on relational data on the grid. In *IPDPS*, page 90. IEEE Computer Society, 2003.

[25] Susan Malaika, Andrew Eisenberg, and Jim Melton. Standards for databases on the grid. *SIGMOD Record*, 32(3):92–100, 2003.

[26] Beniamino Di Martino, Jack Dongarra, Adolfy Hoisie, Laurence Tianruo Yang, and Hans Zima, editors. *Engineering the Grid: Status and Perspective*. American Scientifc Publisher, January 2006.

[27] R. Moore and A. Merzky. Persistent archive concepts. *Global Grid Forum Persistent Archive Research Group, Global Grid Forum 8*, June 2003.

[28] Reagan W. Moore, Igor Terekhov, Ann Chervenak, Scott Studham, Chip Watson, and Heinz Stockinger. Data grid implementations. *available at http://www.ppdg.net/docs/WhitePapers/Capabilities-grids.v6.pdf*, January 2002.

[29] OASIS. Ws-ResourceFramework. 2005.

[30] OpenMP, accessed dec 2005. Available: *http://www.openmp.org*.

[31] Laura Pearlman, Carl Kesselman, Sridhar Gullapalli, B. F. Spencer Jr., Joe Futrelle, Kathleen Ricker, Ian T. Foster, Paul Hubbard, and Charles Severance. Distributed hybrid earthquake engineering experiments: Experiences with a ground-shaking grid application. In *HPDC* [1], pages 14–23.

[32] María S. Pérez, Jesús Carretero, Félix García, José M. Peña Sánchez, and Victor Robles. A flexible multiagent parallel file system for clusters. In Peter M. A. Sloot, David Abramson, Alexander V. Bogdanov, Jack Dongarra,

Albert Y. Zomaya, and Yuri E. Gorbachev, editors, *International Conference on Computational Science*, volume 2660 of *Lecture Notes in Computer Science*, pages 248–256. Springer, 2003.

[33] María S. Pérez, Jesús Carretero, Félix García, José M. Peña Sánchez, and Victor Robles. MAPFS-Grid: A flexible architecture for data-intensive grid applications. In F. Fernández Rivera, Marian Bubak, A. Gómez Tato, and Ramon Doallo, editors, *European Across Grids Conference*, volume 2970 of *Lecture Notes in Computer Science*, pages 111–118. Springer, 2003.

[34] María S. Pérez, Alberto Sánchez, Víctor Robles, José M. Peña Sánchez, and Fernando Pérez. Optimizations based on hints in a parallel file system. In Marian Bubak, G. Dick van Albada, Peter M. A. Sloot, and Jack Dongarra, editors, *International Conference on Computational Science*, volume 3038 of *Lecture Notes in Computer Science*, pages 347–354. Springer, 2004.

[35] M.S. Pérez, A. Sánchez, P. Herrero, and V. Robles. *A New Approach for overcoming the I/O crisis in grid environments.* American Scientific Publisher, 2006. Article belonging to [26].

[36] Particle Physics Data Grid(PPDG), accessed dec 2005 [online]. available: *http://www.ppdg.net.*

[37] X. Qin and H. Jiang. Data grids: Supporting data-intensive applications in wide area networks. Technical Report TR-03-05-01, May 2003.

[38] S. Rhea, P. Eaton, D. Geels, H. Weatherspoon, B. Zhao, and J. Kubiatowicz. Pond: The oceanstore prototype. In *Proceedings of the Conference on File and Storage Technologies.* USENIX, 2003.

[39] Arie Shoshany et al. SRM Interface Specification v.2.1, *http://sdm.lbl.gov/srm-wg/doc/srm.spec.v2.1.final.pdf.*

[40] Arie Shoshany et al. SRM Joint Design v.1.0, *http://sdm.lbl.gov/srm-wg/doc/srm.v1.0.pdf.*

[41] Michael Di Stefano. *Distributed Data Management for Grid Computing.* Wiley-Interscience, July 2005.