Data Management for Grid Environments

Heinz Stockinger[1], Omer F. Rana[2], Reagan Moore[3], and Andre Merzky[4]

[1] CERN, Switzerland, heinz.stockinger@cern.ch
[2] Cardiff University, UK, o.f.rana@cs.cf.ac.uk
[3] San Diego Supercomputer Center, USA, moore@sdsc.edu
[4] Konrad Zuse Zentrum, Berlin, Germany, merzky@zib.de

Abstract. An overview of research and development challenges for managing data in Grid environments is provided. We relate issues in data management at various levels of abstraction, from resources storing data sets, to metadata required to manage access to such data sets. A common set of services is defined as a possible way to manage the diverse set of data resources that could be part of a Grid environment.

1 Foundations

To identify data management needs in Grid environments an assessment can be based on existing components associated with data management, and one can view the Grid as integrating these. In this approach, the emphasis lies on providing interfaces between existing data storage and manipulation systems, to enable systems from different vendors and research groups to work seamlessly. The alternative approach is based on assessing application needs and requirements, and identifying missing functionality. We try to take a middle ground between these two approaches, and identify a set of common services, that may be suitable for both. At present, there are at least three communities that require access to distributed data sources: (1) Digital libraries (and distributed data collections). Digital libraries provide services for manipulating, presenting, discovering, browsing, and displaying digital objects. (2) Grid environments for processing distributed data, with applications ranging from distributed visualisation, to knowledge discovery and management. (3) Persistent archives for maintaining collections while the underlying technology changes. In this context, one must be able to deal with legacy systems that maintain such archives, and enable the migration, or wrapping, of these systems as new technology becomes available.

Hence, an architecture that is used to support Grid based environments should be consistent with the architectures needed to support digital libraries and persistent archives. An important common theme in all of these communities is the need to provide a uniform Application Programming Interface (API) for managing and accessing data in distributed sources. Consequently, specialised operations are needed to manage and manipulate digital objects with varying degrees of granularity. A digital object may be stored in a file system, as an object in an object-oriented database, as a Binary Large OBject (BLOB) in an

object-relational database, or as a file in an archive, and should still utilise a common API. Hence, the concept of a data handling system that automates the management of digital objects stored in distributed data sources is the key concept in managing data in Grid environments.

Our objective is to identify a taxonomy for data management in scientific computing, to elicit requirements of applications that utilise Grid infrastructure, and to provide a means of classifying existing systems. The paper aims to complement work identified in [1], and identify building blocks that could be used to implement data management functions. The criteria needed for managing data in Grid applications are outlined, and based on the notion of (1) local services that must be provided by a given storage resource, (2) global services that need to be provided within a wider context to enable a better sharing of resources and data. It is intended that each data management and storage resource must subscribe to a global service, and must support some or all of the APIs required by such global services. Hence, a data storage resource is said to be 'Grid-enabled' if it can access and interact with these global services. We see two important considerations that distinguish current usage of data storage resources from Grid-enabled use – 'diversity' in operations and mechanisms, and 'performance' tolerance across mechanisms and resources.

2 Data Management - A Wider Perspective

The ability to process and manage data involves a number of common operations, the extent of which depends on the application. These operations include: *Data Pre-Processing and Formating* for translating raw data into a form that can be usefully analysed. Data processing may involve transforming a data set into a pre-defined range (for numeric data), and identifying (and sometimes filling in) missing data, for instance. The data processing stage is generally part of the 'data quality' check, to ensure that subsequent analysis of the data will lead to meaningful results. For numerical data, missing data may be handled using statistical techniques, such as an average value replacing a missing element. Metadata is often used in this context, for translating data from one form to another. Metadata can correspond to the structure of a data source, such as a database schema, which enables multiple data sources to be integrated. Alternatively, metadata may be summary data which identifies the principle features of the data being analysed, corresponding to some summary statistics. Generally, summary statistics have been generated for numeric data, however extensions of these approaches to data that is symbolic is a useful current extension.

Data Fusion for combining different types of data sources, to provide a unified data set. Data fusion generally requires a pre-processing stage as a necessity, in order for data generated by multiple experiments to be efficiently integrated. An alternative to fusion is Data Splitting, where a single data set is divided to facilitate processing of each sub-set in parallel, possibly though the use of filters which extract parts of the original data set based on pre-defined criteria.

Data Storage involves the recording of data on various media, ranging from disks to tapes, which can differ in their capacity and 'intelligence'. Data stor-

age can involve data migration and replication between different storage media, based on a Hierarchical Storage Management (HSM) system, which vary based on access speed to storage capacity. As regards replication, large amounts of data might be transferred over the network (local or wide area) which imposes particular problems and restrictions. Specialised applications, such as scientific visualisation, require specialised data storage to enable data to be shuffled between the application program and secondary (or even tertiary) storage. Data storage hardware and software also differ quite significantly. Hardware resources can include RAID drives, where support is provided for stripping data across multiple disks, and parity support to ensure that lost data can either be reconstructed, or migrated when a disk fails. Large scale data storage units include HPSS (from IBM) and products from FileTek and AMPEX.

Data Analysis can range from analysing trends in pre-recorded data for hypothesis testing, to checking for data quality and filling in missing data. Data analysis is an important aspect of data management, and has been successfully employed in various scientific applications. Analysis approaches can range from evolutionary computing approaches such as neural networks and genetic algorithms, rule based approaches based on predicate/propositional logic to Case Based Reasoning systems, to statistical approaches such as regression analysis. The data analysis approach generally requires a prior data preparation (preprocessing) stage.

Query Estimation and Optimisation is essential if the data analysis is done on large amounts of data in a multi-user environment. Based on the input gained by a single user query, an estimate for how long it takes to transfer the required data to the computational unit may be made.

Visualisation, Navigation and Steering is the emerging area within data management that can range in complexity from output display on desktop machines, to specialised visualisation and (semi-) immersive environments such as ImmersaDesk and CAVE. Visualisation tools such as IRIS Explorer/Data Explorer have been widely used in the scientific community, and provide a useful way to both generate new applications, and for visualising the results of these applications. The next stage - providing computational steering support, will enable scientists to interact with their simulation in real time, and dynamically 'steer' the simulation towards a particular parameter space. Visualisation therefore becomes an enabler in creating and managing new types of scientific experiments, rather than as a passive means for viewing simulation output.

Data management is therefore a unified process that involves a number of stages, and it is important to view it as a whole. Each individual stage within the process has its own family of products and algorithms. To enable Grid-enabled devices to utilise these services, it is important to distinguish services between (1) management services, and (2) support and application services. Management services relate to operations and mechanisms offered within each storage resource, and global services with which the resource interacts - these are identified in Section 3. Support and application services relate to higher level operations which undertake correlations or aggregations on the stored data. These can be imple-

mented in different ways, and are based on particular application needs and user preferences. To initiate a discussion, we identify some categories of such services in Section 4.

3 Support for Data Storage and Access

Support for data management can be divided into a number of services, each of which may be offered within a single system, or may constitute an aggregate set of operations from multiple systems, by multiple vendors. The primary objective in the context of Grid based systems is to support device and vendor heterogeneity, subject to some additional set of constraints, generally related to performance - which might not hold for data intensive high throughput applications. The criteria for categorising storage devices are: *Policy* is related to the division of services that should be directly supported within a storage device, and those that are required to be implemented by a user. The objective in many cases would be to provide a large number of services directly within the storage device. However, this may not be practical or useful from other perspectives, such as access or search time. A compromise is required between functionality and performance/throughput, and the Storage Policy should identify this. As far as possible, the Policy should be exposed to the user, and design decisions undertaken by a manufacturer should be made clear.

Operations identify the kinds of services that are required as a minimum within every Storage resource. These can be limited to 'read' and 'write' operations, or may also include additional services such as 'address lookup', 'access properties' (such as size of file, transfer rate), and 'error handling'. In each case, a minimal subset should be supported and identified. This 'Operations' set is particularly relevant when dealing with heterogeneity of the storage devices.

State Management relates to support for transactions and failure in a storage resource. Hence, a service to maintain and manage the state of a resource is important to enable the resource to be re-started in case of failure. State management can be undertaken within every resource, or a standard service for check-pointing the state of a resource may be provided.

Mechanism identifies how the operations supported within a storage resource are actually implemented. Each resource may implement read and write operations in a different way, and in some cases, this mechanism may need to be exposed to the user. In specialised tape storage devices, such as the ADT from Sony, intelligent mechanisms are provided to maintain a record of head movements during data access. These records may be made accessible to an external user or application, to enable head management by an application. To support Grid applications, it is important that storage resources should enable multiple mechanisms to co-exist, and not rely on the existence of a particular mechanism within a resource.

Errors/Exceptions can relate to a particular resource, or to data management within a group of resources. Hence, error handling may be undertaken locally, and be specific to mechanisms and operations supported within a given resource.

However, each resource should also support error handling at a global level, for groups of resources being utilised within a given application.

Structure can relate to the physical organisation of a data storage resource, or the structure of the contents of the resource. Examples of the former case include number of disks, number of heads, access and transfer rates, and other physical characteristics of the storage resource. Structure of the contents may be specified using a database schema, which defines how content may be accessed or managed. This structure may therefore range from the structure of the file system, to data type based description of the contents. Structure information is important to global services, and for information services which utilise multiple data storage resources.

Accessing and transferring data from secondary and tertiary storage forms an important operation in data management. Generally, the data transfer needs to be undertaken from devices which have different access times, and support different access APIs and mechanisms. Data storage structure may also differ, requiring metadata support for describing this structure, a distinction is made between the structure of a data storage device and its content. This distinction is useful to maintain flexibility in the storage structure and in the managed data, enabling tools from multiple vendors to be used.

3.1 The Minimum Unit and Access Patterns

Access patterns for scientific computing applications are significantly different from business or commercial computing. In business and commercial computing, data access generally involves access to single units of data, often in random order. In scientific computing, access is generally more regular, such as within a loop of a numerical calculation, for instance. However, data access patterns are often very difficult to determine as regards the high throughput applications in HEP. Data structures in scientific high performance applications generally involve bulk data transfers based on arrays. Array accesses can be regular, generally as block, cyclic or block cyclic, or it may be irregular based on irregular strides across an array dimension. Access patterns can be for access to groups of data items, or to a group of files. The unit of transfer is generally determined by the storage device, ranging from single or multiple data items in database management systems such as RasDaMan [13], to file systems such as NFS or AFS, hierarchical storage systems such as the High Performance Storage System (HPSS), and network caches such as the Distributed Parallel Storage System (DPSS). In order to define standardised services, it is also useful to identify the basic unit of data transfer and access. This unit is dependent on the types of information processing services that utilise the stored data. We identify two types of data units: *Primitive types:* A primitive type is a float, integer or character that can be stored within a resource. The unit of storage is dependent on the type of programming language or application that makes use of the storage resource, and the storage medium being employed. Groups of such types may also be considered as primitive types, depending on the storage resource, and include arrays, images or binary objects. *Files:* An alternative unit of storage, not based

on the type of processing or any associated semantics, may be an uninterpreted sequence of bytes – a file. The file may reside in a database or a conventional file system.

3.2 Support for Metadata Management

Metadata could relate to a hierarchical scheme for locating storage resources (such as LDAP), properties of resources that are externally visible, permissions and access rights, and information about the stored content. Developing a consensus on the desired representation for relationships is also important in the context of Grids, although this is not likely to be achieved in the short term. Relationships can have multiple types, including semantic/functional, spatial/structural, temporal/procedural. These relationships can be used to provide semantic operability between different databases, support type conversion between different object oriented systems, and manage ownership of distributed data objects. The ISO 13250 Topic Maps standard (defined below) is one candidate for managing relationships between data sources.

Metadata is also important for tagging attributes of data sets, such as digital objects (the bit streams that represent the data). The emergence of a standard tagging language, XML, has made it feasible to characterise information independently of the data bit stream. XML Document Type Definitions (DTDs) provide a way to structure the tagged attributes. The advantage of XML DTDs is that they support semi-structured organisations of data. This encompasses under one standard representation both unstructured sets, ordered lists, hierarchical graphs, and trees.

Digital objects that are created for undertaking a given analysis, may be re-used within another type of analysis, depending on their granularity. An approach that encapsulates each digital data set into a digital object, and then makes that object a member of an object class is forcing undue constraints. Conceptually, one should be able to manage the attributes that are required to define an object independently from the bits comprising the data set. This means it is possible to build constructors, in which data sets are turned into the particular object structure required by the chosen class. Data sets can be re-purposed if the class attributes are stored in a catalogue. Supporting and maintaining such catalogues then becomes crucial for the re-use of data sets. This requires templates for constructing objects from data sets. An example is the DataCutter system [9]. An XML DTD is used to define the structure of the data set. The DataCutter proxies are designed to read the associated DTD, and then process the data set based upon the structure defined within the DTD. This makes it possible to transform the structure of the data set for use by a particular object class, if the transformation is known to the XML DTD that represents the object class.

3.3 Standards

There are three standards which are of interest for the provision of data storage. One is the IEEE Reference Model for Open Storage Systems Interconnections

(OSSI), previously known as the IEEE Mass Storage Reference Model, and the ISO standard for a Reference Model for Open Archival Information Systems, which is still under development. Whereas the IEEE standard is mainly concerned with architecture, interface and terminology specifications and standards, the ISO standard focuses more on necessary operational issues and interactions between different parts of an data archives. The ISO Topic Maps standard, on the other hand, deals with the contents of a data resource, and deriving an association between terms in the stored data. In this respect both the IEE and the ISO standards can be seen as complementary. The first two of these descriptions are taken from Kleese [6].

IEEE's Open Storage Systems Interconnection (OSSI). This standard started out as the IEEE Mass Storage Reference Model in the 1980s, and is very much focused on technical details of mass storage systems, and contains specifications for storage media, drive technology and data management software. Recently, organisational functions have been added, and the description of connections and interactions with other storage systems have been stressed. Nowadays the Reference Model for OSSI provides a framework for the co-ordination of standards development for storage system interconnection, and provides a common perspective for existing standards. The descriptions used are independent of existing technologies and applications and are therefore flexible enough to accommodate advanced technologies and the expansion of user demands.

ISO's Open Archival Information System (OAIS). The OAIS standard aims to provide a framework for the operation of long term archives which serve a well specified community. Hereby issues like data submission, data storage and data dissemination are discussed. Every function is seen in its entirety, as not only describing technical details, but also human interventions and roles. For the purpose of this standard it has been decided that the information that is maintained needs long-term preservation, even if the OAIS itself will not exist through the whole time span. The OAIS standard addresses the issue of ingestion, encapsulation of data objects with attributes, and storage. OAIS does not, however, address the issue of technology obsolescence. Obsolescence can be handled by providing interoperability support between systems that support the migration onto new technology.

ISO 13250 Topic Maps standard provides a standardised notation for representing information about the structure of information resources used to define topics, and the relationships between topics, to support interoperability. A set of one or more interrelated documents that employs the notation defined by this International Standard is called a 'topic map'. In general, the structural information conveyed by topic maps includes: (1) groupings of addressable information objects around topics (occurrences), and (2) relationships between topics (associations). A topic map defines a multidimensional topic space – a space in which the locations are topics, and in which the distances between topics are measurable in terms of the number of intervening topics which must be visited in order to get from one topic to another, and the kinds of relationships that define the path from one topic to another, if any, through the intervening topics, if any.

4 Support and Application Services

Some communities like HEP are confronted with Grid-enabled data management issues more directly than computing intensive high performance applications that store only a small amount of data compared to their computational effort. Application specific concerns can subsequently be handled by developers directly, or by software libraries specific to a resource. We evaluate two domains of interest in the context of data management for Grid applications, each of which requires data access and analysis from multiple sources, maintained by different authorities, offering different levels of access privileges.

HEP: The HEP user community is distributed almost all around the globe. As for the next generation experiments starting in 2005 at CERN, large computing intensive applications run on several hundreds or even thousand CPUs in different computer centres and produce roughly 1 Petabyte of persistently stored data per year over 10 to 15 years. In particular, the collisions of particles in detectors produce large amounts of raw data that are processed and then transformed into reconstructed data. Once data is stored in disk subsystems and mass storage systems, it has to be replicated to hierarchically organised regional centres. This requires secure and fast migration and replication techniques which can be satisfied with protocol implementations like the GridFTP [12]. Once data is in place, distributed data analysis can be done by physicists at different regional and local centres. An important feature of the data is that about 90% is read-only data. This results in a simplification for replica synchronisation, as well as providing limited concurrency problems. However, the complexity for storing and replicating data is still high. Recently, at CERN the DataGrid [2] project has been initiated that deals with the management of these large amounts of data [5], and also involves job scheduling and application monitoring. The project does not only cover the High Energy Physics community, but also earth observation and bio-informatics. Thus, the research and development will serve several data intensive scientific communities.

Digital Sky Survey: The use of the Storage Resource Broker (SRB) [11] at SDSC as a data handling system, for the creation of an image collection for the 2MASS 2-micron all sky survey. The survey has 5 million images, comprising 10 Terabytes of data that need to be sorted from the temporal order as seen by the telescope, to a spatial order for images co-located in the same region of the sky. The data is read from an archive at Caltech, sorted into 140,000 containers for archival storage at SDSC, and accessed from a digital library at Caltech. When the collection is complete, the images will be replicated in the Caltech archive, and provided for use by the entire astronomy community.

5 Supporting Primitive and Global Services

We identify core services which should be supported on all storage resources to be employed for Grid-enabled applications. The definition of these services does not pre-suppose any particular implementation or mechanism. Such operations

either occur very frequently, or are required to support the minimal functionality in Grid-enabled applications.

Data access operations: These operations include 'read' or 'write' to support data access and update. Such operations must also be supported by addressing information to locate data within the store. Such data access operations may also support access to collections of primitive data units through specialised operations.

Location transparency and global name space: These operations are used to discover the location of a data source, and to keep the location of a data source independent of its access method. Interoperability across heterogeneous systems is required for both federation of legacy data stores, as well as for migration of data stores onto new technology over time. A global name space may be provided through a catalogue. CERN, for instance, has defined a global Objectivity namespace, but has not yet provided the mechanisms to automate the management of the namespace. Digital objects may also be aggregated into containers – based on a different similarity criteria. Aggregation in containers is one way to also manage namespaces, providing a logical view of object location, as long as the data handling system is able to map from the desired object into the physical container. Also containers eliminate latencies when accessing multiple objects within the same container. This requires that the data handling system migrate the container to a disk cache, and support accesses against the cached container for multiple accesses.

Previledge and security operations: These operations are required to enable systems or individuals to access particular data sources, without having an account on the system hosting the data source. In Grid environments, it will not be possible to have an account on every storage system that may hold data of interest. To manage large distributed collections of data, the digital objects which act as data stores, must be owned by the collection, with access control lists managing permission for access independently of the local storage system. In this approach, the collection owns the data that is stored on each local storage system, meaning the collection must have a user ID on each storage system. Access control may be supported through a catalogue.

Persistence and Replication: Within data Grids large amounts of data need to be replicated to distributed sites over the wide area. When accessing such replicated data sources, it is difficult to utilise URLs as persistent identifiers, because the URL encapsulates the storage location and access protocol within the identifier. Similarly, it is not possible to use PURLs [7] because while a mapping to a unique identifier can be preserved, the mapping is not automatically updated when an object is moved. Persistence may be managed by moving all data objects within a data handling system. This makes it possible to update the storage location, the access protocol, and data object attributes entirely under application control. All manual intervention associated with maintaining the persistent global namespace has been eliminated.

Check-pointing and state management: A central service may be supported for recording the state of a transaction. An application user could subscribe to

such a check-point service, to ensure that, on failure, it would be possible to re-build state and re-start the operation. This is not possible for all transactions however, and pre-defined points where a check-point is viable need to be identified by the user application.

6 Conclusion

In Grid computing and especially computing intensive data Grids, data management is a vital issue and has not been addressed to a large extend in high performance computing. In high throughput computing, several scientific domains have chosen the Grid as a basic part for a distributed computing model. Data management issues often need to be addressed in similar ways which requires protocols and standards.

References

1. 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, See Web site at: `http://www.globus.org/`, 1999.
2. DataGrid Project. See web site at: `http://www.cern.ch/grid/`, 2001.
3. Global Grid Forum. See web site at `http://www.gridforum.org`, 2001.
4. GriPhyN Project. See web site at: `http://www.griphyn.org`, 2001.
5. W. Hoschek, J. Jaen-Martinez, A. Samar, H. Stockinger, K. Stockinger, Data Management in an International Data Grid Project, *1st IEEE, ACM International Workshop on Grid Computing (Grid'2000)*, Bangalore, India, December 2000.
6. K. Kleese, Data Management for High Performance Computing Users in the UK, *5th Cray/SGI MPP Workshop*, CINECA, Bologna, Italy, September 1999.
7. Persistent Uniform Resource Locator. See web site at: `http://purl.oclc.org/`, 2001.
8. Ron Oldfield, Summary of Existing Data Grids, white paper draft, Grid Forum, Remote Data Access Group. `http://www.gridforum.org`, 2000.
9. J. Saltz et al. The DataCutter Project: Middleware for Filtering Large Archival Scientific Datasets in a Grid Environment, University of Maryland, 2000. See web site at: `http://www.cs.umd.edu/projects/hpsl/ ResearchAreas/DataCutter.htm`
10. H. Stockinger, K. Stockinger, E. Schikuta, I. Willers, Towards a Cost Model for Distributed and Replicated Data Stores, *9th Euromicro Workshop on Parallel and Distributed Processing (PDP 2001)*, IEEE Computer Society Press, Mantova, Italy, February 2001.
11. Storage Resource Broker Project. See web site at `http://www.npaci.edu/DICE/SRB/`, 2000.
12. The Globus Project - White Paper. "GridFTP: Universal Data Transfer for the Grid", September 2000. See Web site at: `http://www.globus.org/`
13. The RasDaMan Project. See web site at: `http://www.forwiss.tu-muenchen.de/~rasdaman/`, 2000.
14. "XML: Extensible Markup Language". See web site at: `http://www.xml.com/`
15. See Web site at: `http://www.ccds.org/RP9905/RP9905.html`