Eight Times Acceleration of Geospatial Data Archiving and Distribution on the Grids

Frank Z. Wang, *Senior Member, IEEE*, Na Helian, Sining Wu, Yike Guo, Derek Yuhui Deng, Lingkui Meng, Wen Zhang, Jon Crowncroft, Jean Bacon, *Fellow, IEEE*, and Michael Andrew Parker

***Abstract*—A grid-powered Web Geographical Information Science (GIS)/Web Processing Service (WPS) system has been developed for archiving and distributing large volumes of geospatial data. However, users, WPS servers, and data resources are always distributed across different locations, attempting to access and archive geospatial data from a GIS survey via conventional Hypertext Transport Protocol, Network File System Protocol, and File Transfer Protocol, which often encounters long waits and frustration in wide area network (WAN) environments. To provide a "local-like" performance, a WAN/grid-optimized protocol known as "GridJet" developed at our lab was used as the underlying engine between WPS servers and clients, which utilizes a wide range of technologies including the one of paralleling the remote file access. No change in the way of using software is required since the multistreamed GridJet protocol remains fully compatible with the existing IP infrastructures. Our recent progress includes a real-world test that PyWPS and Google Earth over the GridJet protocol beat those over the classic ones by a factor of two to eight, where the distribution/archiving distance is over 10 000 km.**

***Index Terms*—Bulk data transfer, data archiving, geodata set, grid computing, Web Processing Service (WPS).**

I. Introduction

THERE HAVE been a number of developments in Geographical Information Science (GIS), which address data archiving and distribution over the last decade. Much recent attention has focused on developing GIS functionality in the Internet, Worldwide Web (WWW), and private intranets and is sometimes termed WebGIS. WebGIS holds the potential to make distributed geographic information available to a very large worldwide audience. Internet users will be able to access GIS applications from their browsers without purchasing proprietary GIS software. WebGIS makes it possible to add GIS functionality to a wide range of network-based applications in business, government, and education [1].

The Open Geospatial Consortium (OGC) is a contemplating adoption of a technology called Web Processing Service (WPS). The interface to a WPS has been specified. This interface specification provides mechanisms to identify the spatially referenced data required by the calculation, initiate the calculation, and manage the output from the calculation so that it can be accessed by the client. This WPS is targeted at processing both vector and raster data [2]. A WPS can be configured to offer any sort of GIS functionality to clients across a network, including access to preprogrammed calculations and/or computation models that operate on spatially referenced data. A WPS may offer calculations as simple as subtracting one set of spatially referenced numbers from another (e.g., determining the difference in influenza cases between two different seasons) or as complicated as a global climate change model. The data required by the WPS can be delivered across a network or available at the server [2].

The aforementioned WebGIS and WPS applications can be exemplified by Google Earth [3] and PyWPS [4]. Google Earth combines the power of Google search with satellite imagery, maps, terrain, and 3-D buildings to put the world's geographic information at fingertips [3]. PyWPS is an implementation of OGS's WPS standard that has been started in April 2006. It offers environment for programming own process (geofunctions or models) which can be accessed from the public [4].

The challenge lies in creating a middleware that is platform independent and runs on open Transmission Control Protocol (TCP)/IP-based networks, i.e., on any computer capable of connecting to the Internet (or any TCP/IP-based network) and running a Web browser. The underlying infrastructure is communication protocols, which are responsible for implementing much of the structure of the data repository. For example, Google Earth gives a wealth of imagery and geographic information. End users are enabled to open the files and browse them just like a document but in a visually intuitive and interactive interface. They set in motion in a request/response transaction between their Google Earth browser and a Google Earth server, acting in a client/server relationship, which implements that navigation.

Manuscript received February 1, 2008; revised June 6, 2008. First published March 27, 2009; current version published April 24, 2009. This work was supported in part by the U.K. government and in part by the European Commission (EC) through an EPSRC/DTI grant ($1 million) Grid-Oriented Storage (GOS), an EPSRC grant ($470 thousand) "Accelerating NFS/CIFS," an EC grant (€1.01 million) "QuickLinux," and an EC grant (€400 thousand) "EuroAsiaGrid."

F. Z. Wang and S. Wu are with the Centre for Grid Computing, Cambridge–Cranfield High Performance Computing Facility, Cranfield, MK43 0AL, U.K. (e-mail: frankwang@ieee.org).

N. Helian is with the School of Computer Science, University of Hertfordshire, Hatfield, AL10 9AB, U.K. (e-mail: N.Helian@herts.ac.uk).

Y. Guo is with the Department of Computing, Imperial College London, London, SW7 2AZ, U.K. (e-mail: yg@inforsense.com).

D. Y. Deng is with the EMC Corporation, Hopkinton, MA 01748 USA (e-mail: Deng_Derek@emc.com).

L. Meng and W. Zhang are with the School of Remote Sensing and Information Engineering, Wuhan University, Wuhan 430072, China (e-mail: lkmeng@whu.edu.cn).

J. Crowncroft and J. Bacon are with the Computer Laboratory, University of Cambridge, Cambridge, CB3 0FD, U.K.

M. A. Parker is with the Cambridge eScience Centre, University of Cambridge, Cambridge, CB3 0FD, U.K.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TGRS.2008.2010055

0196-2892/$25.00 © 2009 IEEE

Authorized licensed use limited to: UNIVERSITY OF HERTFORDSHIRE. Downloaded on May 12, 2009 at 05:04 from IEEE Xplore. Restrictions apply.

In practice, attempting to share or access geospatial data via conventional Hypertext Transport Protocol (HTTP) [5], Network File System (NFS) [6], and File Transfer Protocol (FTP) [7] across the wide area network (WAN) often proves to be a frustrating time-consuming experience [8]–[10]. The purpose of this paper is to report on the advances in addressing network latencies relating to spatial data archiving and distribution. The emphasis is on the development of a grid-powered system for large volumes of Earth remote sensing data. We fill the gap for two vital missing capabilities: high-performance remote and bulk spatial data distributing/archiving, and secure data management integrated with the existing grid authentication and authorization tools. GridJet is a new approach that extends established Internet architectures and protocols to meet the aforementioned immediate needs and is positioned to adapt to the future needs of WWW/grid through the versioning provision of existing protocols.

We continue in Section II with a discussion in relation to communications and networking, their impact on the infrastructure of the WPS, their related research, and our uniqueness/innovation. Section III discusses our implementation of GridJet. Section IV reports the response-time improvement compared to HTTP-based transfer. Section V describes a real-world WPS test that exploits the latency reduction of our WebGIS/WPS/GridJet prototype. Section V concludes with a discussion of ongoing research.

II. Related Work

There are fundamental performance issues stemming from technical limitations and the architectural design. In order to provide accessible process, massive spatial data need to be assembled together by the server. These data are not just stored in the server but more often are delivered across the network for the distributed nature of geospatial data. In most cases, the amount of needed data reaches to gigabytes or even more; to transfer or archive so much data is obviously a challenge. Of course, the same problem also exists in simple WebGIS services while browsing a map via WWW. At present, obtaining data from remote resource is mainly via HTTP or FTP, resulting in low efficiency and poor security. In 2006, Friis-Christensen *et al.* studied the feasibility of using services offered by a spatial data infrastructure as the basis for distributed service oriented geoprocessing. This paper shows that there are considerable benefits by a distributed geoprocessing environment (e.g., nonreplicated data and reusability) [11]. In 2007, Jagery *et al.* [12] presented a complete framework for registering, discovering, composing, and executing Web services to support online science, which take geoprocessing as workflows and using workflow tools to obtain distributed geospatial data through network.

Many researchers think the grid technology as a new solution to distribute and archive huge amount of distributed geospatial data. In 2004, Gadgil *et al.* designed a grid-based scripting workflow system that is capable of managing both streaming data sources and service-based applications, and applied this system to flood modeling codes coupled to simulated NEXRAD Doppler radar data [13]. In 2007, Millina *et al.* [14] worked for satellite image data service and developed new data delivery infrastructures exemplified with the provision of OGC services, image streaming access, and compliance with geobrowsers such as Google Earth. In 2007, the Open Grid Forum signed a memorandum of understanding to collaborate with the OGC, which contains to integrate WPS to the grid infrastructure such as TeraGrid, National Research Grid Initiative, Enabling Grids for E-sciencE, and the United Kingdom's National Grid Service. The grid data movement tools such as GridFTP [15] are used in WPS.

WebGIS and WPS normally use HTTP to transfer data and requests. Different from HTTP/FTP/NFS/GridFTP, our GridJet protocol supports parallel stream data transfers and single system image (SSI). It can run as a stand-alone tool or can be integrated in the WebGIS/WPS framework. GridJet's goal of acceptance among diverse organizations requires that it provides fast and secure means of cross-domain remote spatial data access, archiving, and bulk data transfer.

III. GridJet: An Underlying WAN/Grid-Optimized Protocol for WebGIS/WPS Applications

A. Overall Design

In this paper, we show how to interface PyWPS and Google Earth with our developed GridJet protocol, as graphically shown in Fig. 1. GridJet currently supports Linux, and future work includes extension to Windows and MacOS X-based systems.

PyWPS is written in Python programming language. The main advantage of PyWPS is that it has been written with native support for Geographic Resources Analysis Support System (GRASS) [4]. PyWPS supports to create and remove on-the-fly generated temporary archives and directories, like temporary GRASS locations and mapsets and other files created as part of the calculation [4]. PyWPS allows one to access and archive distributed geospatial data across the networks and to access GRASS modules via Web interface. In fact, PyWPS has implemented around 95% standard of WPS, of which new improvement will be available in the forthcoming release version [4].

Three important parts can be seen in Fig. 1, which are clients, distributed data nodes (DDNs), and WebGIS server. Users could access the geoprocessing services remotely via the Web browser or desktop GIS via HTTP. Data archiving and distributing are performed between WebGIS server and DDN, with the aid of HTTP. DDNs always manage massive and comprehensive geospatial data. Data communications between the WebGIS and DDN could be frequent when performing some online applications. For this reason, the DDN is deployed on a GridJet server and the WebGIS/WPS server on a GridJet client.

For Google Earth, it should be much simpler, in which a user simply supplies a URL, as well as a protocol (choosing either HTTP or GridJet, or another protocol), in the address bar of the browser or in the hyperlinks of the objects on the Web pages, then can browse, upload, and download spatial data from a WebGIS server.

Authorized licensed use limited to: UNIVERSITY OF HERTFORDSHIRE. Downloaded on May 12, 2009 at 05:04 from IEEE Xplore. Restrictions apply.

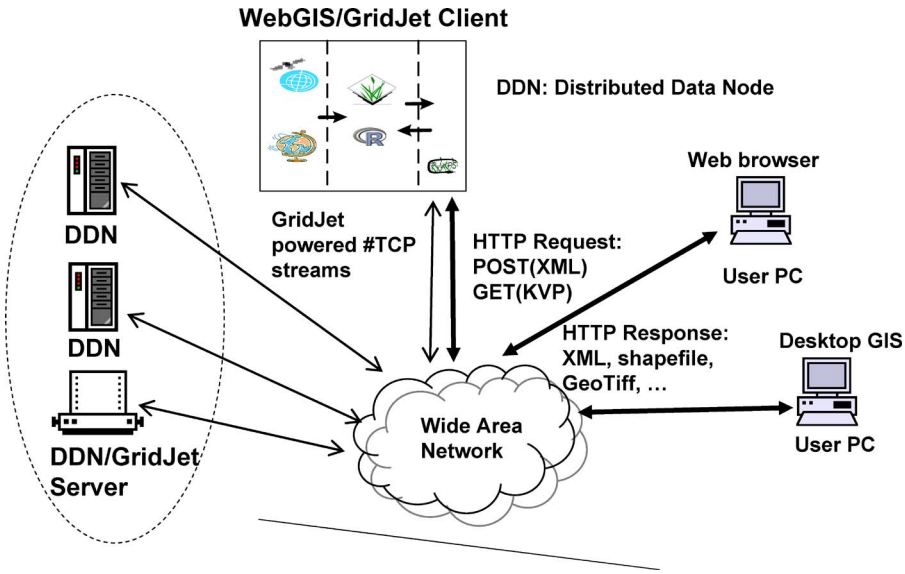

Fig. 1. How the PyWPS works, in which GRASS works as a GIS tool in the WebGIS server.

B. GridJet Implementation

The emergence of high speed WANs makes grid computing a reality. However, grid applications still have difficulties to achieve optimal TCP performance due to the network tuning of TCP window size to improve the bandwidth and reduce latency on a high speed WAN.

Within the context of the U.K. Government Project "Grid-Oriented Storage" (GOS) and the European Commission (EC) Project "EuroAsiaGrid," a grid-based middleware solution known as "GridJet" has been developed in our lab, which melds distributed file system technology with high-performance data transfer techniques to meet the needs of WAN/grid-based virtual organizations (VOs). From its inception, the developed multistreamed GridJet, via the fully integrated parallel stream mechanism, is designed to deal with long-distance, bulk data, cross-domain, and single-image file operations—a capability lacking in the current versions of single-streamed FTP. GridJet alleviates WAN latency issues for applications based on FTP standards. First-of-its-kind GridJet solution enables users to access and save documents across the WAN/grid at near-local area network (LAN) speeds.

GridJet uses parallel streams to achieve very high transfer rates at a fraction of the memory cost. As shown in Fig. 3(c), GridJet opens multiple socket streams between the sender and the receiver for applications. The application data are then partitioned into segments. The number of segments is equal to the number of streams. The segments are sent through streams in parallel by different threads and are then reassembled by the receiver. The final data are presented to applications as if they were transmitted through a single socket. Fig. 2 also shows why sending partitioned data through multiple socket streams can achieve near optimal bandwidth. In the figure, the shaded areas represent socket buffer size, and the large empty rectangles represent TCP pipe capacity (bandwidth $*$ delay). Applications using a single socket stream will not fill the TCP pipe capacity. The capacity efficiency is about 1% when the buffer size is taken as its default value of 64 kB [16]. Applications using either a single socket stream with an optimal buffer size or multiple socket streams can fill the TCP pipe, as shown in Fig. 3(b) and (c). Chen *et al.* [17] showed that using parallel stream is more effective than tuning TCP window size. In addition, acknowledging the entire group requires the entire file to be retransmitted even if a single packet is lost, as shown in Fig. 3(b). Therefore, applications using GridJet can obtain near optimal TCP performance without adjusting TCP window size.

Since Globus is becoming the overwhelming choice for grid development, non-Globus solutions will be difficult to melt seamlessly into popular grid environments [18]. The GridJet uses Globus XIO [19] to communicate with the parallel stream engine during a file read/write operation. As shown in Fig. 2, Globus XIO is broken into two main components: framework and drivers. The Globus XIO framework manages I/O operation requests that an application makes via the user application programming interface (API). The parallel stream driver is responsible for manipulating and transporting the user's data. The grid security infrastructure (GSI) driver performs necessary messaging to authenticate a user and the integrity of the data. The TCP driver executes the socket level transport code contained within to establish a connection to the given contact string.

The virtual filesystem switch (VFS) is implemented in the kernel. This implementation conforms naturally to the standard Portable Operating System Interface (POSIX) semantics and provides applications with seamless access to GridJet. A kernel-memory module, which is the GridJet kernel client, acts as a VFS interface. The GridJet server nominates a user-space daemon, which is the GridJet user client, to penetrate the kernel/user boundary and to communicate with the kernel client [16].

The pair of the aforementioned kernel and user clients is connected by three different message/data passing mechanisms for different considerations, respectively. The first mechanism is a pipe, one end of which is written by the kernel client and another end of which is read by the user client. The second

Authorized licensed use limited to: UNIVERSITY OF HERTFORDSHIRE. Downloaded on May 12, 2009 at 05:04 from IEEE Xplore. Restrictions apply.

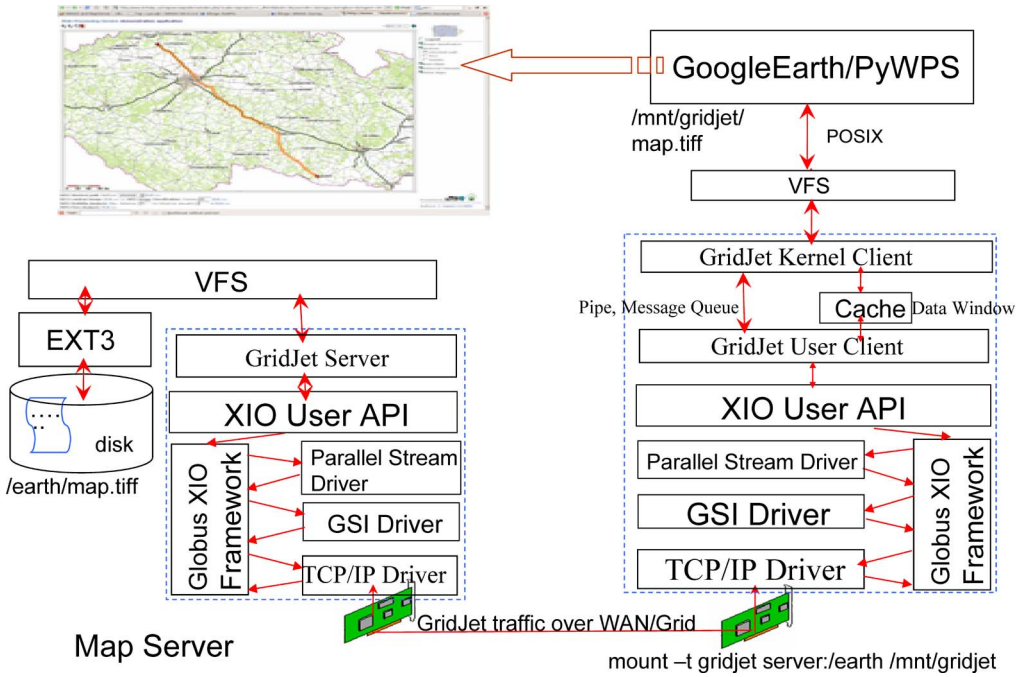

Fig. 2. Google Earth/PyWPS interfaced with GridJet. The VFS implementation conforms naturally to the current information technology infrastructure. The GridJet uses Globus XIO to communicate with the multistream engine and GSI. A GridJet client mounts a file directory/earth/map.tiff on a remote file server to its local file tree/mnt/GridJet/ via the GridJet protocol, and thus, the Google Earth/PyWPS browser can access it as if it was local. EXT3 is a local file system. XIO is a Globus interface. See the main text for details.

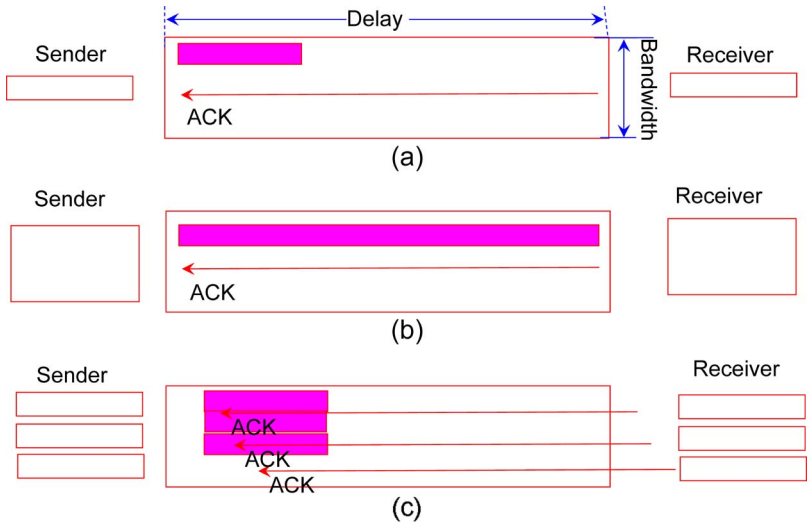

Fig. 3. Single or multiple TCP socket streams with different buffer sizes. (a) Single socket stream with default buffer size. (b) Single socket stream with buffer size = capacity. (c) Multiple socket streams with default buffer size.

mechanism is a message queue, in which each message generated by the user client stays until the kernel client reads it. The third is a new mechanism we call "data window." The "data window" mechanism (Fig. 4) breaks the space limit (32 MB) of the well-used interprocess communication (IPC) shared memory since we focus our attention on the bulk data transfer in the GOS project. Like the IPC shared memory, the implemented "data window" mechanism also avoids copying data between the user and kernel spaces [16].

Hyperlinked objects in a Google Earth browser are typically accessed in chunks of data at a time, slowing down the transfer of objects across the WAN. It is possible to move the entire objects across the WAN before the data are actually required by modifying the file size of the underlying file system calls (the default is 4 kB). Therefore, information is already localized when users need it. Nevertheless, modifying the default file size may be in contradiction with the effort of making our GridJet accelerator universal. We had better use default configurations without applying any optimization. This is the usage condition for most scientists that normally have neither the needed expertise nor the time to configure the tools with high levels of optimization.

As shown in Fig. 2, a cache is introduced in the aforementioned data window of the GridJet to accumulate the file system

Authorized licensed use limited to: UNIVERSITY OF HERTFORDSHIRE. Downloaded on May 12, 2009 at 05:04 from IEEE Xplore. Restrictions apply.

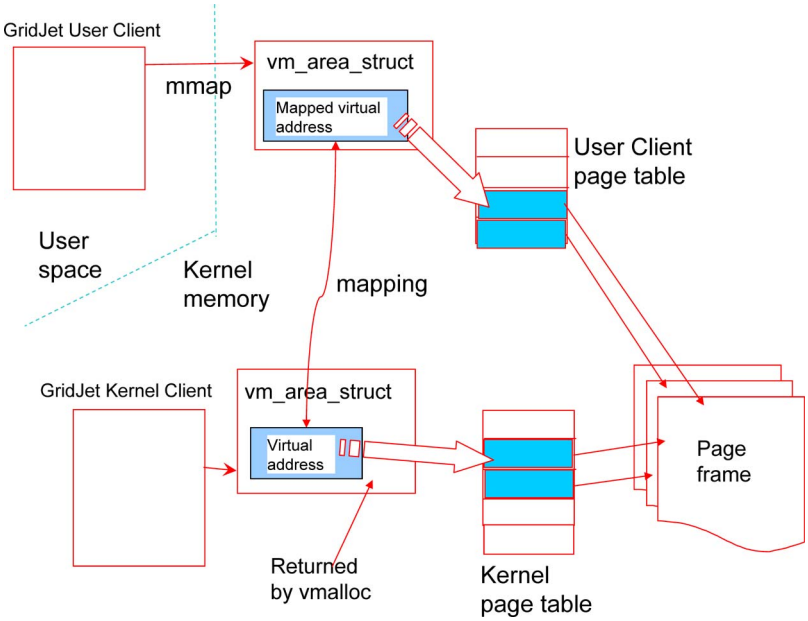

Fig. 4. Proposed "data window" mechanism breaks the space limit (128 MB) of the well-used IPC shared memory. A driver maps a virtual address to the user client's user space (page table), which allows the user and kernel clients to access some common data structures.

calls (with default file size) originated by the browser to the underlying filesystem GridJet. Prefetches mean a number of chunks (comprising the chunk being currently requested and the following ones) of the downloaded data are localized before those following ones are actually required. Note that Web browsing would benefit a lot from those prefetches because sequential reads are dominant. Deferred writes are allowed to accumulate in the cache until a threshold value for the overall size of a number of file chunks in cache is reached. File writes can often be flushed to GridJet in bulk file write operations that may be more efficient than the way the original application was specified. The result is more data sent in fewer trips and less WAN waiting time. In addition, applications using multiple socket streams fill the TCP pipe to achieve optimal bandwidth. This combination reduces file access time while reducing the number of performance-impacting round trips over the WAN.

Note that the GridJet is a considerably complex piece of software. The source code of the developed GridJet is of 40 000 lines in length, and we have spent nine man years in developing and revamping it. In 2007, this technology won an Association for Computing Machinery/IEEE SuperComputing Storage Challenge Finalist Award.[1]

The PyWPS/Google Earth protocol that moves objects across WWW is HTTP based. Because HTTP is single streamed, users attempting to work over HTTP must often deal with unacceptable wait times in WWW (60–1000 ms) [20]. In addition, a network filesystem protocol, in contrast to HTTP protocol, allows a user to use files on other network machines as if they were local. In addition, a filesystem protocol also creates an SSI unique to each user on each of a plurality of nodes in a VO.

Both FTP, FlashGet [21], and GridFTP can be used to transport data from one location to another. In the Pioneering High Energy Nuclear Interactions eXperiment (PHENIX), grid tools were used by the PHENIX to send recently acquired data to a regional computing center for the experiment in Japan [22]. The PHENIX data acquisition can sustain a peak data rate of up to 600 MB/s and runs at a typical rate of 250 MB/s [22]. GridFTP is much faster than FTP [15]. However, FTP/FlashGet/GridFTP are only independent tools, and they do not provide any API to enable it to be integrated into other applications. They cannot be used as an underlying engine to speed up other applications. We wanted to extend them.

C. GridJet Security Mechanism

Grid applications not only require efficient transfer of a large amount of data in WANs but also demand security and authentication services that enhance data integrity and enable data access control. The GridJet and its variations use public key infrastructure associated with GSI [19] to authenticate the identities of WAN/grid members and to secure resource allocation to these members. The GridJet server requests and receives a host certificate from the Certificate Authority. The GridJet (service) also needs to be registered in the VO—Lightweight Directory Access Protocol (LDAP). Furthermore, the VO LDAP generates a gridmap file to the GridJet server from the LDAP database for authorization and mapping. Beyond the role of certificate registration, the VO LDAP can perform other important roles for the VO, such as a service search engine and user authorization.

[1]Finalist Award of Storage Challenge at the ACM/IEEE Super Computing in the U.S. The Awards Ceremony was on November 15, 2007. Finalist team: Frank Z. Wang (Centre for Grid Computing, Cambridge–Cranfield HPCF); Na Helian (London Metropolitan University); Sining Wu, Yuhui Deng, Vineet R. Khare, and Chenhan Liao (Centre for Grid Computing, Cambridge–Cranfield HPCF); Amir Nathoo Rodric Yates Paul Fairbairn (IBM Hursley Laboratory); Jon Crowcroft, Jean Bacon, Michael Andrew Parker (Cambridge University); Zhiwei Xu (Institute of Computer Technology); and Yike Guo (Imperial College). Finalist project: Grid-Oriented Storage: Parallel Streaming Data Access to Accelerate Distributed Bioinformatics Data Mining.

Authorized licensed use limited to: UNIVERSITY OF HERTFORDSHIRE. Downloaded on May 12, 2009 at 05:04 from IEEE Xplore. Restrictions apply.

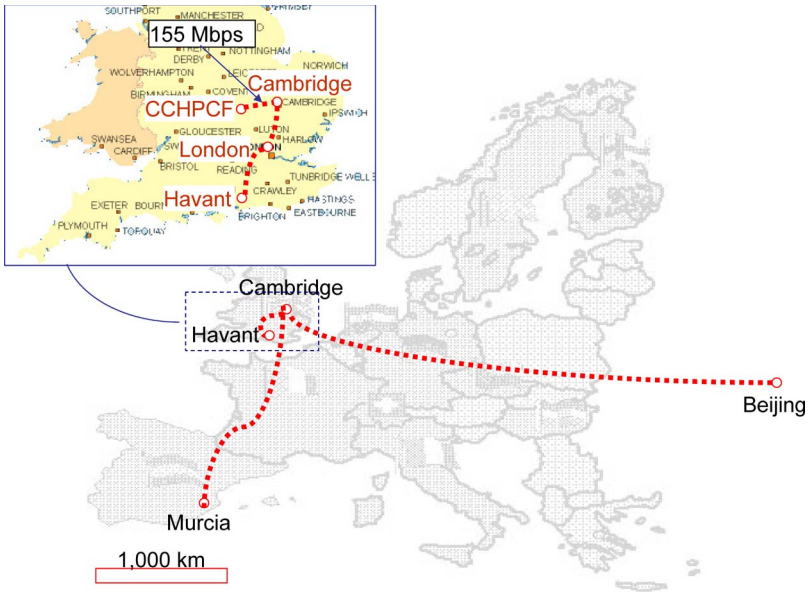

Fig. 5. Connectivity map of the EuroAsiaGrid participating sites.

One of the interesting parts of a GridJet network is how often users communicate with the VO LDAP in relation to how often they communicate with GridJet servers. Essentially, the LDAP only needs to be accessed to initialize a session between a user and a GridJet server. After that, all communications can be handled between the user and the GridJet server. Data transfers that are broken off prematurely will need to be reestablished, but it may not be necessary to authenticate again with the LDAP if the public key included in the certificate is still available to use. This design removes the VO LDAP as a potential storage I/O bottleneck [23].

IV. Real-World GridJet-Powered WebGIS/WPS Geospatial Data Archiving and Distribution

The purpose of this investigation was to examine the prototype of WebGIS/WPS over GridJet, test, and compare its performance to standard single-stream HTTP transportation. Because the GridJet will be the underlying data transfer engine, it is very important to have a first-hand experience of its performance and capabilities in WWW environments. We choose PyWPS to realize a real-world GridJet-powered WPS. Data transformation, distributing/archiving, and overlay analysis are chosen as typical geoprocessing operations. As shown in Fig. 2, a WPS server mounts a remote data server with the directory/earth/$**$ to its local file tree/mnt/GridJet/ via the GridJet protocol, and thus, the geoprocessing operation can process these data online as if they were local. In fact, GridJet is a grid middleware that is independent of the GIS application, although we implement it for accelerating the access and distributing geospatial data during geoprocessing. The applications of GridJet are not confined to WPS that could also be used in a common GIS system; therefore, we also use Google Earth to do some experiments, in which a client mounts a remote map server with the directory/earth/map.tiff to its local file tree/mnt/GridJet/ via the GridJet protocol, and thus, the Google Earth client can browse it online as if it was local.

A. EuroAsiaGrid Configuration

The following real-world tests demonstrate the value of the GridJet-powered WebGIS/WPS communications over the EuroAsiaGrid network resources. The EuroAsiaGrid Project is funded by the EC, consisting of over ten European and Asian partner institutions.

Five EuroAsiaGrid sites, which are Cambridge, London, Portsmouth, Murcia, and Beijing, have been participating in the tests. At each site, a Linux/Globus machine with installed GridJet client gateway was used as a client to access a dedicated WPS or Web map server with installed GridJet server gateway at the Cambridge–Cranfield High Performance Computing Facility (CCHPCF) in the investigation (see Fig. 5). The CCHPCF has a 155-Mb/s permanent connection to the Internet using 5-Gb/s SuperJANET backbone via the EastNet Cambridge node, while at the time of the investigation, Murcia had a 10-Mb/s connection, London had a 2-Mb/s connection, and Beijing and Portsmouth had a 100-Mb/s connection.

During the investigation, it was found that the background load of the network link had a significant influence on the Web performance: for example, the read time of a map of 16 MB over the 32tcp GridJet from Cambridge and Beijing could change from 134 s at 22:00 GMT to 1081 s at 14:00 GMT. Therefore, in order to eliminate the effect of changing network load, operation times were compared only if the operations were performed over the same network link and taken within the same short time interval; furthermore, all the unreconstructable and outstanding values were discarded.

The main tests were carried out between Cambridge and Beijing. The Cambridge–Beijing datalink represents the longest network connection in these online browsing tests over GridJet (geographic distance of 10 000 km, 27 router hops, and $average\ RTT = 539$ ms). The CCHPCF has a 155-Mb/s permanent connection to the Internet, while the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China, has a 100-Mb/s LAN connection. The data

Authorized licensed use limited to: UNIVERSITY OF HERTFORDSHIRE. Downloaded on May 12, 2009 at 05:04 from IEEE Xplore. Restrictions apply.

travel from Cambridge to London, from London through a trans-Atlantic link to Huntington, the American West Virginia gateway, then through a trans-Pacific link via Australia, and finally arrive in Beijing.

B. Test Methodology

Since GridJet, to support PyWPS and others, is built on top of Globus libraries, we introduce an additional layer as an abstraction of the communication infrastructure. To quantify any possible overhead, we compared the transfer times in seconds for accessing online geoprocessing between a WPS server at CCHPCF and a remote data center in Beijing using different transmission tools: HTTP and GridJet for remote data transferring and archiving during geoprocessing. We choose geospatial data transformation, overlay analysis, and map distributing/archiving in experiments, because these three operations are well used during crisis transaction or dealing with emergencies by WPS. At the same time, we test the transfer times in seconds for browsing online maps via Google Earth/PyWPS remotely in a same way, for which a map server is at CCHPCF and a client is in Beijing. We ran the experiments three times for each protocol. The figures reported in this section are the average values of those measurements.

C. Archiving and Distributing Geospatial Data via GridJet on EuroAsiaGrid

Extensive performance tests were conducted to measure the effectiveness of WebGIS/WPS over GridJet, in terms of the response-time improvements delivered to users. HTTP is chosen as the standard transfer protocol. Speedup is defined as the transfer time of HTTP over the transfer time of Google Earth/PyWPS deployed on GridJet protocol. The GridJet security was also verified.

Archiving is preliminarily realized by a newly developed Data Interface All-iN-A-place (DIANA) [24]. DIANA includes the POSIX standard (in the kernel), the Structured Query Language standard (in the user space), and an extensible interface reserved for metadata operations. A global multidimensional index facility indexes all saved geodata objects based on their metadata tags and provenance. It is found that a speedup of 5000 in indexing has been achieved at the expense of a slowdown of 100 in extracting attributes [24]. DIANA supports inherent interactions between files and databases, automates the collection of metadata while the data it refers to are being generated, and also automates the consistency between data and its associated metadata while the data are being updated.

1) Google Earth Over GridJet: We measured the access time of a client browsing online a map of 41.8 MB across a 100-Mb/s link that is typical in a wired WAN environment. It took 400 s to transfer and display this map using HTTP when a WAN latency is 539 ms. When we added the GridJet server/client gateways (choosing GridJet rather than HTTP or another protocol in the address bar of the browser) to the link with $\#tcp = 16$, the same 41.8-MB map required just 73 s to display online. A speedup of 5.5 can be seen. When $\#tcp = 64$, the time viewing (transferring) the 41.8 MB map is 52 s, representing a response-time improvement of 7.7 times (Table I).

TABLE I
GOOGLE EARTH PERFORMANCE IN A REAL-WORLD TEST BETWEEN CAMBRIDGE AND BEIJING

	access time in seconds, rtt=539ms		Speedup
#tcp	GoogleEarth/GridJet	GoogleEarth/HTTP	
1tcp	448	400	0.9
16tcp	73	400	5.5
64tcp	52	400	7.7
128tcp	51	400	7.8

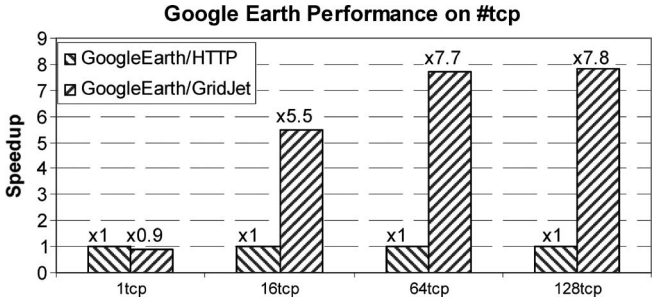

Fig. 6. Test measures the speedup as a function of the number of TCP streams, namely, Google Earth over HTTP and GridJet, respectively. A 42-MB map is used in all the measurements. The maximum improvement goes up to 7.8 times with 128 parallel streams. However, GridJet would eventually consume too much time in managing too many TCP streams, slowing down such an improvement with further increased $\#tcp$.

It is interesting to see that, although GridJet is characterized by an initial transfer time (448 s) that is even larger than HTTP (400 s) when $\#tcp = 1$ (no acceleration is added), this gap tends to decrease as $\#tcp$ increases, and for $\#tcp$ larger than four, the GridJet overtakes HTTP, reaching a speedup of 7.8 when $\#tcp = 128$. The reason for the additional overheads in the GridJet functionalities is due to the additional layer of abstraction of GridJet on top of the Globus libraries.

Fig. 6 shows the response-time improvement as a function of the number of TCP streams $(\#tcp)$. It is a standard open of a 41.8-MB map over the link with 539-ms round-trip time (RTT). Increases in the response-time improvement with increased $\#tcp$ can be seen toward the right side of the graph. The maximum improvement goes up to 7.8 times with 128 parallel streams. However, GridJet would eventually consume too much time in managing too many TCP streams, slowing down such an improvement with further increased $\#tcp$.

2) PyWPS Over GridJet: Aiming at the performance improvement of PyWPS via GridJet for geospatial data archiving and distributing, we carry out a real-world test between Cambridge and Beijing. The main network parameters are the same as the Google Earth tests, but the data set is changed for more complex geodata processing. We choose three maps: one shapefile of 5.2 MB, one spot image of 5.4 MB, and another spot image of 5.6 MB.

The experiments include three operations, namely, remote geospatial data transformation, online overlay analysis, and map distributing/archiving. To carry out a remote geospatial data transformation, we access and transform remote data via GridJet. In an online overlay analysis, not only the analysis service could be requested via the Internet/grids but also the data set for the analysis comes from a remote source. Distributed geospatial data are read from a remote data node via GridJet

Authorized licensed use limited to: UNIVERSITY OF HERTFORDSHIRE. Downloaded on May 12, 2009 at 05:04 from IEEE Xplore. Restrictions apply.

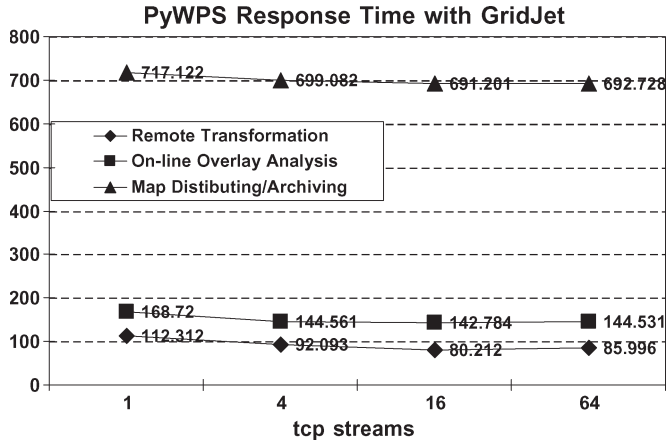

Fig. 7. Measured response time of PyWPS as a function of TCP streams. It is found that the system performance reaches its best when $\#tcp = 16$.

TABLE II
PyWPS PERFORMANCE IMPROVEMENT AS $\#tcp = 16$ WHEN CHANGING RTT ($S = Speedup$)

RTT(ms)	PyWPS Test time in seconds, #tcp=16								
	Remote Transformation			On-line overlay analysis			Map Distributing/Archiving		
	GridJet	HTTP	S	GridJet	HTTP	S	GridJet	HTTP	S
40	6.9	10.3	1.5	11.1	18.0	1.6	52.2	99.6	1.91
80	12.5	20.4	1.63	20.2	43.2	2.1	102.2	199.4	1.95
160	27.7	41.0	1.48	39.4	88.3	2.2	203.2	403.2	1.98
320	47.4	81.6	1.72	76.7	176.9	2.3	402.7	836.9	2.08

and are analyzed by running the overlay analysis locally. From time to time, people need to preserve their processed geospatial data in a remote site. Possible reasons include the limited usage privilege, limited local storage space, or presenting data to a third party. In our test, a new map, generated from the overlay analysis, is archived to a remote data server via GridJet. Fig. 7 shows the measured response time of PyWPS as a function of TCP streams. It is found that the system performance reaches its best when $\#tcp = 16$ (Table II).

GridJet is designed to deal with long-distance, cross-domain, and single-image data access and transfer operations. In the transformation test, we choose geospatial data of 5.4 MB to be transformed (from vector to raster format) online across a 100-Mb/s link that is typical in a wired WAN environment. When $RTT = 80$ ms and $\#tcp = 16$, we get an improvement of 1.63 times, compared with that using HTTP. In the overlay analysis tests, we read three heterogeneous spatial data files from a remote server and overlay them on a local display. These three data files are a shapefile map (in vector format) and two spot images (in raster format), each of which is larger than 5 MB. When $RTT = 320$ ms and $\#tcp = 16$, we reach a speedup of 2.3. In the distributing/archiving test, a new overlaid map, generated from the previous overlay analysis, is archived back to the same remote data server hosting the mentioned three heterogeneous spatial data files. Four groups of data have been tested with RTT between 40 to 320 ms, in which GridJet outperforms HTTP by a factor of 1.5–2.3. Fig. 8 shows the improvement ($\#tcp\ streams = 16$) in response time when performing the aforementioned geoprocessing operations.

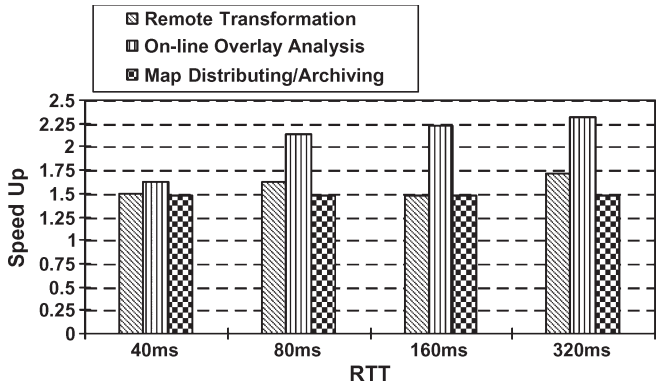

Fig. 8. Test measures the speedup as a function of the RTT, namely, PyWPS over HTTP and GridJet, respectively. The average improvement goes about 2 times with 16 parallel TCP streams.

D. Result Analysis

It is found that PyWPS and Google Earth, deployed on GridJet, behave differently. An improvement of 7.8 times has been observed with Google Earth, whereas 2.3-times acceleration with PyWPS can only be seen. Possible reasons are speculated as follows.

The underlying GridJet architecture must efficiently provide increased transfer bandwidth to accelerate WebGIS/WPS I/O. This is why a WAN/grid-optimized architecture is attractive for WebGIS/WPS servers. Intuitively, the developed WebGIS/WPS over GridJet is most effective if the size of request is sufficiently large to allow GridJet to transport them efficiently via multiple TCP streams. We verify WebGIS/WPS's ability to accumulate spatial data requests. The WebGIS/WPS is submitting I/O requests both synchronously (physical reads and writes that are blocking the request) and asynchronously (prefetching and preflushing that are nonblocking). There is thus a tradeoff: The WebGIS/WPS should submit as many asynchronous requests as necessary to increase throughput while making sure the filesystem is ready to process synchronous requests whenever they are submitted. Currently, GRASS uses hard page size 4 KB (1 chunk) for data files and buffer pool. Those requests with a large size would benefit from the increased transfer bandwidth provided by GridJet. Unfortunately, they do not occupy a large proportion from our observation via Strace [25].

As measured, a relatively large proportion of 1-chunk requests can be seen. These small read requests arise from fetching head files and other associated files. Even those large spatial data files are also accessed in chunks. In addition to those small read requests, some small write requests can also be seen. For example, each chunk needs to be written back to its resulting data file, resulting in a series of small writes. In addition, small updates to metadata are also necessary. However, those read requests for head and spatial data files are synchronous. These small requests cannot benefit a lot from the increased bandwidth of GridJet.

As shown in Fig. 2, the GridJet buffer is flushed if the buffer is filled with chunks, at various intervals depending on memory pressure and I/O activity. Few-chunk requests are issued from the flushing operations of a GridJet buffer with discontinuous blocks. As can be seen in the observation, most

Authorized licensed use limited to: UNIVERSITY OF HERTFORDSHIRE. Downloaded on May 12, 2009 at 05:04 from IEEE Xplore. Restrictions apply.

of small write requests have been satisfied within the GridJet buffer, resulting in a short response time. Instead of controlling the latency/throughput tradeoff by limiting the number of submitted I/O requests, the WebGIS/WPS should rely on the underlying GridJet architecture to accelerate critical I/Os in a timely manner without sacrificing throughput even if there are many outstanding requests.

A PyWPS is more advanced and complex than a Google Earth, because it provides powerful geoprocessing functions. Google Earth tends to generate large requests, as it uses streaming technology to get rich geodata to users, which takes full advantage of the underutilized network bandwidth. Our Google Earth tests are constrained to downloading and displaying maps on a client, whereas a PyWPS needs to access remote data when it is processing local data, which is time consuming.

Approximately 10 TB of geodata were transferred. After a few days of fine tuning the transfer parameters, the transfers became part of the regular data-handling operation, requiring experts to intervene only occasionally. This seems to be the first time that a data transfer of such magnitude was sustained over many weeks in actual production and was handled as part of routine operation by nonexperts. The successful completion of this large-scale transfer project demonstrates both the maturity of the developed Grid platform and the real feasibility of interfacing popular geoscience applications into the data handling and processing chain of large experiments.

V. Conclusion

This paper provides insights into the geospatial data distribution and archiving that can provide architectural, connectivity, mobility, and quality of service support on GIS. We expect this paper to provide innovative, flexible, and scalable solutions that can be widely deployed and adopted by end users over heterogeneous wired–wireless networks.

It is concluded that the usage of the underlying GridJet engine radically increased the transfer rate of the WebGIS/WPS, compared to those over single-streamed HTTP. WebGIS/WPS over GridJet is an innovative combination of the accelerated communication and heterogeneous wired–wireless networks to expand the capabilities and usability of WPS and common WebGIS and provides a simple access to grid technology. The GridJet-based end-to-end service support addresses the challenge of sharing documents over a WAN/grid with "LAN-like" performance, which is a highly beneficial aspect to the GIS community.

The core features include the following:

1) a 7.8-times improvement in response times for WebGIS (e.g., Google Earth) over GridJet protocol;
2) a 2.3-times improvement for WPS (e.g., PyWPS) over GridJet protocol;
3) the design divorces the GridJet communication from the Web GIS service object manipulation. That is to say, no change in the way of using software is required;
4) allows operators to maintain both maximum data access performance and security in their Web/grids attributed to the integration of GSI.

Acknowledgment

The authors would like to thank Prof. G. Gibson of Carnegie Mellon University, Prof. K. Li of Princeton University, Prof. R. Parker of Rolls Royce, Dr. R. Wright of BBC, Dr. F. Donno of European Council for Nuclear Research, Dr. S. Vandebroek of Xerox, Dr. P. Francis of the European Aeronautic Defence and Space, Dr. A. Simpson of Oxford University, Prof. C. Turner of King's College, Prof. I. Darwazeh of the University College London, Dr. D. Ferguson of Edinburgh University, Prof. L. Ni of Hong Kong University of Science and Technology, Dr. D. P. Vidyarthi of Jawaharlal Nehru University, and Dr. B. Fenix of HP for viewing our demonstrations and providing us with comments that much improved the GOS work.

References

[1] *Geographic Information Systems Resource—GIS*, 2008. [Online]. Available: http://www.webgis.com/
[2] OpenGIS, *Web Processing Service*, 2008. [Online]. Available: http://www.opengeospatial.org
[3] *Explore Google Earth*, 2008. [Online]. Available: earth.google.com
[4] J. Čepický and L. Becchi, "Geospatial processing via Internet on remote servers—PyWPS," *J. Open Source Geospatial Found.*, vol. 1, pp. 1–5, 2007.
[5] R. Fielding *et al.*, *Request for Comments: 2616*, Jun. 1999. Hypertext Transfer Protocol.
[6] "Network File System (Protocol)," *FreeBSD Handbook*, 2008.
[7] J. Postel and J. Reynolds, *Request for Comments: 959*, Oct. 1985. FILE TRANSFER PROTOCOL (FTP).
[8] P. Mayaux, H. Eva, J. Gallego, A. H. Strahler, M. Herold, S. Agrawal, S. Naumov, E. E. De Miranda, C. M. Di Bella, C. Ordoyne, Y. Kopin, and P. S. Roy, "Validation of the global land cover 2000 map," 1*IEEE Trans. Geosci. Remote Sens.*, vol. 44, no. 7, pp. 1728–1739, Jul. 2006.
[9] A. Abubakar, T. M. Habashy, V. L. Druskin, and L. Knizhnerman, "An enhanced Gauss–Newton inversion algorithm using a dual–optimal grid approach," *IEEE Trans. Geosci. Remote Sens.*, vol. 44, no. 6, pp. 1419–1427, Jun. 2006.
[10] S. Zine, J. Boutin, P. Waldteufel, J.-L. Vergely, T. Pellarin, and P. Lazure, "Issues about retrieving sea surface salinity in coastal areas from SMOS data," *IEEE Trans. Geosci. Remote Sens.*, vol. 45, no. 7, pt. 1, pp. 2061–2072, Jul. 2007.
[11] A. Friis-Christensen, L. Bernard, I. Kanellopoulos, J. Nogueras-Iso, S. Peedell, S. Schade, and C. Thorne, "Building service oriented applications on top of a spatial data infrastructure—A forest fire assessment example," in *Proc. 9th AGILE Conf. Geogr. Inf. Sci.*, Visegrád, Hungary, 2006, pp. 119–127.
[12] E. Jagery, I. Altintasy, J. Zhangz, B. Ludäscher, D. Penningtonz, and W. Michenerz, "A Scientific Workflow Approach to Distributed Geospatial Data Processing Using Web Services," in *Proc. 17th Int. Conf. Sci. Statistical Database Manage.*, 2005, pp. 87–90.
[13] H. Gadgil, J.-Y. Choi, B. Engel, G. Fox, S. Ko, and S. Pallickara, "Management of data streams for a real time flood simulation," Indiana Univ., Bloomington, IN, Jun. 2004.
[14] G. R. Millina, P. Ekina, and K. Kitmitto, "Proving spatial data support and access to freely available satellite imagery for the UK Academic Community: A review of data acquisition and data delivery infrastructure," in *Proc. RSPsoc Annu.*, 2007, pp. 1–5.
[15] *GridFTP: Protocol Extensions to FTP for the Grid*. RFC-2119.
[16] F. Wang, S. Wu, N. Helian, Y. Deng, A. Parker, Y. Guo, and V. Khare, "Grid-oriented storage: A single-image, cross-domain, high-bandwidth architecture," *IEEE Trans. Comput.*, vol. 56, no. 4, pp. 474–487, Apr. 2007.
[17] J. Chen, W. Akers, Y. Chen, and W. Watson, III, *Java Parallel Secure Stream for Grid Computing*, 2005. [Online]. Available: http://www.ihep.ac.cn/~chep01/abstract/10-008.htm
[18] Y. Deng and F. Wang, "A heterogeneous storage grid enabled by grid service," *ACM SiGOPS Oper. Syst. Revi.*, vol. 41, no. 1, pp. 7–13, Jan. 2007.
[19] *GlobusWORLD*, 2007. [Online]. Available: www.globusworld.com/program/program.php

Authorized licensed use limited to: UNIVERSITY OF HERTFORDSHIRE. Downloaded on May 12, 2009 at 05:04 from IEEE Xplore. Restrictions apply.

[20] A. S. Tanenbaum, *Computer Networks*, 4th ed. Englewood Cliffs, NJ: Prentice-Hall, 2002.
[21] *FlashGet*, 2008. [Online]. Available: http://www.flashget.com/index_en.htm
[22] CERN Courier, *PHENIX Experiment Uses Grid to Transfer 270 TB of Data to Japan*, Aug. 23, 2005. [Online]. Available: http://cerncourier.com/cws/article/cern/29401
[23] F. Wang, Y. Deng, N. Helian, S. Wu, V. Khare, C. Liao, and A. Parker, "Evolutionary storage: Speeding up a magnetic disk by clustering frequent data," *IEEE Trans. Magn.*, vol. 43, no. 6, pp. 2295–2297, Jun. 2007.
[24] F. Z. Wang, S. Wu, N. Helian, W. Zhang, T. Breckon, P. Dantressangle, R. Yates, P. Fairbairn, J. Bacon, and M. A. Parker, "DIANA: Data Interface All-iN-A-place," presented at the Cranfield Multi-strand Conf., Milton, U.K., May 6, 2008.
[25] 2008. [Online]. Available: http://sourceforge.net/projects/strace/

Frank Z. Wang (SM'04) has been with the Cambridge–Cranfield High Performance Computing Facility (CCHPCF), Cranfield, U.K., where he is a Professor and the Chair of e-Science and Grid Computing and the Director of the Centre for Grid Computing since 2004. His appointment is seen as crucial to the initiative of the CCHPCF, which is a collaborative research facility in the Universities of Cambridge and Cranfield, U.K., with an investment size of 40 million GBPs. He serves the High End Computing Panel for Science Foundation Ireland and the U.K. e-Science Panel. He is the Coeditor-in-Chief of the Encyclopedia of Grid Computing and the *International Journal of Grid and High Performance Computing*. He is also on the Editorial Board of another four international journals. He has been invited to report his work at Princeton University, Princeton, NJ; Carnegie Mellon University, Pittsburgh, PA; Oxford University, Oxford, U.K.; Edinburgh University, Edinburgh, U.K.; York University, Toronto, ON, Canada; Manchester University, Manchester U.K.; Hong Kong University of Science and Technology, Kowloon, Hong Kong; National Tsing Hua University, Hsinchu, Taiwan; King's College London, London, U.K.; etc. He has fostered a number of collaborations with industrial giants, including IBM, Microsoft, BBC, Xerox, the European Council for Nuclear Research, Rolls Royce, HP, and the European Aeronautic Defence and Space Company. He has a publication record including an edited book titled *Encyclopaedia of Grid Computing*, 67 journal papers, and 49 conference papers.

Prof. Wang was elected as a Fellow of the British Computer Society and the Chairman (U.K. and Republic of Ireland Chapter) of the IEEE Computer Society, both in 2005.

Na Helian received the Ph.D. degree in computer science from Huazhong University of Science and Technology, Wuhan, China, in 1992.

She has various working experiences in Japan, Singapore, and U.K. She is currently a Senior Lecturer with the School of Computer Science, University of Hertfordshire, Hatfield, U.K. She is the coinvestigator of the U.K. Government Engineering and Physical Sciences Research Council/Department of Trade and Industry grant "Grid-Oriented Storage."

Sining Wu received the Ph.D. degree from the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China, in 2004.

He was involved in the development of 10-TFLops Dawning, listed in the Top 500 Supercomputer List. He is currently a Research Officer with the Centre for Grid Computing, Cambridge–Cranfield High Performance Computing Facility. His current research interests include distributed operating, storage, and file systems.

Yike Guo received the Ph.D. degree in logic and declarative programming from Imperial College London, London, U.K.

He is currently a Professor in computing science with the Department of Computing, Imperial College London. His research is in the areas of parallel applications and network computing including parallel data mining algorithms, distributed data mining systems, decision support systems, and parallel symbolic computation. He is the coinvestigator of the U.K. Government Engineering and Physical Sciences Research Council/DTI grant "Grid-Oriented Storage."

Derek Yuhui Deng received the Ph.D. degree in storage system from Huazhong University of Science and Technology, Wuhan, China, in 2004.

He was with the National Key Laboratory of Data Storage System, China, for almost four years, being involved in three Chinese Natural Science Foundation projects and covering computer architecture, parallel I/O, network storage, virtual storage, etc. He is currently with EMC Corporation, Hopkinton, MA.

Lingkui Meng is currently a Professor and the Deputy Dean of the School of Remote Sensing and Information Engineering, Wuhan University, Wuhan, China. His current research interests include geoscience, grid computing, and distributed systems. He was a Visiting Professor at the Centre for Grid Computing between 2006 and 2007.

Wen Zhang is currently working toward the Ph.D. degree at the School of Remote Sensing and Information Engineering, Wuhan University, Wuhan, China.

She was a Visiting Scholar at the Centre for Grid Computing between 2007 and 2008.

Jon Crowncroft received the B.S. degree in physics from Trinity College, Cambridge University, Cambridge, U.K., in 1979 and the M.Sc. degree in computing and the Ph.D. degree from the University College London (UCL), London, U.K., in 1981 and 1993, respectively.

He is a Marconi Professor of communication systems with the Computer Laboratory, University of Cambridge. He is a Fellow of Wolfson College, Oxford, U.K. Until the end of September 2001, he was a Professor with the Department of Computer Science, UCL. His research interests are communications and multimedia systems, particularly Internet related.

Dr. Crowcroft is a Fellow of the Association for Computing Machinery, the British Computer Society, the Institute for Ethics and Emerging Technologies, and the royal academy of engineering. He is a member of the IEEE Kobayashi Technical Field Award Committee and was the Chair of the IEEE Internet Technical Field Award Committee.

Jean Bacon (F'06) is a Professor of distributed systems with the Computer Laboratory, University of Cambridge, Cambridge, U.K. She authored the *Concurrent Systems* (Addison Wesley) and coauthored the *Operating systems: Concurrent and Distributed Software Design* with Tim Harris.

Prof. Bacon is a Fellow of the British Computer Society. She was elected to the IEEE Computer Society's Board of Governors in 2002–2004 and 2005–2007. She was the Founding Editor in Chief of the IEEE Distributed Systems Online. She supervised the U.K. Government Engineering and Physical Sciences Research Council/DTI grant "Grid-Oriented Storage."

Michael Andrew Parker received the M.A. degree from Oxford University, Oxford, U.K., and the Ph.D. degree from the University of London, London, U.K.

He is currently the Director of the Cambridge eScience Centre, University of Cambridge, Cambridge, U.K. He is responsible for the grid computing initiatives across the university, covering a wide variety of projects in physics, life sciences, earth sciences, medicine, chemistry, and engineering, all requiring the sharing of large computational and data resources. He is with the Management Committees of the Cambridge–Cranfield High Performance Computing Facility.

Authorized licensed use limited to: UNIVERSITY OF HERTFORDSHIRE. Downloaded on May 12, 2009 at 05:04 from IEEE Xplore. Restrictions apply.