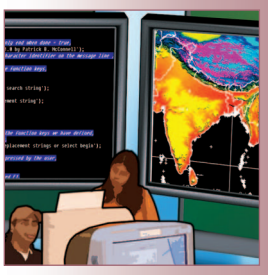

Active Management of Scientific Data

Sophisticated data-distribution schemes and recent developments in sensors and instruments that can monitor the lower kilometers of the atmosphere at high levels of resolution have rapidly expanded the quantity of information available to mesoscale meteorology. The myLEAD personalized information-management tool helps geoscience users make sense of this vastly expanded information space. MyLEAD extends the general Globus Metadata Catalog Service and leverages a well-known general and extensible schema. Its orientation makes it an active player in large-scale distributed computation environments characterized by interacting grid and Web services.

Beth Plale and Dennis Gannon
Indiana University

Jay Alameda, Bob Wilhelmson, Shawn Hampton, and Al Rossi
US National Center for Supercomputing Applications (NCSA)

Kelvin Droegemeier
Oklahoma University

Meteorological research into forecast and prediction models for severe weather is poised for dramatic advances over the next several years. Forecast models have, for some time, been able to utilize data from the 100 or so WSR-88D "Doppler" radars deployed throughout the continental US. The community is now moving toward gathering observational data in real time to obtain forecasts based on up-to-the-minute weather conditions. Additionally, compact high-definition regional sensors that will provide detailed data about the lower 3 km of the atmosphere are now under development. These sensors will generate data at rates an order of magnitude higher than the Doppler radars, thus posing a daunting challenge for researchers to manage the wealth of information. Our group at Indiana University is addressing the data-management aspects of real-time forecasting as part of the larger Linked Environments for Atmospheric Discovery (LEAD) project (http://lead.ou.edu).[1]

A particularly vexing problem is the lack of tools to help researchers make sense of the expanded information space. General search, tagging, and publishing tools let users construct personal, manageable views of the Internet's information space, but the tools in widespread use are poorly suited to managing scientific data for several reasons. First, the Internet's general sharing model is write by a single source and read-only by many. The ability to tag read-only data objects for later access via bookmarks is useful for general Internet use, but scientific users must be able to operate on bookmarked objects and store the derived objects

 1089-7801/05/$20.00 © 2005 IEEE Published by the IEEE Computer Society

with histories, including their origins in the bookmarked objects (that is, their provenance). Second, the Internet's default availability model for data objects (Web pages) is group or world access. In scientific investigation, access must default to user only, but give researchers the option of granting access to a broader audience. Finally, scientific users must be able to control the number, type, and meaning of attributes describing any given data object — information that is highly guarded as trade secrets by today's search engines.

As part of the LEAD project in mesoscale — or regional-scale — meteorology, our group at Indiana University developed myLEAD, an active, personal catalog for managing scientific metadata.

Observational data used as input to the model might come from airplanes, satellites, and balloons, or Doppler radars or sensors.

The tool includes specialized facilities for search, content storage, data-object cataloging, and active engagement, through which users can capture new data objects. MyLEAD extends the Globus Metadata Catalog Service (MCS),[2] a well-known general metadata catalog tool.

Requirements

Indiana University's myLEAD tool is, at its core, a metadata catalog. It stores the metadata associated with data products generated and used in the course of scientific investigation. The data products themselves reside either in the database along with the metadata or in a separate storage repository. The tool's main benefit is that it organizes and keeps track of all relevant information for experimental runs, including users' preferred compute servers, scripts and input files, associated documentation, the generated data products' provenance, and run status. MyLEAD is targeted toward mesoscale meteorology researchers and high school and college students studying weather. The system's requirements are best illustrated with the following simple scenario.

One morning, Ann, a scientist at Oklahoma University, begins research into the supercell (a convective storm with strong vertical winds) that produced 14 tornados across north-central and northeastern Illinois on 20 April 2004. She logs on to her laptop, opens a browser, and brings up the LEAD portal, hosted at Indiana University (http://lead.extreme.indiana.edu:9080/uPortal). Through the site, she clicks on the myLEAD portlet to create an *investigation*, which she names `supercell1Apr04`. Ann then opens another window in the LEAD portal to browse her *myLEAD catalog* for a workflow template that she will customize for this investigation. She searches for the observational files that will initialize the Weather Research and Forecasting (WRF) model,[3] and configures the workflow script to include the data transformations that must take place for model results to be analyzed by a data-mining algorithm as the last step in the workflow.

General Issues

Underlying this simple scenario, a wealth of different data objects must be moved, stored, accessed, and staged. The observational data used as input to the model might come from airplanes, satellites, buoys, and balloons, or it could be real-time streaming data from Doppler radars or sensors. The model might use files from grid-based data-assimilation systems or checkpoints from previous model runs. Then there are the parameter files, data-analysis results, performance histories, and investigation notes that must be captured and stored for later use. Data can be stored in a range of formats, including compressed, binary, image, and text. The meteorologist must be able to search for data sets, convert between formats, and use the new products as input to the model. Finally, the data objects must be private unless the user explicitly chooses to make selected objects available to other researchers.

The key requirements implicit in this example fall into two general categories: system-level and data-model requirements. Each presents distinct challenges that we must integrate into a workable solution in developing a data-management tool.

System-level requirements are driven by the system's service-oriented architecture (SOA). The tool must:

- support concurrency with hundreds to thousands of instances of a personal catalog,
- employ a data replication mechanism to ensure fault tolerance,

- ensure secure access to items in users' catalogs, and
- catalog products in real time as they are generated.

In contrast, data-model requirements derive from the need to provide search, storage, and browsing capabilities. The tool must provide:

- rich search capabilities over personal collections,
- rich metadata descriptions that include application-level characteristics, usage, and provenance, and
- the ability to add attributes seamlessly to existing metadata descriptions.

The personal information-management tool must operate as one component in a loosely coupled, wide-area distributed system. The SOA[4] endorsed by the Global Grid Forum (www.ggf.org) provides one way to organize these wide-area systems. In an SOA, distributed components are coded as Web services, which means they conform to standard ways of communicating, describing themselves to enable automated discovery, and so on. Because myLEAD is built as a Web service, the tool can more easily integrate with the scientific databases and other data resources on the Grid, thus enabling interorganizational groups to leverage each other's data resources.

Data Model Requirements

As an information space grows, the search capability over that space must become richer. For instance, we can accomplish this by scanning text documents and building indexes for them. Given that data products in mesoscale meteorology are generally large binary data files, however, we must use other approaches to enhance search capabilities. One way to do this is to augment typical Unix-type attributes and syntactic metadata (describing how a binary file is laid out) to capture application-specific metadata such as a data product's spatial and temporal aspects.

Meteorological research data resides in multiple forms in numerous repositories served by various kinds of data servers. For example, a data product might be located on an NFS-mounted file system, a high-performance storage server (HPSS), an OpenDAP data server,[5] an HTTP or FTP server, or a local disk mounted from a storage-area network; it might even arrive as a data stream (a time-stamped sequence of events) by means of a data-distribution network. To provide uniform access to products located across heterogeneous storage media, we model myLEAD after such tools as Thematic Realtime Environmental Distributed Data Services (THREDDS),[6] MCS,[2] and Storage Resource Broker (SRB).[7] Each of these tools locates the metadata about all data products in a separate catalog, sometimes stored at separate locations from the actual data objects.

In scientific investigation, a deeper understanding of a data product often emerges only after significant study. Hence, metadata catalogs must provide dynamically expandable descriptions — that is, they must let users add descriptive attributes that are as easily searchable as the original attributes for existing data objects. Indeed, this is one of the MCS tool's strengths, and the primary reason we chose to use it in developing myLEAD.

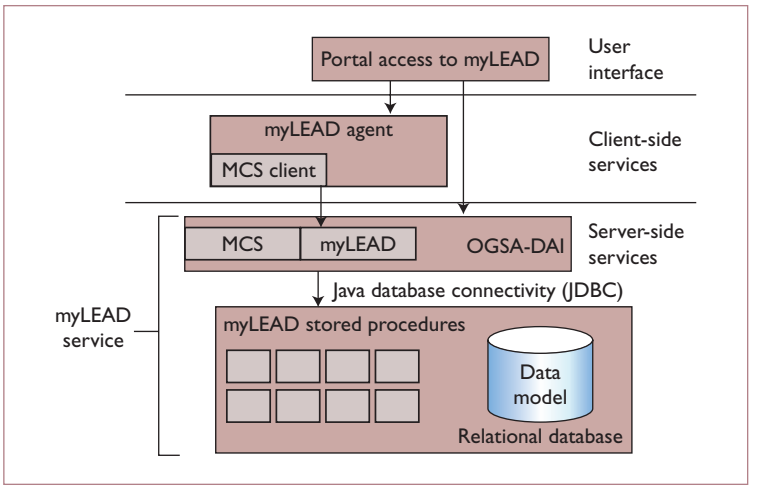

Figure 1. MyLEAD architecture. The myLEAD service is a persistent grid service providing storage and access to a user's information space. The myLEAD agent is a transient Web service that interacts with other services on the user's behalf. The user manages his or her space through the LEAD portal.

MyLEAD Functionality

The myLEAD architecture consists of three primary functional units: a server-side service, a client-side service, and a user interface. Starting at the bottom of Figure 1 and working up, the myLEAD service is a persistent grid service built on top of a relational database management system (RDBMS). The myLEAD service extends the MCS schema and interface; MCS is built on top of the Open Grid Services Architecture Data Access and Integration (OGSA-DAI) Web service[8] — a set of generic grid services that provide access to any database man-

agement system. As Figure 1 illustrates, OGSA-DAI interacts with the database using Java Database Connectivity (JDBC), bringing the DBMS onto the grid by providing a generic query-and-update Web service interface to it.

The MCS catalog service extends OGSA-DAI by providing methods to access particular tables in the database schema. MyLEAD extends the schema further with support for spatial and temporal attributes and for "experiments" as entities. These extensions are supported through added methods for database access (shown as an additional box in the OGSA-DAI rectangle) and making performance improvements over MCS by means of database-stored procedures (shown as cubes to the left of the RDBMS). Eventually, replicated myLEAD servers will be distributed across the LEAD test bed, achieving reliability through a master–slave replication scheme.

The user must have complete control over the mechanisms for publishing data products to the larger community.

The next level up is the myLEAD agent, a transient, short-lived service that serves a single user for a single session. As such, numerous agents will exist at any one time — each tasked with representing a user during an experimental run. As Figure 1 shows, the agent embeds the MCS client, which is a Java interface to the myLEAD service. MyLEAD provides additional functionality on top of the MCS interface for negotiating with other services that carry out the user's experiment.

At the user-interface level, clients (users) interact with myLEAD through the LEAD portal. The tool provides several portlets for managing, browsing, and searching personal information spaces.

MyLEAD Catalog

The myLEAD data model consists of the following elements:

- investigations comprising one or more experiments;
- collections comprising one or more logical files;
- views, or groups of logical files that belong to multiple collections;
- user-defined complex attribute types;
- predefined spatial and temporal attribute types; and
- an abstraction hierarchy that maps from a low-level vocabulary, such as "dx" and "dy" (marking the diagonals of bounding boxes), to higher-order concepts.

MyLEAD supports this extra functionality by extending OGSA-DAI's modular design, which allows extensions by means of additional *activities*. An activity enables state retention across connections to the database, iterative entry building, and complex triggering schemes. By building support for complex geoscience types into the database-management system, we can enforce consistency constraints that ensure its appropriate use, and offer more intuitive querying than MCS could otherwise provide.

MyLEAD also provides higher-level functions over the data model. For instance, it introduces a bulk loading capability, associated with the `create collection` operator, to be used if a collection describes a set of logical files — say, those created as part of a single WRF model run. In one step, myLEAD lets the client create a logical collection in the metadata catalog and register a list of logical files named in the command as members of that collection.

The document interface exported by OGSA-DAI also serves as the interface to the myLEAD metadata catalog. That is, the agent communicates with the server by passing XML documents that encode one or more activities that the server is to perform on the users' behalf. A typical `perform document` might contain a query to the database as the first action, followed by instructions on where to ftp the results. OGSA-DAI supports this functionality through built-in activities for querying databases, exposing management information about databases, and transporting results back to the user.

MyLEAD Agent

The myLEAD agent is a transitory grid service that works on the user's behalf to negotiate with other services in generating metadata and storing, recording, and accessing data products generated and used during investigations. It performs several tasks, including:

- representing the user in negotiations with other Web services (to complete a metadata template to register a new collection, for example),
- initiating file movement to the storage repository,
- ensuring consistency between the storage repository and myLEAD, and
- managing the space available locally to the user.

Although the agent's functionality could be implemented as part of the server-side service and instantiated whenever a user needs to access a database, we believe an agent model enables better extensibility. The agent can take on characteristics other than database access; for example, it can perform rule-based decision-making, thus enabling sophisticated negotiations with other services. This separation of functionality — between a transient client-focused service and a persistent service that provides database-style access — is an important distinction to preserve.

MyLEAD stores metadata about data objects, but the actual data objects reside in separate repositories. A repository could be as simple as a local file system, but an emerging solution is to employ a storage repository such as Replica Locator Service (RLS),[9] Storage Resource Broker (SRB),[7] or Storage Resource Manager (SRM).[10] These repositories generally provide a grouping abstraction (sometimes called a *container*) for storing files and provide location-transparent data storage and access by mapping a physical path name to a location-transparent logical name. Though the myLEAD metadata catalog could work with any of these repository tools, SRM and SRB tightly couple a different metadata catalog to their storage systems.

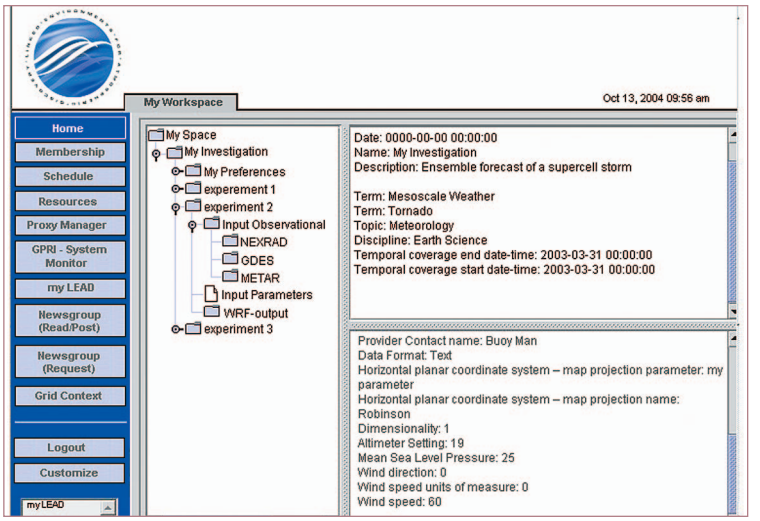

Figure 2. MyLEAD browser in the LEAD portal. The left side of the interface shows a tree organization of the user's space. Clicking an entry in the tree brings up an associated metadata description.

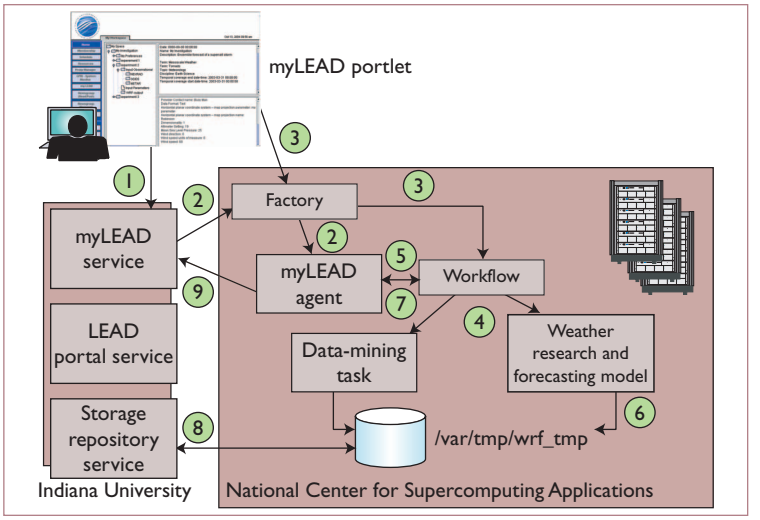

Figure 3. Interface to myLEAD service. A myLEAD agent, acting on behalf of a single user, interacts with other services to catalog data products captured during the investigation.

MyLEAD Portal Interface

Figure 2 shows the LEAD portal — the primary entry point for user access to LEAD services. Scientific investigators working in the portal can use the buttons on the left-hand side to invoke different functionality. The tree in the left pane of the Workspace window depicts the user's information space in hierarchical form, showing three experiments. The second experiment includes three input observational files (`NEXRAD`, `GOES`, and `METAR`), as well as a file containing input parameters. MyLEAD holds the files generated from the model run in a collection labeled `WRF-output`. When the user clicks an entity such as `myInvestigation`, the portal depicts the associated metadata along the right-hand side of the screen (in this case, in the upper box). MyLEAD's browsing capabilities extend earlier work done at Indiana University on myGridContext.[11]

Service Interaction

Figure 3 illustrates the service interactions that take place during an experiment's execution. We use it to demonstrate the control flow between myLEAD's service components in support of a user investigation. The user initiates action by cre-

Related Work in Metadata Management

Several projects have also explored metadata management for scientific user communities working in a Web services environment.

MyLEAD shares much in common with the MyGrid Information Repository,[1] which manages information for bioinformatics researchers using a native XML database. Both systems gather and store information for users while they conduct science experimentation. MyGrid IR emphasizes textual documents, such as index support, because much bioinformatic information is textual, whereas myLEAD primarily supports large binary files. Hence, myGRID IR supports information retrieval on text documents while myLEAD is investigating automated generation of application-domain attributes.

The Network for Earthquake Engineering Simulation Grid (NEESgrid)[2] metadata catalog is built on the Resource Description Framework. RDF's basic building block — the subject-predicate-object — supports the representation of complex relationships between entities at a very fine granularity. Because myLEAD doesn't demand that fine a granularity level, we use a relational data model instead. MyLEAD requires the query efficiencies that SQL achieves because of its strong theoretical foundations.

MySpace[3] is a tool developed for astronomy researchers to manage the large federation of data archives the community shares. It creates "swatches" (shared spaces that cross multiple file systems) shared by a community in the data archives for cached and persistent data and provides common query access over the cached and persistent space, both of which are organized as file systems. Because myLEAD is oriented toward rich application metadata, a file system solution isn't viable.

MyGridContext[4] is an earlier version of myLEAD that managed metadata for a user. MyLEAD expands on the simpler data model of myGridContext by adding support for application metadata, as well as support for controlled publishing of products to a larger community.

References

1. R.D. Stevens, A.J. Robinson, and C.A. Goble, "MyGrid: Personalised Bioinformatics on the Information Grid," *Proc. 11th Int'l Conf. Intelligent Systems for Molecular Biology*, Int'l Soc. for Computational Biology, 2003.
2. J. Peng and K. Law, *Reference NEESgrid Data Model*, tech. report, Network for Earthquake Engineering Simulation Grid (NEESgrid), NEESgrid-2004-40, 2004.
3. C.L. Qin et al., "Myspace: Personalized Work Space in Astrogrid," *Proc. UK E-Science All Hands Meeting*, University of Southampton, 2003, pp. 361–365; www.allhands.org/uk/2003/proceedings.
4. D. Gannon et al., "Building Grid Portal Applications from a Web-Service Component Architecture," to appear in *Proc. IEEE*, IEEE Press, 2005.

ating a new experiment at the portal (as when Ann created `supercell1Apr04` in the earlier example). The portal notifies the myLEAD catalog service, which creates a new experiment entry in the metadata catalog. From that point on, all data products used or created in that experiment are recorded as associated with it until the experiment terminates or the scientist switches to another experiment (step 1).

When the user selects a host site for executing the WRF model — in this example, the US National Center for Supercomputing Applications (NCSA) — the catalog service spawns a myLEAD agent (step 2) to act on the user's behalf throughout the session. A factory at NCSA (the site at which the user will be initiating model runs) creates the myLEAD agent to represent the user during the capture, storage, and retrieval of metadata.

Once the experiment is set, the user constructs the workflow visually through the portal by graphically connecting computational components, searching for new input files, and graphically linking the resulting filenames to the computational elements' input sources. Once the workflow script is configured, the user launches it at the designated site (step 3).

The simple script in our example runs a single instance of the WRF model for some number of time steps before invoking a postprocessing data-mining algorithm on the resulting files. The script's sequenced execution is step 4 in the figure.

When the model is under a workflow script's control, the script makes the decision regarding where to store the output files generated during this model's execution. Given that postprocessing of the files is immediate (that is, data mining is the next major step in the script), the workflow can store them as intermediate files to temporary space at NCSA. To determine where such space exists, the workflow consults the active myLEAD agent (step 5), which looks up the proper location and responds to the script (step 6) — in this case, to `/var/tmp/wrf_tmp`. Today, meteorologists hard-code into the input parameter file the path name to the scratch space on their favorite machines. As computations move onto widely distributed grid platforms, however, information such as file system paths must be determined dynamically.

Next, the script executes the data-mining task on the results. The myLEAD agent interacts with the workflow to ensure that the analysis results are recorded to myLEAD and that the analysis file is

moved to a persistent store (step 7). The workflow passes the completed metadata description and a pointer to the analysis file to the myLEAD agent, which invokes an FTP service to move the file to the storage repository service (step 8). The myLEAD agent then writes the resulting logical name into the metadata description and adds the entry to its metadata catalog (step 9).

We demonstrated myLEAD at Supercomputing 2004 as part of the LEAD portal demo, but it is a work in progress. By leveraging existing projects (MCS and OGSA-DAI), we quickly developed a prototype implementation to elicit early feedback from scientists and move on to more challenging issues. Future work will focus on four key areas:

- immutable experiments,
- agent services,
- publishing, and
- building trust.

Immutable experiments allow users to reuse experiments, in full or part, without destroying the integrity of earlier investigations. The proactive agent infers new metadata from the context of active experiments through case-based reasoning. The user must have complete control over the mechanisms for publishing data products to the larger community. The system must also convey visual cues that build the user's trust — perhaps by guiding the user through a set of Web pages in a portlet that convey a sense of entering a secure space. Publishing of data products would then be allowed only within that secure space. Support for these four features will complete the base functionality of the system. An initial release is planned for Spring 2005.

Acknowledgments

The lead author thanks the LEAD team members — particularly, Sara Graves and Steve Tanner of the University of Alabama, Huntsville; Scott Jensen who developed much of the myLEAD data model; and Ann Chervenak, of the University of Southern California's Information Sciences Institute, and Neil Chue Hong and Mario Antonnelli of the Edinburgh Parallel Computing Center at University of Edinburgh, who have been invaluable in our gaining a deeper understanding of MCS and OGSA-DAI, respectively.

References

1. K.K. Droegemeier et al., "Linked Environments for Atmospheric Discovery (LEAD): A Cyberinfrastructure for Mesoscale Meteorology Research and Education," *Proc. 20th Conf. Interactive Information Processing Systems for Meteorology, Oceanography, and Hydrology*, Am. Meteorological Soc., 2004.
2. G. Singh et al., "A Metadata Catalog Service for Data-Intensive Applications," *Proc. ACM/IEEE Supercomputing 2003* (SC 03), IEEE CS Press, 2003, pp. 33–49; http://csdl.computer.org/comp/proceedings/sc/2003/2113/00/21130033abs.htm.
3. J. Michalakes et al., "Development of a Next Generation Regional Weather Research and Forecast Model," W. Zwieflhofer and N. Kreitz, eds., *Developments in Teracomputing: Proc. 9th ECMWF Workshop on the Use of High Performance Computing in Meteorology*, World Scientific, 2001, pp. 269–276.
4. I. Foster et al., "The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration," white paper, Globus Project, 2002; www.globus.org/research/papers/ogsa.pdf.
5. J. Gallagher and G. Milkowski, "Data Transport within the Distributed Oceanographic Data System," *Proc. 4th Int'l World Wide Web Conf.*, World Wide Web Consortium, 1995; www.w3.org/Conferences/WWW4/Papers/67/.
6. B. Domenico et al., "THREDDS: Incorporating Real-Time Environmental Data and Interactive Analysis Tools into NSDL," *J. Digital Information*, vol. 2, no. 4, 2002, article 114.
7. A. Rajasekar, M. Wan, and R. Moore, "MySRB and SRB — Components of a Data Grid," *Proc. 11th IEEE High Performance Distributed Computing* (HPDC), IEEE CS Press, 2002, pp. 301–310.
8. M. Antonioletti et al., "Experiences of Designing and Implementing Grid Database Services in the OGSA-DAI Project," *Proc. Global Grid Forum Workshop on Designing and Building Grid Services*, Global Grid Forum, 2003; http://www.ogsadai.org.uk/docs/OtherDocs/OGSA-DAI-dbgsws-v1.0.pdf.
9. B. Allcock et al., "Secure, Efficient Data Transport and Replica Management for High-Performance Data-Intensive Computing," *Proc. IEEE Mass Storage Conf.*, IEEE CS Press, 2001, pp. 13–28.
10. A. Shoshani, A. Sim, and J. Gu, "Storage Resource Managers: Middleware Components for Grid Computing," *Proc. 10th NASA Goddard Conf. Mass Storage Systems and Technology*, IEEE CS Press, 2002.
11. S. Shepler et al., *NFS Version 4 Protocol*, IETF RFC 3010, Dec. 2000; www.rfc-editor.org/rfc/rfc3010.txt.

Beth Plale is an assistant professor in the Computer Science Department at Indiana University. Her research interests include data management, high-performance computing, and grid middleware. Plale received a PhD in computer science from the State University of New York, Binghamton, and performed her postdoctoral work at the Georgia Insti-

tute of Technology. She is a member of the ACM and the IEEE. Contact her at plale@cs.indiana.edu.

Dennis Gannon is a professor in the Computer Science Department at Indiana University. His research interests include high-performance distributed computing and problem-solving environments for computational science. Gannon received a PhD in mathematics from the University of California, Davis, and a PhD in computer science from University of Illinois. He is the science director for the Indiana Pervasive Technology Labs. Contact him at gannon@cs.indiana.edu.

Jay Alameda is a research scientist at the US National Center for Supercomputing Applications. His research interests include chemical engineering, grid computing, and high-performance computing. Alameda received a master's degree in chemical engineering from University of Illinois, Urbana-Champaign. Contact him at jalameda@ncsa.uiuc.edu.

Bob Wilhelmson is a professor of atmospheric sciences at University of Illinois, Urbana-Champaign, and chief scientist at NCSA. Wilhemson received a PhD in computer science from the University of Illinois Urbana-Champaign. Contact him at bw@ncsa.uiuc.edu.

Shawn Hampton is a research programmer at NCSA. His research interests include grid computing, Web services architectures, and user interface architectures. Hampton received a BSc in computer science from the University of Illinois, Urbana-Champaign. Contact him at Hampton@uiuc.edu.

Al Rossi is a research programmer at NCSA. His research interests include grid computing environments, workflow management and orchestration, data management, data mining, and programming languages. Rossi received a PhD in comparative literature from Princeton University and an MS in computer science from Indiana University. Contact him arossi@ncsa.uiuc.edu.

Kelvin Droegemeier is Regents Professor of Meteorology and Roger and Sherry Teigen Presidential Professor at Oklahoma University. Droegemeier received a PhD in atmospheric science from the University of Illinois, Urbana-Champaign. He cofounded the Center for Analysis and Prediction of Storms (CAPS) and the Collaborative Adaptive Sensing of the Atmosphere (CASA). Contact him at kkd@ou.edu.