Virtual Data Grid Middleware Services for Data-Intensive Science
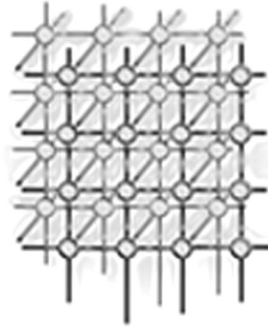
Yong Zhao[*1], Michael Wilde[2], Ian Foster[1,2], Jens Voeckler[1],
James Dobson[3], Eric Glibert[4], Thomas Jordan[4],
and Elizabeth Quigg[4]

[1] *Department of Computer Science, University of Chicago*
[2] *Math and Computer Science Division, Argonne National Laboratory*
[3] *Department of Psychological & Brain Sciences, Dartmouth College*
[4] *Fermi National Accelerator Laboratory*

SUMMARY

The GriPhyN Virtual Data System provides a suite of components and services for data-intensive sciences that enables scientists to systematically and efficiently describe, discover, and share large scale data and computation resources. We describe the design and implementation of such middleware services in terms of a virtual data system interface called Chiron, and present virtual data integration examples from the QuarkNet education project and from functional-MRI-based neuroscience research. The Chiron interface also serves as an online "educator" for virtual data applications.

KEY WORDS: Virtual Data, Workflow, Data Grid, Data Integration, Portal, Provenance

Introduction

Next-generation data-intensive applications pose unprecedented demands on information technology. Communities of users distributed worldwide need to extract information from large collections of data, and would like to share and reuse both data products and the resources needed to produce and store those data products [1, 2].

So-called virtual data mechanisms virtualize data with respect to location, representation, and materialization. A common data model is used to describe data types and representations, and the recipes for derivations of data are specified in a declarative manner. Requests for data products can be transparently mapped into computation and/or data access operations across multiple Grid computing and storage locations [3].

*Correspondence to: yongzh@cs.uchicago.edu

The GriPhyN Virtual Data System, VDS, [4] implements such virtual data mechanisms. Definition and query operations are expressed in a virtual data language (VDL), and information about data and computational procedures is stored in a virtual data catalog (VDC).

VDS has been applied to a variety of data-intensive applications, including physics event simulation [5], galaxy cluster finding [6], genome analysis [7], and biomedical image analysis [8]. However, the initial virtual data interfaces (a Java API with command-line wrappers), while powerful for scripting-level integration, proved cumbersome for both deeper application integration and direct end-user use. Many user communities want both to integrate virtual data capabilities more deeply into science applications and to encapsulate the tasks of setting up and configuring the virtual data system and its associated grid compute and storage resources. End users need an interactive environment in which they can easily discover and share virtual data products, compose workflows, and monitor workflow executions across multiple grid sites.

These considerations have led us to develop, apply, and evaluate a set of interfaces that provide more powerful access to virtual data system data structures. On top of these, we developed the Chiron portal for convenient interaction with the virtual data system. Chiron allows users to:

- manage user accounts and track user activities;
- describe and publish applications and datasets;
- discover, validate, share and reuse applications and datasets published by other users;
- configure and discover Grid resources;
- compose workflows and dataset derivations;
- execute workflow either locally or in a Grid environment and monitor job progress; and
- record and query provenance information.

With these enhancements, the virtual data system provides both user-level and service (middleware)-level functionality that portal users can integrate to build customized virtual data interfaces for specific user communities and science applications.

The rest of this paper is organized as follows. We first briefly review the GriPhyN Virtual Data System and Virtual Data Language and then describe the Chiron system architecture. Next, we talk about the dimensions of the virtual data discovery space and then present virtual data integration examples from the QuarkNet project and from functional MRI (Magnetic Resonance Imaging) experiments. We also describe our work on dataset type and dataset iteration, and lastly we compare with related work and draw our conclusions.

This paper builds on and extends an earlier paper [9], updating its description of various system components and detailing the mechanisms used to organize and access datasets.

VIRTUAL DATA SYSTEM

VDS supports the description, discovery, and reuse of information on how data is derived by computations. A common data model is used to describe datasets and procedures. This data model defines representations for information about datasets, transformations, derivations, invocations, and metadata.

A dataset is a unit of data managed by VDS, and is associated with a type. A transformation (TR) is a typed definition for a computation procedure, which takes certain types of datasets as inputs and outputs. A compound TR can be defined to comprise other TRs in an acyclic fashion; a derivation (DV) instantiates a TR by giving the actual arguments and other necessary information, a DV record can serve both as a historical record of what was done and as a recipe for operations that can be performed in the future; an invocation (IV) is the actual execution of a DV. Datasets can be replicated at multiple locations for performance and availability purposes, and IV keeps track of the specific replicas used or produced in the execution. Datasets and transformations can be annotated with metadata (attributes) to facilitate discovery, access and composition.

Virtual Data Lifecycle

The virtual data lifecycle begins when a virtual data definition, described in VDL, is entered in the VDC. This definition can then be discovered by virtual data query operations, and the associated scientific analysis procedures can be executed and validated. New algorithms and procedures can be derived from the knowledge gathered; another round of the lifecycle starts when the derived virtual data is published into the VDC or recorded by VDS.

Virtual Data API

In our earlier development of VDS, we only provided users with a limited set of command-line interfaces. These interfaces limited the applicability of the system since only specific functionalities were exposed. Subsequently, we have defined and developed a complete set of Java APIs so that users can have fine-grained control over the system, integrate VDS with their own applications and analysis environments, and build virtual data services and portals based on the APIs.

This virtual data API provides support for:

- system property setup such as database backend/driver configuration, schema locations, and logging;
- VDL conversion and parsing, i.e., conversion between textual and XML syntaxes;
- VDC manipulation, i.e., support for inserting, updating, deleting, and searching virtual data in the VDC;
- querying and editing the catalogs that VDS maintains for mapping logical to physical entities;
- Grid resource management;
- workflow composition, execution, and monitoring; and
- provenance tracking.

These interfaces allow customized implementations for certain components. For example, we implement a number of database backends that variously support file-based, relational database, and native XML database storage, all conforming to the database interface. Similarly, the site selection interface allows developers to enforce Grid site selection policies. The API
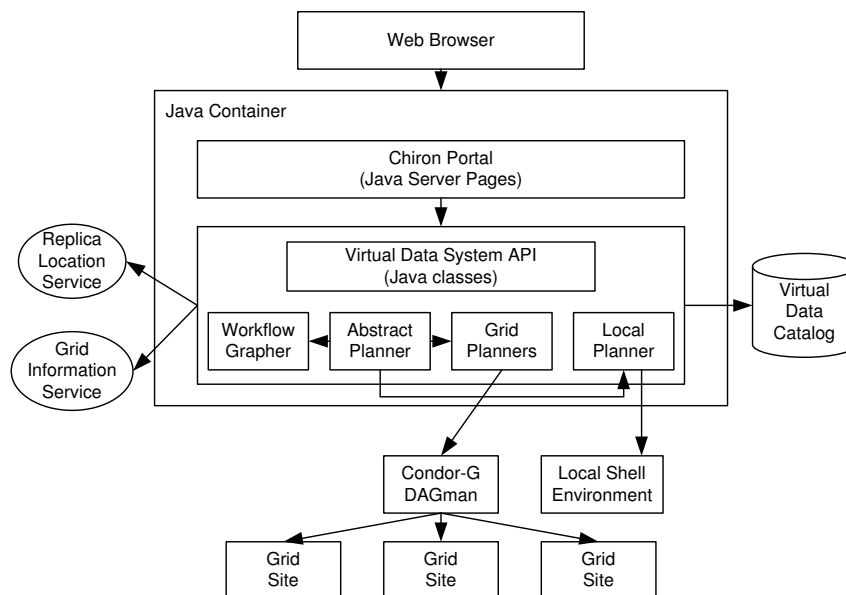
Figure 1. Chiron System Achitecture

also makes it easy to develop and deploy portals, and to integrate VDS into specific scientific domains.

CHIRON SYSTEM ARCHITECTURE

The Chiron portal is an example of a system implemented in terms of our virtual data system APIs. The Chiron implementation combines those APIs and libraries with standard commodity Web technologies, databases, and Grid technologies. We show the system architecture in Figure 1.

A Chiron client only needs a Web browser that supports SSL for security, DHTML and Java scripts for interactivity, and cookies for customization. The Chiron server comprises an Apache Tomcat Web server, Java Server Pages (JSP), the virtual data system API and libraries (jar files), backend databases for the virtual data system, GraphViz for graph visualization, workflow engines for local and Grid execution, and Condor-G [10] for Grid job submission and execution.

Chiron also uses the open source Apache Axis SOAP toolkit to provide SOAP-based Web services interfaces to the underlying virtual data system. Chiron is configured to start Axis automatically and the Axis servlet listens on http://localhost:8080/chiron/services.

System Configuration

The Chiron portal code and libraries are deployed into a Tomcat servlet engine. Chiron extends the VDS "properties" file to configure various runtime parameters, such as database driver and schema, MDS (monitoring and discovery service) [11] and RLS (replica location service) [12] host information, file transfer mode, logging options, and scheduler-specific configuration. Backend database support is provided by a uniform data management interface on top of JDBC that leverages Java reflection and dynamic class loading to make multiple database management systems available transparently to the higher-level code.

User Management

All portal users must create a user account and should have a valid Grid certificate. We use user ID and password authentication via secure HTTP connections, so that the client side only needs a Web browser to access the portal. Different users are assigned different roles at the portal, such as guest, administrator, and super user, and different levels of controls are associated with each different role. The portal also keeps track of sessions and user preferences. User activities are tracked as provenance data in the VDC, where analysis patterns could be uncovered by leveraging data mining approaches.

Job Submission

Once users have registered and published data derivation recipes in the VDC, they can request that these virtual datasets be materialized. We term the process of mapping these requests to the execution of computation procedures "planning" as it is suggestive of query planning in database systems. Following a request, the virtual data system finds all datasets and procedures required to derive the requested dataset, and generates an "abstract" derivation workflow - abstract in the sense that it is not tied to specific executables, dataset replicas, or grid sites. This workflow is represented in an XML format that can be interpreted and executed by different planners and workflow engines.

Chiron supports two different planners. The *local planner* takes a workflow definition and produces a multi-level shell script that invokes the required graph of programs. This planner is designed to run derivations on a single host and thus is useful for initial debugging of applications and for determining whether specific errors are caused by the Grid, virtual data system, or the application.

The *Pegasus Grid planner* [13] uses information from the Grid information service (MDS) and replica location service (RLS) to convert the abstract plan into a concrete execution plan. That concrete plan is then submitted to DAGMan and executed on remote Grid sites via Condor-G and Globus. A Grid site usually has compute elements (CEs) managed by a local scheduler (Condor, PBS, LSF, etc), and storage elements (SEs) accessed via GridFTP.

The portal also displays execution progress and job status once the workflow is converted into a concrete plan and executed. Upon successful completion, requested datasets can be displayed as images or text, if their format permits.

Workflow Visualization

Chiron uses GraphViz, a graph drawing tool developed by AT&T Laboratories, to visualize workflows. The XML representation of a workflow is first converted into GraphViz's "dot" format, and then transformed into a displayable image. Some sample workflows are presented in Figure 3 and Figure 4. Parallelograms and ellipses represent datasets and computation procedures, respectively. When displayed in Chiron, a user can click on a dataset node to show all derivations that process the dataset; clicking on a procedure node displays detailed information about the corresponding transformation. These graphs provide insights into both individual steps and the workflow process in the large.

Web Services Interface

While the VDS Java API provides a powerful interface, it is suitable only for programatic integration. However, many science communities require deeper integration of virtual data capabilities into their applications and problem-solving environments, and the encapsulation of system setup and grid resource configurations. The Chiron servlet and JSP scripts expose higher level functionalities and facilities for portal and Web interface development.

Chiron also provides a Web service interface for more flexible, loosely-coupled integration. This interface is provided via an embedded Apache Axis toolkit. A set of virtual data Web services is deployed in the Axis engine, which provides the same functionalities as the JSP pages for virtual data discovery, composition and integration. The service client stub classes have been generated from the WSDL service description and distributed with the portal, but other Web services invocation mechanisms can be used to communicate with the services.

This combination of VDS API, service-oriented JSP scripts, and Web services interface allow the Chiron portal to provide science communities and applications with a suite of Grid middleware services, of different levels and granularities, to explore the power of virtual data and Grid computing.

VIRTUAL DATA DISCOVERY

The two keys to virtual data reuse and sharing are efficient description and discovery. To this end, Chiron allows users to search the VDC for virtual data entities along three different dimensions: namespace and/or attributes, metadata annotations, and provenance history. Definition namespaces organize virtual data procedures into a structured hierarchy; metadata associates arbitrary annotations with virtual data products; provenance information records the derivation lineage of virtual data products.

Definition

A definition is a declarative specification of some data analysis procedure. It can be either a transformation ("TR" in VDL), which is similar to a function definition in that it defines
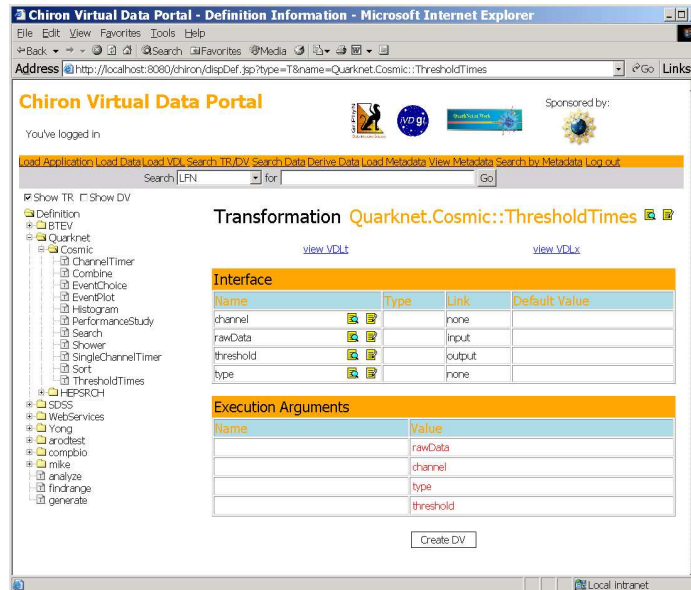
Figure 2. Chiron Screenshot

a function name and formal parameters, or a derivation ("DV" in VDL), which acts like a function call, with the actual parameters (scalars and data files) identified [3, 4].

Definitions can be organized hierarchically using namespaces, which might be associated with, for instance, institutions, projects, groups, or users. Chiron displays VDL definitions in a tree view that can be customized to show namespaces, transformations, and/or derivations (Figure 2). A user can explore different namespaces and display the detailed definitions of each transformation and derivation by exploring the tree. Definitions can also be shown in VDL [3, 4] for advanced users.

Chiron also provides a search bar for quick exploration. A user can search for transformations, derivations, datasets, and metadata using keyword and wildcard searches. These definitions provide an interface-level understanding of data analysis procedures.

Metadata

Metadata annotation is essential to facilitate the discovery and understanding of the datasets and procedures stored in the VDC. We allow annotations to be associated with five categories of virtual data objects: datasets, transformations, derivations, parameters, and the nested calls in compound transformations. (A compound tranformation consists of a graph of calls to other transformations.) Annotations are represented by tuples of the form (attribute_name, type, attribute_value), where type can be integer, float, string, date, or boolean.

Metadata about a dataset might include cataloging information, such as description, creator, creation date, and publisher; content information such as data format, structure, and size; and provenance information, i.e., how data is acquired, transformed, and transferred. For code and procedures, besides information about their inputs and outputs, and execution environments, metadata might describe functionality, prerequisites, and parameter constraints.

We define a metadata query language to search for virtual data objects that have specific metadata. A user does not need to specify the types of the attributes being searched: a query parser takes care of type conversion and parses it into an abstract syntax tree, which is then optimized and translated into SQL or, if available, XML XPath or XQuery statements. The query statements are submitted directly to the backend database and the results are interpreted and then returned in the appropriate format.

The Chiron portal provides Web interfaces for associating metadata with different virtual data objects, displaying metadata about a specific virtual data objects, and searching for virtual data objects with specific metadata.

Provenance

Provenance information captures the derivation history of data [14]. In our case, it consists of an "invocation" record for a data derivation, which records when, how, and where the data is produced, what kind of resources are used, and so on. Combined with the abstract workflow produced for the data derivation, provenance information gives the audit trail of the data production process. A user can use this information to repeat a derivation, to explain or validate a result, and so forth.

An invocation record is recorded for each task that is executed within a workflow. These records can be used to provide reports on data quality, run time, resource consumption, or even computation anomalies; to re-execute past workflow or to modify/improve future workflow; and to answer ad-hoc queries.

VIRTUAL DATA INTEGRATION

Science communities and scientific applications must often integrate virtual data into an existing analysis environment, a portal or a Web interface. Chiron facilitates the development of application-specific portals and Web interfaces, and provides reusable core services for accessing virtual data and Grid resources. We illustrate virtual data integration scenarios drawn from the QuarkNet science education project and a neuroscience research project.

QuarkNet Cosmic Ray Application

The Chiron portal is being applied in the QuarkNet project [15], an NSF- and DOE-funded project that aims to create a research community of physicists, high school teachers and their students. QuarkNet engages teachers and students in scientific investigations about the structure of matter and the fundamental forces of nature. Students learn fundamental physics as they analyze live, online data and participate in inquiry-oriented investigations, and teachers

join research teams with physicists at local universities or laboratories. Currently, the project involves six large experimental physics collaborations, 52 research groups, 158 physicists, 514 high school teachers, and thousands of students.

The QuarkNet Cosmic Ray Project is an experimental activity designed to give students hands-on experience in scientific collaboration and discovery. Students learn to construct, deploy and test cosmic ray detectors, and analyze real-time data gathered by the detectors. About 40 detectors have been installed in high school classrooms across the U.S. QuarkNet helps students understand and explore the origin of highly energetic cosmic rays, the source of their energy, and the clues they hold about the early universe.

QuarkNet has created a specialized Web site and Web interfaces for science education [16]. This portal, driven by virtual data middleware, allows students to launch, configure and control remote applications as though they are using a desktop computing environment. The key requirements for the Web interfaces are to give students the means to a) discover and apply datasets, algorithms, and data analysis methods; b) collaborate by developing new transformations and by sharing results and observations; and c) learn data analysis methods that will motivate and prepare them for a scientific career. These requirements match those for Chiron, but here it is the Web applications that need to interact with Chiron. Thus, we have encapsulated relevant Chiron portal functions into APIs so that these Web applications can integrate virtual data mechanisms seamlessly into their framework.

For a specific data analysis task, students interact directly with the QuarkNet Web interface, which in turn accesses Chiron services to upload data and Perl code, process data using simple form-based tools, annotate and share data and results, and create posters to describe and share analysis processes. Students can use metadata queries to discover each other's workflows and data products.

In Figure 3, we show the customized Web page for a cosmic-ray-shower study. This study involves a data analysis process that combines detector and GPS data gathered from distributed geographical locations and determines whether a cosmic ray shower has occurred in the sampled period and location. The plot is generated by clicking the "analyze" button, which sends the request to the derivation script in Chiron. The script first generates the abstract workflow (shown to the right in Figure 3), then calls the shell planner to create a concrete, executable workflow, and finally executes the corresponding graph of analysis code. A link to a visual rendering of the workflow graph is returned. These steps all occur transparently: the user only interacts with the QuarkNet Web page.

Chiron also provides mechanisms to generate application-specific Web pages automatically, by using HTML templates, transformation definitions, and annotations. For example, the title of a transformation can be rendered into the title of the Web page, default/eligible values for parameters can be rendered as options, input datasets can be filtered by dataset types, and output filenames can be generated by context.

Cognitive Neuroscience Application

Automated data provenance and workflow management is a critical requirement for the functional MRI (fMRI) research community [17]. Our virtual data middleware has been used to create population-based atlases from the fMRI Data Center's archive [18] of high resolution
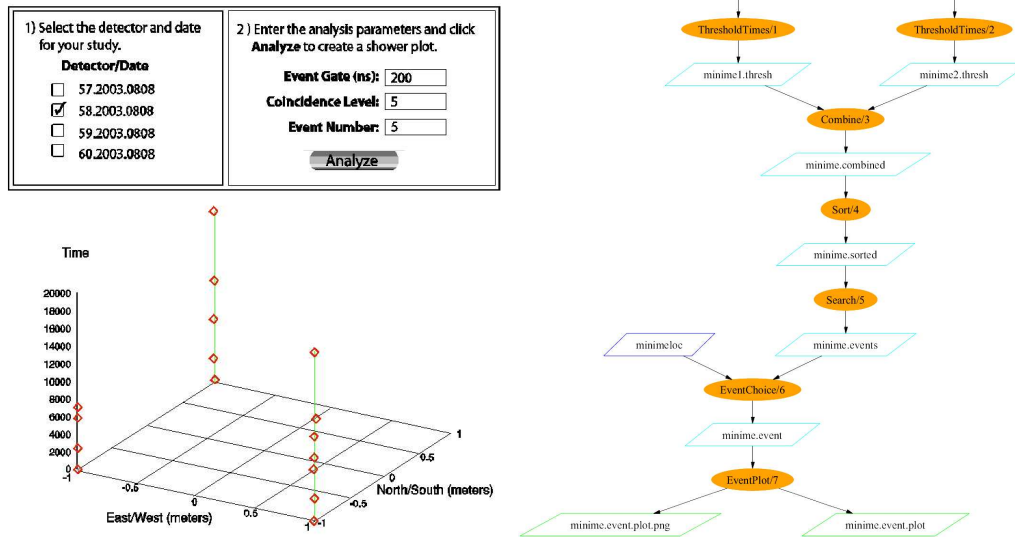
Figure 3. Cosmic Ray Shower Study

anatomical MR data, using a multi-stage I/O intensive pipeline that "warps" the anatomical features of the brains of multiple subjects into a standard space.

The VDL workflow for the creation of such a brain "atlas" (Figure 4) involves several derivations that employ the AIR (automated image registration) suite [19, 20] to create an "averaged" brain from a collection of high resolution anatomical data. The workflow warps each anatomical volume to a reference volume, using a high-order nonlinear multi-parameter model. The brain structures are warped to define a similar shape, then each volume is resliced using the warp values, and the resulting high resolution volumes are averaged using the AIR suite's "softmean" application.

In the context of this atlas-generation use case, we experimented with applying the middleware services of the virtual data system to various science-driven discovery problems. A few of these are illustrated in the following queries and corresponding responses.

Which TRs can process a subject image?

```
Q: xsearchvdc -q tr_meta dataType subject_image input
A: fMRIDC.AIR::align_warp
```

Which TRs can create an ATLAS?

```
Q: xsearchvdc -q tr_meta dataType atlas_image output
```
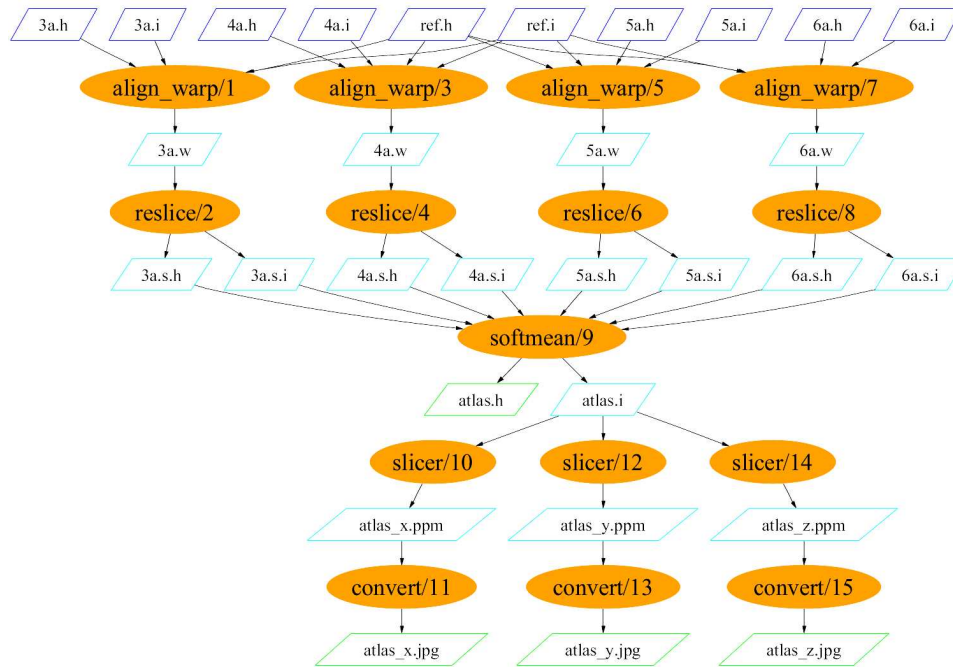
Figure 4. fMRI Workflow

A: fMRIDC.AIR::softmean

List anonymized subject-images for young subjects. This query searches for files based on their metadata.

```
Q: xsearchvdc -q lfn_meta dataType subject_image privacy anonymized
                subjectType young
A: 3472-4_anonymized.img
```

For a specific patient image, 3472-3, show all DVs and files that were derived from this image, directly or indirectly.

```
Q: xsearchvdc -q lfn_tree 3472-3_anonymized.img
A: 3472-3_anonymized.img
   3472-3_anonymized.sliced.hdr
   ...
   3472-3_anonymized.sliced.img
```

Similar queries can be used for data validation processes, and to construct scripts that compose new virtual data workflows based on existing ones, and then catalog, run, and record them.

MANAGING DATASET COLLECTIONS

A commonly observed pattern in scientific computation is that a procedure to process a large dataset is usually specified by higher-level scripts that iterate over the dataset, applying a transformation to each sub-component. For example, in the fMRI spatial normalization application (AIRSN), a single study consists of groups of subjects, each subject has a series of fMRI scans (runs), and each run contains a series of images (volumes), where a script would iterate over each volume in a run, each run in a subject, and each subject in a group.

Another observation is that data are often represented in different physical formats, which may change over time. Computational procedures designed for a given format may no longer work correctly once formats change. Such format dependency prevents code reuse over datasets that are conceptually identical but have different physical representations.

To address these two issues, the XML Dataset Typing and Mapping (XDTM) model [21] was developed. XDTM allows virtual datasets to be defined, and procedures written that select subsets of the datasets and iterate over those subsets, in a manner that is independent of the datasets' concrete physical representations. XDTM employs a two-level description of datasets, defining separately via a type system (XML Schema) the abstract structure of datasets, and the mapping of that abstract data structure to physical representations. The use of XML Schema as a type system has the benefit of supporting the same powerful standardized query language that we use in our selection methods.

We have implemented a prototype of the XDTM model by extending the current VDS, and successfully applied the system to the fMRI AIRSN workflow.

RELATED WORK

The GridAnt [22] system provides a flexible workflow model based on Jakarta Ant and Grid execution. However, it does not address provenance tracking, planning support, or the declarative representation of application logic, nor does it contain the sophisticated mechanisms that can transparently perform a transformation on a set of input data objects in a location-independent manner, anywhere in a distributed Grid or local desktop environment.

The SAM system deployed by the Collaboration for Multiscale Chemical Science [23] is based on the uniform action model of the WebDAV protocol [24], and has a well-developed "laboratory notebook" model of provenance tracking. However, its middleware approach for Grid execution has the same location-dependence limitations as GridAnt.

The Grid Access Portal for Physics Applications [25] supports job submission, job status checking, and resource management. Each application is packaged as a *notebook* comprising Web pages and editable parameterized scripts. Notebooks can be published and stored in Web based archives for others to retrieve and modify. The notebook concept is similar to our

transformation and derivation descriptions, but our virtual data descriptions are more general and are decoupled from physical entities to support community-wide sharing and reuse.

The MyGrid project [26] has developed powerful distributed execution semantics and enactment engines, visual workflow specification capabilities, and provenance tracking. However, these capabilities are only provided for the relatively "regular" execution model of Web services, but not for the standard application programs that constitute the majority of today's scientific computing toolkits.

CONCLUSIONS

We have described a virtual data system interface, Chiron, that allows users to describe, discover, share, and reuse scientific processes and data. Chiron provides user-level Web interfaces, service-oriented scripts, and Web services-based middleware functionalties that facilitate integration of virtual data mechanisms into specialized data analysis applications and problem solving environments. Chiron has been applied successfully within the QuarkNet education project and tested in the context of a functional MRI research project.

Chiron also serves as a virtual data training tool. We have loaded procedures and workflows from several virtual data applications, including high energy physics, cosmic ray detection, bioinformatics, and fMRI, as examples into the portal, thus allowing first-time and novice virtual data users to explore core virtual data concepts and technologies without being distracted by software configuration details.

We have also extended the virtual data language to use XDTM mechanisms to abstract dataset types from their physical representations, and to enable selection and iteration on large composite datasets.

In future work we plan to leverage knowledge representation techniques to enhance process definition and metadata annotation, and to enable semantics-driven virtual data discovery, composition and integration.

REFERENCES

1. Avery P and Foster I. The GriPhyN Project: Towards Petascale Virtual Data Grids. **http://www.griphyn.org**, 2001.

2. Chervenak A, Foster I, Kesselman C, Salisbury C, and Tuecke S. The Data Grid: Towards an Architecture for the Distributed Management and Analysis of Large Scientific Datasets. *Journal of Network and Computer Applications*, 23:187–200, 2001.

3. Foster I, Voeckler J, Wilde M, and Zhao Y. The Virtual Data Grid: A New Model and Architecture for Data-Intensive Collaboration. In *First Biennial Conference on Innovative Data Systems Research*, Jan 2003.

4. Foster I, Voeckler J, Wilde M, and Zhao Y. Chimera: A Virtual Data System for Representing, Querying, and Automating Data Derivation. In *14th Conference on Scientific and Statistical Database Management*, 2002.

5. Arbee A, Avery P, Bourilkov D, Cavanugh R, Katageri S, Rodriguez J andGraph G, Voeckler J, and Wilde M. Virtual Data in CMS Productions. In *Proceedings of Computing in High Energy and Nuclear Physics*, March 2003.

6. Annis J, Zhao Y, Voeckler J, Wilde M, Kent S, and Foster I. Applying Chimera Virtual Data Concepts to Cluster Finding in the Sloan Sky Survey. In *SC2002*, 2002.

7. Rodriguez A, Sulakhe D, Marland E, Nefedova N, Wilde M, and Maltsev N. Grid Enabled Server for High-Throughput Analysis of Genomes. In *Workshop on Case Studies on Grid Applications*, March 2004.

8. Leung K, Heckemann R, Saeed N, Brooks K, Buckton J, Changani K, Reid D, Ruckert D, Hajnal J, Holden M, and Hill D. Analysis of serial MR images of joints. In *IEEE International Symposium on Biomedical Imaging (ISBI)*, pages 221–224, 2004.

9. Zhao Y, Wilde M, Foster I, Voeckler J, Dobson J, Jordan T, and Quigg E. Grid Middleware Services for Virtual Data Discovery, Composition, and Integration. In *Proceedings of the 2nd workshop on Middleware for grid computing*, 2004.

10. Frey J, Tannenbaum T, Foster I, Livny M, and Tuecke S. Condor-G: A Computation Management Agent for Multi-Institutional Grids. *Cluster Computing*, 5(3):237–247, 2002.

11. Czajkowski K, Fitzgerald S, Foster I, and Kesselman C. Grid Information Services for Distributed Resource Sharing. In *Proceedings of the Tenth IEEE International Symposium on High-Performance Distributed Computing (HPDC-10), IEEE Press*, August 2001.

12. Chervenak A, Deelman E, Foster I, Guy L, Hoschek W, Iamnitchi A, Kesselman C, Kunst P, Ripeanu M, Schwartzkopf B, Stockinger H, Stockinger K, and Tierney B. Giggle: A Framework for Constructing Sclable Replica Location Services. In *Proceedings of Supercomputing 2002 (SC2002)*, November 2002.

13. Deelman E et al. Pegasus: Mapping Scientific Workflows onto the Grid. In *2nd EU Across Grids Conference*, 2004.

14. Buneman P, Khanna S, and Tan W.-C. Why and Where: A Characterization of Data Provenance. In *International Conference on Database Theory*, 2001.

15. The QuarkNet Project. **http://quarknet.fnal.gov/**, June 2004.

16. Bardeen M, Gilbert E, Jordan T, Nepywoda P, Quigg E, Wilde M, and Zhao Y. The QuarkNet/Grid Collaborative Learning e-Lab. In *Second International Workshop on Collaborative and Learning Applications of Grid Technology and Grid Education*, 2005.

17. Van Horn J, Dobson J, Woodward J, Wilde M, Zhao Y, Voeckler J, and Foster I. Grid-Based Computing and the Future of Neuroscience Computation. In *Methods in Mind*. MIT Press (In Press).

18. Van Horn J. Online Availability of fMRI Results Images. *Journal of Cognitive Neuroscience*, 15(6):769–770, 2003.

19. Woods R. Automated image registration: I. General methods and intrasubject, intramodality validation. *Journal of Computer Assisted Tomography*, 22:139–152, 1998.

20. Woods R. AIR 5 Suite. **http://bishopw.loni.ucla.edu/AIR5**.

21. Moreau L, Zhao Y, Foster I, Voeckler J, and M Wilde. XDTM: XML Dataset Typing and Mapping for Specifying Datasets. In *European Grid Conference*, 2005.

22. Amin K et al. GridAnt: A Client-Controllable Grid Workflow System. In *37th Hawaii International Conference on System Sciences*, 2004.

23. Myers J et al. A Collaborative Informatics Infrastructure for Multi-scale Science. In *Proceedings of the Challenges of Large Applications in Distributed Environments (CLADE) Workshop*, June 2004.

24. Stein G. Web Digital Authoring and Versioning (WebDAV) Resources Community Website. **http://www.webdav.org/**, June 2004.

25. Krishan S et al. The XCAT Science Portal. SC2001, November 2001.

26. Goble G, Pettifer S, and Stevens R. Knowledge Integration: In silico Experiments in Bioinformatics. In Ian Foster and Carl Kesselman, editors, *The Grid: Blueprint for a New Computing Infrastructure*, pages 121–134. Morgan Kaufmann, 2004.