Web Enabled Collaborative Climate Visualization in the Earth System Grid

Wesley Kendall, Markus Glatter, Jian Huang
University of Tennessee, Knoxville
E-mail: {kendall, glatter, huangj}@cs.utk.edu

Forrest Hoffman and David E. Bernholdt
Oak Ridge National Laboratory, Oak Ridge
Email: {hoffmanfm, bernholdtde}@ornl.gov

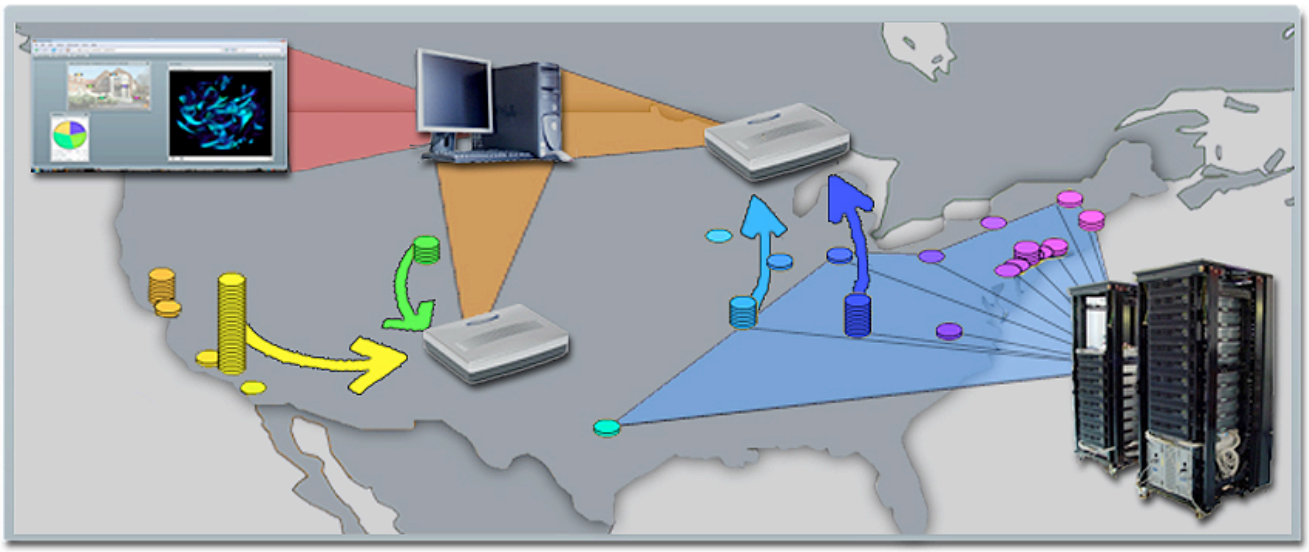

Figure 1. Illustration of our web-enabled data-intensive visualization system.

ABSTRACT

The recent advances in high performance computing, storage and networking technologies have enabled fundamental changes in current climate research. While sharing datasets and results is already common practice in climate modeling, direct sharing of the analysis and visualization process is also becoming feasible. We report our efforts to develop a capability, coupled with the Earth System Grid (ESG), for sharing an entire executable workspace of visualization among collaborators. Evolutionary history of visualizations of research findings can also be captured and shared. The data intensive nature of the visualization system requires using several advanced techniques of visualization and parallel computing. With visualization clients implemented through standard web browsers, however, the ensuing complexity is made transparent to end-users. We demonstrate the efficacy of our system using cutting edge climate datasets.

KEYWORDS: Large Data Visualization, Climate Modeling, Distributed Collaborative Visualization.

1. INTRODUCTION

The recent advances in high performance computing, storage and networking technologies have enabled fundamental changes in modern climate research. An ever-closer integration of observational data with simulations of an unprecedented scale, both in terms of resolution and size of ensembles of runs, is already under way. Further enabled by the recent creation of the Earth System Grid [1,14], as a universal global repository of datasets from different scientific models, climate science, and climate modeling in particular, has become fundamentally collaborative in a widespread manner. Many previously unapproachable problems can now be studied. In a way, today's globally collaborative climate research represents a central trend that is taking place in all areas of computational sciences. Scientists not only share datasets and research results, but more importantly they can now also share their research processes.

In this work, we describe a prototype system that enables sharing large-scale visualization of climate simulations in an executable form. By exchanging a few compact specification files and URLs (for instance, via email), geographically separated scientists can asynchronously execute and view the same visualization workspace for

collaborative scientific insights. The same specification can be applied to other datasets for comparative study.

Scientists can also edit and tweak the specifications in ways that reveal additional scientific findings and efficiently communicate with their colleagues about the modified visualization. An email thread, with the evolving versions of visualization specifications as attachments (or the same content published via web blogs), is a natural way to document and share a collaborative research process that has lasted over a long period of time. This computing interface not only makes it easier to reproduce the visualization results on other datasets, but also in essence provides a possibility to critique and disseminate a practice of science.

The data intensive nature of collaborative climate visualization presents many challenges that computer scientists must resolve. However, the technical complexities must be made transparent to end users as much as possible. In our system, we have implemented our user interfaces universally through web browsers. Users are not required to install any additional software beyond those that are standard with all web browsers.

Since in essence our collaborative visualization system and the ESG are both designed to support the global community of climate research, let us try to distinguish the role that each system plays from a high level point of view. To date, the ESG's primary function and benefit has been to make data available to a much larger and more general community than had previously been possible. A key goal of the next generation ESG architecture is to serve as a point of integration coupling data access with routine analyses and visualizations, helping to reduce download volumes and user-side storage and processing requirements.

The ESG team is working to integrate with some "standard" climate analysis and visualization modules, such as the Live Access Server [10], but the development of novel visualization techniques and tools is beyond the scope of the ESG project, rather, they are expected to be developed through external collaboration. Our work is one early example of this concept in action. Collaborative server-side visualization is novel for the ESG community, where most of the ESG server-side processing use cases focus on the scientific needs of individual users. With the idea of sharing analysis and visualization results in a model similar to social networking just starting to gain attention in the field [13], no existing efforts can provide such a level of collaboration for data intensive applications. These large data applications are the norm for our system and the ESG, both in terms of the amount of data that must be handled and the complexity that is involved in the scientific process of research.

The rest of the paper is organized as follows. Section 2 contains a brief discussion of the related background of our work. The technical components for handling data intensive visualization are provided in Section 3. Our web browser based visualization workspace is described in Section 4. In Section 5, we present a sample workflow using our system.

2. Background

Our driving application is to visualize results from global coupled climate-carbon cycle model simulations produced by different models. We draw our research motivation from the current thrust to generate accurate simulations of the global carbon cycle that model the interactions and feedbacks between the terrestrial biosphere and the climate system. The Carbon-Land Model Intercomparison Project (C-LAMP, www.climatemodeling.org/c-lamp [5] was initiated to allow the international scientific community to thoroughly test and intercompare such terrestrial biogeochemistry models through a set of carefully crafted experiments. Well-defined metrics have been established for comparison of model results against best-available observational datasets, and models are graded on their scientific performance with respect to these metrics. Visualization tools and diagnostics are particularly helpful in uncovering model biases and discovering ways for improving individual models.

The visualization aspect of this task is very demanding for several reasons. First, there are a large number of variables involved in each simulation. Exacerbated by the need to study multiple runs in a cohesive manner, the combinatorial space that needs to be explored, even just the task of studying two variables from two simulation runs, is overwhelming. For instance, scientists already have some understanding of how net ecosystem exchange relates to net primary productivity. Does the relationship exist as expected in a peta-scale simulation, maybe across different time spans of different runs? This model of investigative study and the need of high interactive rates present a challenge for large data visualization.

Second, a very desirable feature in data intensive visualization tools in this field is for the tools to be available on-demand. In climate research, exhaustively iterating over all possible combinations is prohibitively expensive. One can only use the domain expertise as a guide to practically sample a small portion of the entire combinatorial space. This process of scientific research often involves an element of spontaneity. Hence it is preferable not to require advanced reservation before each use. We do not know of any previous systems that can support such spontaneous use of visualizations by geographically separated concurrent users. Thankfully, powered by the ESG, it is now feasible to investigate how

on-demand visualization capabilities can be developed for large-scale climate modeling research and tested in a real-world environment. A brief overview of the ESG follows.

2.1. Earth System Grid

The Earth System Grid (ESG, www.earthsystemgrid.org) [1, 14] has been developed as a tool to make the large volumes of climate simulation data more easily discoverable and accessible to a larger community of researchers interested not only in the simulations themselves, but also in understanding the impacts of climate change, and possible technological and policy strategies for their mitigation. The ESG collaboration includes seven US research laboratories (ANL, LANL, LBNL, LLNL, NCAR, NOAA/PMEL, and ORNL) and a university (USC/ISI), and is sponsored by the US Department of Energy, Office of Science.

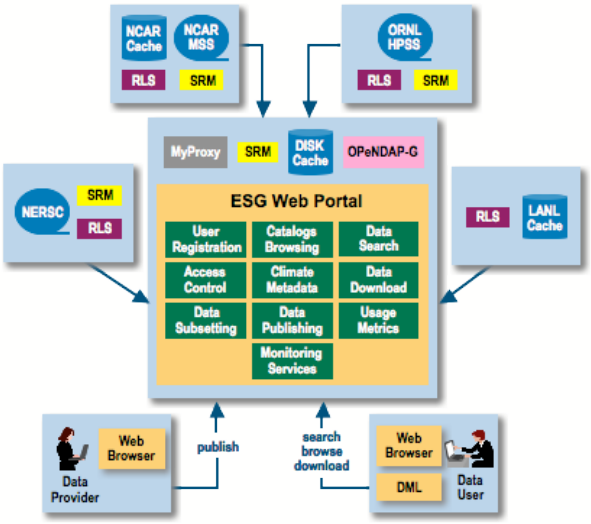

Figure 2. Portal Architecture of the ESG.

When climate simulation results are published into the ESG, metadata is extracted providing a detailed description of the simulation context and the variables represented in the datasets. ESG users can browse or query the metadata catalog through a web portal to discover datasets of interest, and then request their retrieval from the archive site that holds them, most often at the computer center where the simulation was originally run (see Figure 2).

Among the key goals of the next generation of the ESG architecture, currently under development, are much more widely distributed deployment, and integration of analysis and visualization capabilities on the server side in order to minimize the amount of data that has to be delivered to users over (relatively slow) wide-area networks, and to reduce the local computing and storage requirements for many users. The work presented in this paper can be seen as an example of the server-side processing the next-generation ESG plans to support. Because of community interest in the next generation of biogeochemistry models, our work has focused on visualization of the data held in the C-LAMP [5] archive, which is physically located at ORNL, but the technique described is broadly applicable.

2.2. Collaborative Parallel Visualization using Heterogeneous Resources

Researchers in distributed graphics pioneered the creation of distributed collaboration capabilities involving 3D graphics. The resulting technology is already in widespread use by distributed systems of virtual reality, networked games and visualization [2, 9]. Unfortunately, when it comes to data intensive applications, involving terabyes (TB) to petabytes (PB) datasets by today's standards, we still do not have a widely accepted solution that could scale up both in terms of the amount of data that can be handled and the number of concurrent users that can be supported.

Large-scale data intensive applications dictate the use of parallelism. However, computing platforms that are capable enough to handle large datasets are still far from being a commodity. Even with a large-scale parallel machine available, it still takes time to develop capabilities based on non-primitive use of parallelism. This problem is particularly acute when it is impractical to provision a homogeneous large-scale visualization facility for dedicated use by a collaborative project. With today's project needs evolving faster than before, the issue of not having the right kind of parallel platform promptly available is a rather fundamentally limiting issue.

In this regard, it is very appealing to alternatively consider using distributed heterogeneous resources as the common platform to support on-demand parallel visualization. This would allow one to efficiently use all resources available, regardless of their geographical location. On top of this platform, visualization can be implemented according to the software-as-a-service model and allow fast evolution and adaptation. In a way, this approach has been well studied by researchers in networked games and virtual environments [12] and grid computing [3]. In those applications, each user's own personal computer is an individual piece of heterogeneous resource. However, to assemble data intensive visualization services with fault-tolerance, high scalability and high performance, we still have significant road blocks to overcome, due to the data-intensive aspect and investigative mode of usage that restricts orchestrating user interaction according to planned scripts.

As we have discovered through extensive previous research in large-scale distributed visualization, distributed collaborative visualization systems that do not include fault-tolerance and heterogeneity as primary design goals would be very hard to use, even just to carry out experiments mimicking real-world scenarios for multiple concurrent users [6]. This is because, in a real scenario, users join or leave a collaborative session on different schedules. The client software used by geographically separated users could frequently be turned on and off. To assume that a resource, regardless of type (e.g. graphics, computation, storage, and a guaranteed level of network bandwidth), would always be available, even with prior reservation, could quickly render a system impractical. We have discovered through previous research that the primary factor is that the parallel scheduling algorithm must be designed to consider heterogeneity [7].

Although collaborative visualization has been a traditional topic of research, its focus has been mostly about a synchronous mode of collaboration. In other words, all participants are assumed to come online at the same time and work on the same subject. With the advent of Web 2.0 technology, especially represented by the introduction of ManyEyes [13], focus in many fields has now been shifted to study how geographically separated users can asynchronously collaborate by means such as community-shared annotations and blogs. It would be particularly interesting and impactful if such asynchronous mode of collaboration can be inherently supported by data intensive visualization applications.

In summary, while distributed collaboration is a concept that has been pioneered, reinvented in the past and is already in widespread use by millions of current Internet users, the main technical hurdle for users to effectively collaborate on tera- to peta-scale datasets is still a fundamental one. It is not a matter of simply addressing a few tradeoffs. We must develop creative technologies as well as creative ways to use current technologies to meet this challenge. As an attempt, our system is described in Section 3 using climate visualization as an example.

3. PARALLEL VISUALIZATION

3.1. Overview

Our prototype system inherently utilizes heterogeneous computing resources. All nodes involved can be distributed by nature; in fact, each node is uniquely identified by its IP address. Any regular personal computer can serve as a node used in a session of parallel computing. Figure 1 illustrates the concept. The high performance computers (lower-right corner) represent the typical hardware maintained by ESG. Climate-carbon cycle model results are queried, subsetted and retrieved from high-performance systems maintained by the ESG. The data are transformed, partitioned and replicated on standard Internet computing nodes for fault-tolerance and later use in on-demand parallel visualization. In a way, the nodes in our visualization system can be considered as a set of function-rich data caches. A set of visualization servers manages the parallel visualization process (illustrated as blade servers). The colored arrows illustrate parallel visualization results computed on heterogeneous nodes are being assembled by the schedulers, which then forward the results to the web browser running on the client workstation. In the client's visualization workspace, the final visualizations are displayed for viewing, annotation and interaction.

A set of concurrent users can switch on the visualization clients through their web browser to work in their individual visualization workspaces. The visualization operations initiated by each workspace are computed in parallel and orchestrated by a set of fault-tolerant and heterogeneity-aware visualization schedulers. The most generic visualization operation in our system centers around the notion of query-based visualization. After discussing some details of how to implement this technique in a scalable manner, we follow with a discussion of how to integrate parallel query-based visualization using heterogeneous computing resources.

3.2. Parallel Query-Based Visualization

Decadal to century time scale climate simulations typically output a large number of two- and three-dimensional variables at regular intervals, usually monthly. While identifying features is difficult, it is actually intuitive and practical to select a subset of the data from within that high-dimensional value/variable space to obtain a qualitative understanding of the overall results [11]. This approach is not hard to implement if the resulting dataset is small enough to be stored entirely in-core, but to handle datasets sized at hundreds of gigabytes and beyond, more sophisticated solutions are required.

Our solution [4] involves designing specialized scalable visualization data servers with large-scale parallelism. It indexes general data items, including vertex, voxel, particle, or a general tuple describing height, weight and gender. Due to this design consideration, it is independent of grid type. The core data structure is a heavily optimized M-ary search tree to support parallel visualization. The tree structure as metadata only amounts to ~1% the size of the whole dataset, which can exist entirely on external hard drives and can be compressed. Our method only decompresses the parts of the data (and caches the decompression results) that are used by the

user. The compression rates vary from dataset to dataset. For typical datasets, we saw 20x compression rates, while on highly turbulent or noisy datasets, we could obtain as low as 4x compression. Using a conservative compression rate of 4 times as a benchmark, a typical mid-sized cluster could already support parallel visualization of a dynamically queried dataset sized at 1 TeraByte (TB).

The M-ary tree serves the role of meta data to guide a search process. Assuming the balanced M-ary search tree is of depth N, a practical combination could be M = 256 and N = 3; in all situations, M should always be significantly larger than N. This is one of the primary differences between this search data structure and previous data structures, such as interval tree, k-d tree, quadtree and octree. Due to the large branching factor, M, the M-ary search tree requires little storage space. Actual data items are not stored in the tree, but in a linear list sorted by a key function. The leaf nodes of the tree store pointers to the respective records in a sorted linear list.

M-ary search trees have been widely used, for instance in the form of B-trees as in database systems. We have discovered that conventional methods to access records by traversing B-trees to be too expensive, both in terms of the large number of addressing operations and caching performance. To address this, we use a novel method to accelerate range searches in an M-ary tree, optimized specifically for multivariate datasets [4].

Data items are partitioned into groups by round-robin assignment according to high-dimensional space filling curve [8] order in attribute space. By using this type of data partition to distribute data amongst all visualization data servers, we can achieve a nearly optimal load-balance for almost all kinds of queries. An M-ary search tree is then used to manage the data on each server.

For a user of our system, e.g., an application scientist, it is often of interest to interact with the dataset of a simulation to verify correctness or to explore newly formulated hypotheses. To this end, one should be able to navigate not only in time and space, but also in the high-dimensional attribute space. Additionally, every server in our system accepts queries in the same way that web servers function. This allows parallel servers to operate persistently, while the client can choose to dynamically start, stop or simply shutdown.

Our approach is relatively easy to deploy on networked commodity computers, whether they are clustered or not. The necessary number of parallel data servers depends on the size of the dataset. As long as more computers are available to function as data servers, our system could be scaled to handle larger and larger datasets. For performance reasons, it is desirable that the combined memory of all data servers be sufficient to hold the entire preprocessed data in core. However our implementation remains operable even if this requirement is not met.

3.3. Fault-Tolerant Job Scheduling

Each data server operates as an independent parallel process. Server processes will be replicated on multiple nodes for scalable performance as well as fault-tolerance. Considering each request to any one of the servers as a task to be completed in parallel, the scheduler of the parallel tasks is crucial for our system.

The scheduler we use was designed to achieve scalability while considering heterogeneity and fault-tolerance [7]. Our scheduler implements co-scheduling of computation and replication, by adaptively measuring node performance to dynamically assign visualization tasks and direct runtime data movements. This is necessary because the nodes used for parallel computation will be heterogeneous, non-dedicated nodes. Geographically distributed users access the system with competing goals of obtaining computing resources.

Our scheduler needs to discover fast processors on the fly, assign as many tasks to them as possible and avoid being stalled by slow or faulty nodes. We devised three generic mechanisms for this need: (i) a dynamically ranked pool of nodes, (ii) a two-level priority queue of tasks and (iii) a competition avoidant task assignment scheme. Each processor is ranked by its estimated time to process a task of unit size. This measurement roughly reflects performance of processors delivered to the experiment. It is updated adaptively by computing a running average of measured processor performance, with older measurements given an exponentially decreasing weight.

A two-level priority queue maintains unfinished tasks. The higher priority queue (HPQ) contains tasks that are ready to be assigned and the lower priority queue (LPQ) contains tasks that have been assigned to one or more processors but not finished. Initially, only the first w tasks are placed in the HPQ, where w is the size of task window. It controls the degree of parallelism and number of tasks that can be finished out-of-order. In most cases, w is greater than the number of available processors so that every processor can contribute. However, w is decreased in cases of severe resource contention as a processor "back off" strategy. Each task in the HPQ is keyed by the minimum unit task processing time of all nodes holding the required data partition. This priority ranks new tasks by their likelihood of being finished by a fast node. Each task in the LPQ is keyed by its estimated waiting time. Tasks in both the HPQ and LPQ are sorted by their keys in decreasing order. Figure 3 shows an illustrative

example of how the scheduler operates in general, and a snapshot of the window-based task assignment.

When a parallel visualization starts, tasks in the HPQ are sequentially assigned to available processors and demoted to the LPQ. In case of failure, the task in the LPQ is promoted back to the HPQ so that other processors can take it over. When tasks are completed, every available node will be directly assigned the first task in the HPQ that it can handle. In this way, slow processors do not compete for tasks with fast processors to ensure fast processors are assigned as many tasks as possible. If a processor is not assigned a task in the HPQ, the first task in the LPQ that it can handle is considered. This is the slowest task among all unfinished tasks that the processor can assume. Thus, multiple processors can work in parallel on the same unfinished task. If any instance of the duplicated tasks is done, other instances are aborted.

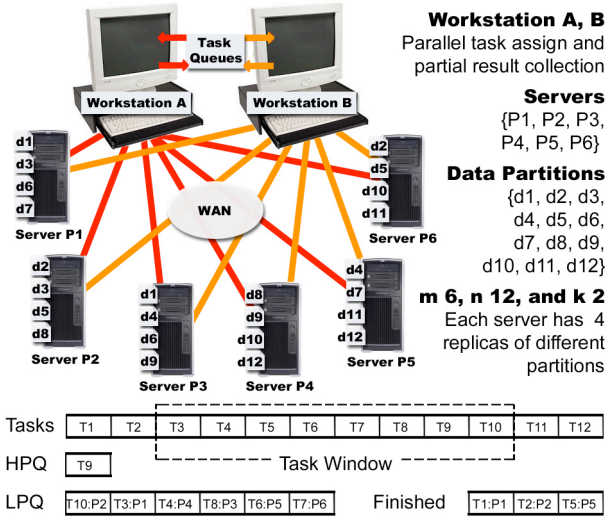

Figure 3. Our Heterogeneous Scheduler.

There are also situations where moving data from a slower processor to a vacant faster processor may be beneficial. To ensure that the overhead of data movement does not exceed the benefit that we might gain, bandwidth information between processors needs to be acquired. We then leverage a multi-source partial download scheme with deadline to drive data movement between nodes.

3.4. System Integration

Currently data replication is our only method of fault-tolerance for addressing failing processors as well as overloaded processors. Each data server node holds a portion of the entire dataset to be visualized. Through experimentation, we have found a replication factor of k = 3 to be sufficient for most situations and not cause too much operating overhead. In this case, each piece of data is held by 3 server nodes, and each server node holds 3 pieces of data. In other words, assuming a 1 TB dataset has been partitioned for hosting by 100 data server, each server holds 30 GBs of the data (before compression).

Any visualization query is parsed and issued first to only one server holding data that are needed. Secondary requests are only issued when a result is not returned by a server before a timeout period has expired. The processes for different visualization queries take place in parallel. It is typical for each user's visualization workspace to include a handful of different visualizations, while several users can use the system concurrently.

4. WEB-ENABLED INTERFACE

All the visualization results are viewable through a standard web browser, together with an information visualization viewer that monitors the parallel run. The web interface was designed using Flex Builder 2 and Action Script 3. To create a generalized framework for any web visualization, the web interface is set up by an XML configuration file. This configuration file specifies what controls are to be shown on the interface along with the properties of the controls and the visualization server (hostname and port). Each control is given a separate smaller window inside the web console, giving users their own desktop for organizing the visualization.

Once the controls for the visualization are set up on the web interface, it then connects to the visualization server with the Action Script 3 Socket library. Using our own visualization server API, the visualization server can relay messages to the web interface using notification messages. This message transmission procedure is set up in the configuration file. The user has complete control of which components of the web interface the messages are sent to.

To allow the components of the web interface to interact with the visualization server, the visualization server must register callback functions for any component of the interface that might interact with it. For example, if a query comes from a video viewer on the web interface, the user would register a specific callback function, and any queries from the video viewer with that ID would be sent to the function in a special generalized format that is easily parsed by another function in the API. This API and the configurable web interface combined allow any user to create their own web visualizations without any knowledge of web or socket programming. It also allows the user to plug in any back-end system, regardless of a cluster or a home desktop, to their web visualization.

4.1 Data Selection Interface with ESG

In the absence of more flexible interfaces to the metadata and data, which are currently under development by the ESG team, we have implemented our prototype system as a post-processing client to the ESG. A web-based client (shown in Figure 4, top-left) utilizes the the FTP-based access provided to the C-LAMP archive to allow the user to browse and select the data to be downloaded to our system. When submitting their request, users are asked to provide their name, affiliation, and email address. When the download is complete, they receive an email with their session ID, password, and a link to process the data. The password provides a simple mechanism to restrict control of the processing of the data to the requesting user, but anyone will be able to analyze the data in a given session. In the next-generation ESG architecture, we anticipate that it will be possible to provide the user with a seamless interface, including integration with the ESG's authentication and authorization model to allow more control over access to the analysis results if desired.

4.2 Data Processing & Linking with Data Servers

After the files have been transferred, data processing starts for visualization purpose. The user goes to a web address, enters their session ID and password, and then browses/selects the variables in the NetCDF files. As illustrated in Figure 4 (top-right), the variables are added to a cart similar to how the previously mentioned FTP client worked. Once the users select all the variables that are of interest to them, the data is then sent to the data servers and processed. A configuration file for the web interface is automatically generated, and a link to the web visualization is sent to the user. The user may then query the variables chosen in the data processing stage. Any person may go to this link and analyze the data.

4.3 Collaboration through the Web Interface

Each web visualization has a specific session ID that is assigned to it when created. Users can type in the URL to the web interface along with the session ID as a parameter in the URL and this will bring up the visualization. Along with each interface is a control that allows users to leave comments along with the previous comments that have been left. Since the interface is connected to a background server via sockets, this allows the comments to be updated in real time. If multiple users are running the same session, they will all see any new comments posted without having to refresh the browser, simulating a chat room. Another component of each web visualization is a button to capture the state of the visualization. When pressed, a URL will be created that automatically generates the same visualization of the captured one. Since these captures are still associated with the individual sessions, multiple scientists could be in a web visualization session at once sending each other these URLs to simulate a live, real-time collaboration of the same visualization.

5. TYPICAL USE CASES & DISCUSSIONS

We use the screen capture shown in Figure 4 (bottom) to demonstrate a typical use of our visualization system. Our current user interfaces illustrated in Figure 4: (top-left) for selecting datasets from ESG; (top-right) for specifying which subsets of the selected datasets are to be further processed for visualization and analysis; and (bottom) is a screen capture of the web-enabled visualization interface in operation. This example shows that the user selected C-LAMP runs from the ESG and chose variables such as CO_2 concentrations, net ecosystem exchange (NEE), net primary productivity (NPP), and vegetation temperature (TV). The time span of interest is from 1800 to 2005. A user could either examine how each variable evolves over time in a 2D movie, or do the same with the (first order) derivative of each variable.

In addition, we allow the user to study pairs of variables. It may be interesting, for instance, to visualize the first derivative of NEE vs. the first derivative of NPP. If both derivatives are of the same sign, we assign the colors blue (both positive) and red (both negative), while green and yellow are used for cases where the two derivatives are of opposite signs. This is useful for revealing patterns in which the two variables, NEE and NPP, vary in tandem of each other. The two variables could be given a different time lag as well. For instance, have the time sequences of NEE start from January 1950 and NPP start from January 1980. Similarly, we can also study the monthly pattern variations between NEE from 1900-1920 vs. NEE from 1980-2000.

A user can create an unlimited number of analyses. Each analysis is given a separate floating viewport. All viewports can be moved around in the workspace in the web browser. This design allows the user to selectively position viewports in neighboring positions for closer comparisons. For an intuitive control by our domain users, each viewport essentially acts as a movie viewer, with its own set of controls including play, pause, and time tick. In addition, at the bottom of the browser, there are overriding play and pause buttons for the whole workspace. This design allows each viewport to be set to a different starting point and to be started and paused in a synchronized manner. At the lower-right corner, there is a universal frames per second control. For fast browsing, we have found 10 frames per second or more is necessary. For detailed study, the frame rates can be manually changed to low single digit values such as one or two.

In total there are several orthogonal dimensions of controls: which variables, the starting time tick of each variable and the starting point within each viewport. The overall configuration of the whole workspace can be captured and saved. The configuration is identifiable as a URL and can also be retrieved using a session ID. Password authentication is used to restrict editing privilege to only those who have the passwords. General users can have viewing access without editing privileges.

Through our work, we realized that the impact of making previously hard to find datasets available as a common commodity for scientific research, like how climate research is undergoing influences by the ESG, cannot be over estimated. Visualization and analytical tools that operate on ESG-caliber scientific datasets must embrace this future model of sharing data. For this purpose, it is crucial for the visualization community to proactively explore ways to develop new technology as well as find creative ways to use existing technology.

ACKNOWLEDGMENTS

The authors gratefully acknowledge the primary source of funding for this work is provided through Institute of Ultra-Scale Visualization (www.ultravis.org), under the auspices of the DOE Scientific Discovery through Advanced Computing (SciDAC) program. This work is also supported in part by NSF grant CNS-0437508 and a DOE Early Career PI grant awarded to Jian Huang (grant no. DOE DE-FG02-04ER25610). We are in debt for our crucial collaboration with the Logistical Computing Laboratory (LoCI) co-directed by Dr. Micah Beck and Dr. Terry Moore at the University of Tennessee. LoCI lab researcher, Huadong Liu, is the primary designer of the heterogeneous scheduler [7] used in our system. We would also like to thank Dr. David Erickson a nd Dr. John Drake (ORNL) for the many in-depth discussions that helped us to conceive our system, and Jamison Daniel (ORNL) for his expertise on climate datasets.

The Earth System Grid (ESG) and the Carbon-Land Model Intercomparison Project (C-LAMP) are partially sponsored by the Climate Change Research Division (CCRD) of the Office of Biological and Environmental Research (OBER) and the Computational Science Research and Partnerships (SciDAC) Division of the Office of Advanced Scientific Computing Research (OASCR) within the U.S. Department of Energy's Office of Science (SC). This research used resources of the National Center for Computational Science (NCCS) at Oak Ridge National Laboratory (ORNL), which is managed by UT-Battelle, LLC, for the U.S. Department of Energy under Contract No. DE-AC05-00OR22725.

REFERENCES

[1] Ananthakrishnan, R., D. E. Bernholdt, S. Bharathi, D. Brown, M. Chen, A. Chervenak, L. Cinquini, R. Drach, I. Foster, P. Fox, D. Fraser, K. Halliday, S. Hankin, P. Jones, C. Kesselman, D. Middleton, J. Schwidder, R. Schweitzer, R. Schuler, A. Shoshani, F. Siebenlist, A. Sim, W. Strand, N. Wilhelmi, M. Su, D. Williams. “Building a global federation system for climate change research: The earth system grid center for enabling technologies”. SciDAC 2007, June 2007, Boston. *Journal of Physics: Conference Series*, Vol. 78, page 012050. Institute of Physics, 2007.

[2] Bethel, W. “Visualization dot com”. *IEEE Computer Graphics and Applications*, 20(3): 17–20, 2000.

[3] Brodlie, K., D. Duce, J. Gallop, M. Sagar, J. Walton, and J. Wood. “Visualization in grid computing environments”. Proc. of IEEE Visualization Conference, Austin, TX, 2004.

[4] Glatter, M., C. Mollenhour, J. Huang, and J. Gao. “Scalable data servers for large multivariate volume visualization”. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):1291–1299, 2006.

[5] Hoffman, F., C. Covey, I. Fung, J. Randerson, P. Thornton, Y-H Lee, N. Rosenbloom, R. Stockli, S. Running, D. E. Bernholdt, and D. Williams. “Results from the Carbon-Land Model Intercomparison Project (C-LAMP) and availability of the data on the Earth System Grid (ESG)”. SciDAC 2007, June 2007, Boston. *Journal of Physics: Conference Series*, Vol. 78, page 012026. Institute of Physics, 2007.

[6] Huang, J., H. Liu, J. Gao, A. Gaston, M. Beck, and T. Moore. “Visualization viewpoints: Dynamic sharing of large-scale visualization.” *IEEE Computer Graphics and Applications*, 27(1): 20-25, 2007.

[7] Liu, H., M. Beck, and J. Huang. “Dynamic co-scheduling of distributed computation and replication”. Proc. of IEEE/ACM CCGrid: Intl Symp. on Cluster Computing and the Grid, Singapore, 2006.

[8] Pascucci, V., and R. Frank. “Global static indexing for real-time exploration of very large regular grids”. In ACM/IEEE Conference on Supercomputing (SC’01), page 2, 2001.

[9] Singhal, S., and M. Zyda. “Networked Virtual Environments: Design & Implementation”. Addison-Wesley Professional, 1999.

[10] Sirott, J., J. Callahan, and S. Hankin. “Inside the live access server”. In 17th Intl Conference on Interactive Information and Processing Systems for Meteorology, Oceanography, and Hydrology, AMS, 2001.

[11] Stockinger, K., J. Shalf, K. Wu, and W. Bethel. “Query-driven visualization of large data sets”. In IEEE Visualization’05, page 22, 2005.

[12] Trefftz, H., I. Marsic, and M. Zyda. “Handling heterogeneity in networked virtual environments”. In Proc. of IEEE VR, pages 7–15, Orlando, FL, 2002.

[13] Viegas, F., M. Wattenberg, F. van Ham, J. Kriss, and M. McKeon. "Manyeyes: a site for visualization at internet scale". *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1121–1128, 2007.

[14] Williams, D., D. E. Bernholdt, I. Foster, and D. Middleton. "The earth system grid center for enabling technologies: Enabling community access to petascale climate datasets". *CTWatch Quarterly*, 3(4), November 2007.

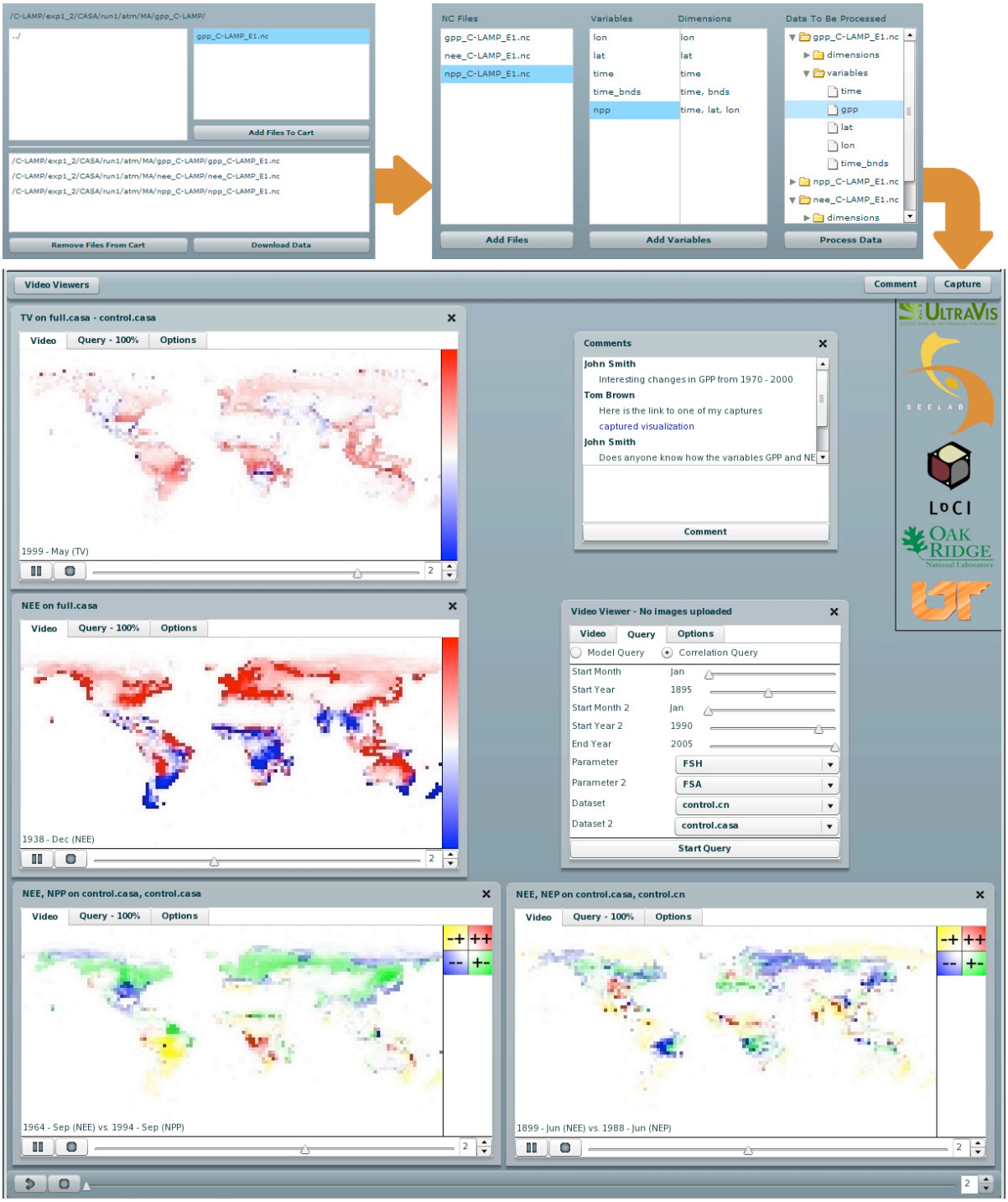

Figure 4. Our Current User Interfaces and a Runtime Screen Capture of Our System.