Collaborative Visualization Spaces for Petascale Simulations

Scott Klasky
ORNL
klasky@ornl.gov

Roselyne Barreto
ORNL
rbarreto@ornl.gov

Ayla Kahn
Univ. of Utah
ayla@sci.utah.edu

Manish Parashar
Rutgers
parashar@rutgers.edu

Norbert Podhorszki
ORNL
pnorbert@ornl.gov

Steve Parker
Univ. of Utah
sparker@cs.utah.edu

Deborah Silver
Rutgers
silver@caip.rutgers.edu

Mladen A. Vouk
NC State University
vouk@ncsu.edu

ABSTRACT

Petascale computing is approaching quickly, but the task of building simulations on these machines is daunting. These computers contain hundreds of thousands of processors, and the simulations can run for weeks to produce vital results, which then must be analyzed and visualized. As the complexity of the simulations and computers increases, so does the complexity of the information that they generate. This data must be carefully analyzed by teams of scientist to achieve the desired scientific insights. In this paper, we focus on one aspect of the Fusion Simulation Project (FSP): collaboration amongst OASCR and OFES. The principal characteristic of the targeted simulations is the coupling of several fusion codes. The principal characteristic of our analytics is that it is highly collaborative in nature. The key to enabling the latter is a backend that allows scientist to perform powerful analyses and visualizations using a web-portal (i.e., our "eSimMon"). Issues such as the orchestration of the flows among simulation, management of storage and analytics resources, movement of the data, and collection of the meta-data are addressed using workflow technologies.

KEYWORDS: visualization, collaboration, workflows, dashboard, provenance, petascale, automation, analytics.

1. INTRODUCTION

Leadership class computing has enabled a new age of computing where simulations can run for weeks on hundreds of thousands of processors. These simulations typically address multi-physics, and the complexity of coupling the different areas of physics is staggering. A key requirement for addressing this challenge is effective visualization and analytics support that can handle the scale and complexity of these simulations. Our motivation for building the collaborative visualization space described in this paper is the ITER [1] project. ITER is an international research and development project that aims to demonstrate the scientific and technical feasibility of fusion power. In order to make this project successful, the United States is attempting to launch a Fusion Simulation Project (FSP) which aims to leverage leadership class computing to develop a predictive capability for integrated modeling of magnetically confined burning plasmas. The knowledge that we gain from the FSP will ultimately enhance our understanding of data from burning plasma discharges and provide new opportunities for scientific discovery.

The FSP will need to predict the behavior of plasma discharges which span a huge order of time and spatial scales. This means that the project will ultimately require the coupling of a large number of complex codes, developed independently, to model this behavior. This means that we must need a comprehensive computational framework and tools based on a collaboration among different areas in physics, computer science, and applied mathematics. At first, these simulations must be verified, which means that we must make sure that the coupled simulations are doing the right thing, and can be tested against individual codes. This verification involves collaboration amongst many physicists, including students and mentors. Once the data is verified, it will then need to be validated against current fusion experiments, and then ITER when data from that project becomes available. Once the simulations are validated, we hope to have a predictive capability. This capability requires access to petascale-class and exascale-class computer facilities in order to span the relevant time and spatial scales.

The FSP will be one of the largest collaborations ever attempted by the magnetic fusion program. This means that it will be necessary for the program to create collaborative infrastructure which will allow the scientists to work together to solve the complex problems. This is a daunting task. One of its most

978-1-4244-2249-4/08/$25.00 ©2008 IEEE

important aspects is creating a collaborative space that enables monitoring and documentation of each simulation, and sharing of this information with collaborators. Given that this is a research program, we understand that the data will be fluid, changes in codes will be common, and researchers need to understand their necessary pieces and their roles in the project.

Central to what we are creating is a "**data store**" - a database, which stores all of the information, and collects all the provenance information about the simulations and the entire lineage of the data. This allow researchers to look at their own piece (or other pieces they are given permissions to view), and be able to understand which codes they ran in a coupled simulation, what the performance of the code was, and what the data produced from the other pieces of code they coupled to was. If a researcher sees a problem he/she can send messages to the other collaborators, and as that is done the relevant information (codes used, monitors used, etc.) would automatically be shared. Because we need to automate all of these tasks, we believe that it is important to do this within a strong workflow framework.

Another key requirement for the collaboration space is that it needs to be **easy to use** regardless of the end-user information technology expertise - from the theorists who commonly use their pencil and paper, to the experimentalists, applied mathematicians, computational scientists, and computer scientists. This requirement has been one of the central design themes of the infrastructure presented in this paper. We built our dashboard system, called eSimMon, using enterprise web tools; i.e. Web 2.0 technologies. By placing the burden of the system on the backend services, we reduced the complexity of software that one must install. One of our key goals is to create a unifying software infrastructure to make it easy for scientists to collaborate and use the data analysis tools that are common in their workplace. For instance, in the workflow we describe below, we use several visualization and analysis tools, but hide their complexity from the end users.

The rest of this paper is organized as follows. In the next section we introduce the fusion simulation project driving this research. In Section III we discuss workflow and provenance issues. Section IV describes the users' interface to the proposed infrastructure – our "dashboard". Section V discusses the current status of the project, as well as future work and concludes the paper.

2. FUSION SIMULATION PROJECT

Our initial design leverages experiences of two successful projects: the Center for Plasma Edge Simulations (CPES) [2], and the Scientific Data Management center (SDM) [3]. Both SciDAC (Scientific Discovery through Advanced Computing) projects funded jointly by the U.S. Department of Energy (DOE) Office of Fusion Energy Sciences and Office of Advanced Scientific Computing Research. The goal of the CPES project is to develop an integrated predictive plasma edge simulation package for existing magnetic fusion facilities and next-generation burning plasma experiments. The goal of SDM is to develop information technology tools that facilitate management of scientific data, and through that discovery of knowledge.

Our case study centers on the simulation of hot plasma in a tokamak fusion reactor. If the hot edge of the plasma is allowed to come in contact with the reactor wall in an uncontrolled way, it can sputter the wall material into the plasma. This in turn may degrade or extinguish the fusion burn, and may shorten the wall lifetime to an unacceptable level. Anomalous tokamak plasma transport is thought to be associated with small-scale plasma turbulence. When the heating power to the core plasma is above a certain threshold level, it has been observed experimentally that a thin plasma layer forms in which the plasma is almost free of turbulence. The central plasma temperature and density rise, and the fusion probability increases dramatically. However, at the same time this layer triggers large magneto-hydrodynamic instabilities, which destroy this layer, by lowering the fusion power in the core, and dump the plasma energy to the material walls. At the present time, it is not well understood how this layer builds up and how crash that follows occurs. The success of next-generation burning plasma experiments is heavily dependent upon achieving the former without the latter. The understanding of these processes is therefore is of the highest priority in fusion plasma research [4].

In a long-term effort, that requires intense collaboration between physicists, applied mathematicians, and computer scientists working on high-performance computing platforms, CPES is developing a complete kinetic simulation code, ***XGC1***, to address the problem described above [5]. XGC1 is a constantly changing code; some of its features are now in the rigorous validation phase, while others will be developed in the future. Therefore, it is very important that the tools described in this paper help both the code developers, as well as the applied mathematicians and physicists who evaluate each run of XGC1.

In order to model the edge of the plasma, we ultimately must couple the XGC1 code with the XGC0 code. The XGC0 code does not model turbulence, and thus, it can run for much longer, in physical time, with much less

computational resources than the XGC1 code. However, the XGC0 code must constantly be monitored to see if calculations are stable, in the magneto hydrodynamic (MHD) sense. If calculations become unstable, XGC0 needs to be coupled with a two-fluid MHD code (either the M3D [6] or the Nimrod [7] code). The complexity of coupling these codes is considerable as each code is changing and each code has different computational requirements. Petascale computing architectures further complicate the problem. .

In our CPES case-study data management has also become a crucial problem. The codes are generating tremendous amounts of data, Terabytes now, and Petabytes in the near future. As a result, data management tasks need to be automated. The infrastructure and tools described in the rest of this paper address these specific issues in the context of the CPES case-study. The Kepler [8][9] system that is used to automate this CPES workflow and address the issues of provenance..

3. WORKFLOW AND PROVENANCE

In this section we first describe the workflow that is at the heart of the simulation and discovery process we work with, and then we discuss the provenance (or meta-data) collection process we use.

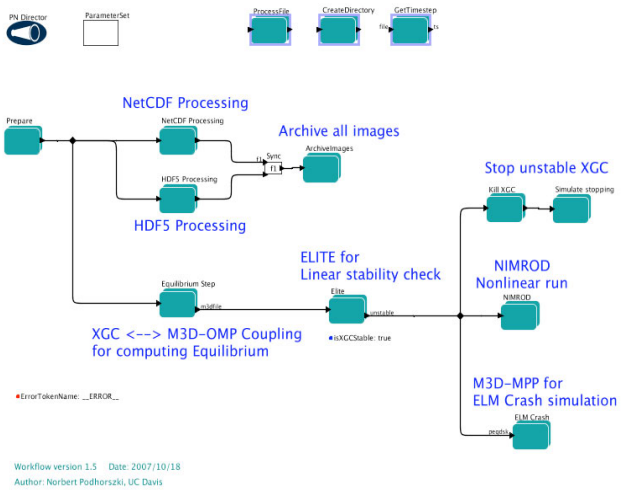

Figure 1. Workflow for the Coupled Fusion Codes

The top-level of the Kepler implementation of the fusion code coupling workflow of is shown in Figure 1. It enables the execution of a multi-physics simulation of edge pressure pedestal buildup, and the subsequent edge localized mode (ELM) instability and crash, in a typical DIII-D [1] H-mode discharge. It loosely couples the gyrokinetic code XGC [10] with an ideal MHD linear stability analysis code ELITE [11] and a two-fluid MHD initial value code such as M3D or NIMROD. There are three pipelines branches: one for monitoring and archiving the NetCDF data, one for generation of HDF5 data, and one for executing the M3D-OMP, ELITE, NIMROD and M3D-MPP steps. The arrows indicate the flow of data/control between those steps.

XGC regularly outputs its state as it runs on the simulation host. This information is then transferred to the post-processing host. XGC is coupled with the M3D-OMP code, possibly running on a separate computer, to periodically compute new equilibrium and to feed that information back to XGC. After that a linear stability test is performed on the simulation data outputs using ELITE. This step is a parameter-study in itself. Different toroidal modes have to be tested to accurately determine when the XGC simulation becomes unstable. The workflow stops the XGC run when ELITE finds that the flow has become unstable. The workflow then initiates an ELM crash simulation using M3D-MPP. The coupling workflow also includes a monitoring and archiving sub-workflows for XGC [12]. The M3D-MPP step is a complete job preparation, submission, monitoring and archiving workflow. In addition, images (plots) are generated from the ELITE step using IDL and gnuplot.

Two pipelines realize monitoring of XGC. One is NetCDF *Processing*. It starts by watching for one dimensional (1D) diagnostic variables of XGC stored in NetCDF files. As each file grows (they are extended after every diagnostic period), the workflow efficiently mirrors them on the processing site. Images are generated using *xmgrace* for all output variables and for each diagnostic time step. Images are then placed into a predefined directory. This directory is accessible to dashboard. The *HDF5 Processing* pipeline transfers binary files written in BP format - an efficient binary output format using ADIOS [13], these files are individually written for each diagnostic time step. The BP files are asynchronously converted to HDF5 using an external transformation code and then images are created for all two dimensional (2D) slices of the three dimensional (3D) data stored in those files using an *AVS/Express* network. The pipeline starts AVS/Express job on the processing site when the first file is transferred, and then uses it as a (private) service by making imaging requests to it throughout the whole run.

The complete workflow is built from the Kepler's Java components (called Actors – those are the green boxes in Figure 1), as well as from the python and bash scripts, and C programs that run on the underlying nodes (Figure 2). While the Kepler-level flows are of relatively low intensity, computational and data-transfer actions can be quite intense on the nodes that perform

[1] https://fusion.gat.com/global/DIII-D

simulations, complex analytics, or on the networks that move large amounts of data. These composite actors perform the many operations needed to affect the coupling run. All major steps (M3D-OMP, ELITE, M3D-MPP, NetCDF Processing, and HDF5 Processing) are sub-workflows that execute the scripts and programs on remote hosts to accomplish specific tasks.

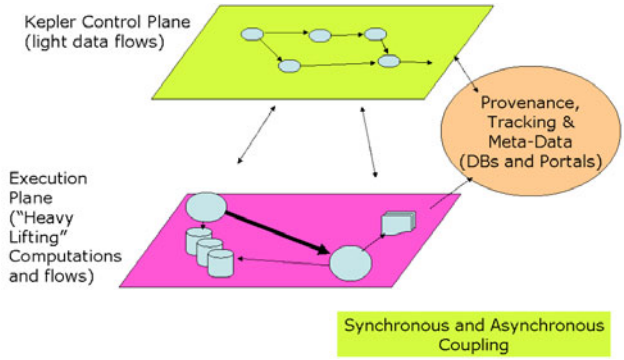

Figure 2. Kepler Control Flow Orchestrates Simulations, Analyses and Data Collection.

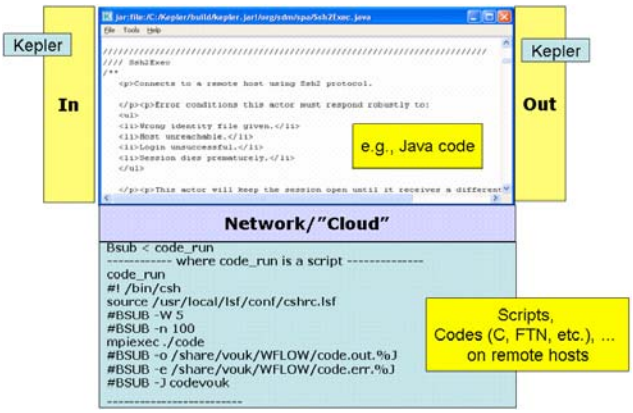

Figure 3. Illustration of an Extended Actor

Figure 3 illustrates the relationship between the control plane actor, and the actor parts running on remote hosts. Typically, the control plane (Kepler) actors are written in Java, while its remote parts can be written in any appropriate language. In the example below the remote component is an LSF job submission script. The progress of the workflow can be followed on the dashboard in the form of images and movies, as illustrated in Figure 4, and other relatively granular indicators. It can also be followed over the Kepler graphical user interface, or by querying the Provenance data-base directly.

The thousands of small operations, information about flows, code versions, location of all files, and so on, stay hidden from the user unless the user wishes to access that information. For example, when the user selects a plot to display, he/she may also want to download the corresponding data, for that full data-paths may be needed. Similarly, a user may wish to compare several variables from different simulation runs in order to detect any subtle differences between variables. It is important to shield the user from the manual operations such as: get the raw data for variable A from run 1, get the raw data from variable B from run 2, and then overlay this information. To repeat a coupling run, we also need information about the codes and inputs used.

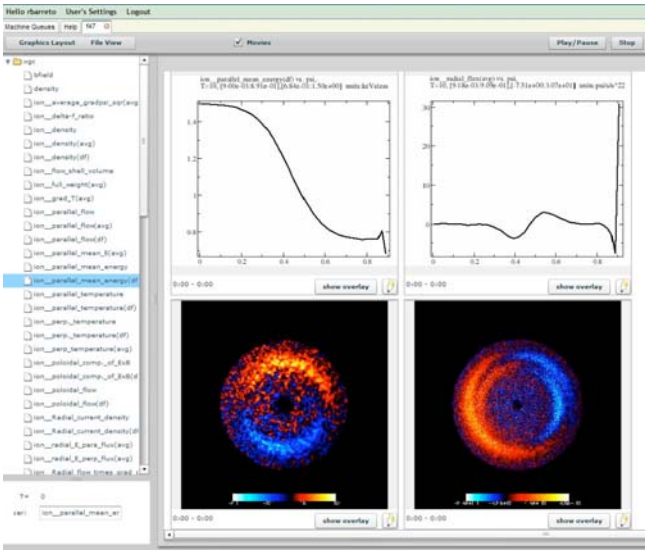

Figure 4. A Snapshot of the eSimMon

Collection of all such information is the role of the provenance recording and review system we have built around our workflow system. The top-level architecture of the system is illustrated in Figure 5. We collect four different categories of information:

Process provenance which is about the steps performed in the workflow, the progress through the workflow control flow, etc. This execution-time information is automatically recorded by the Kepler Provenance Recorder component, and its execution-plane counterparts;

Data provenance which records history and lineage of each data item associated with the actual simulation (inputs, outputs, intermediate states, etc.);

Workflow provenance which is about the history of the workflow evolution and structure;

System provenance which is about recording of all external (environment) information relevant to a complete run. This information includes compilation history of the codes, information about the shared libraries used by the codes, source code related information and run-time environment settings of each participating machine, etc.

The Provenance information we collect should allow complete end-to-end audit of individual runs as well as comparison among the runs. It may also need to be collected at different levels of granularity and by different mechanisms. The architecture in Figure 5 illustrates this. Central to the solution is a "data store" (currently a relational data-base, but soon RDF capability will also be added). Access to the data store is via recording, query and management API's. Kepler delivers its provenance information via its Provenance Recorder, while some of the supercomputer and analytics scripts and codes may write to it directly. Administrators access the system through the management interface, while user portals (such as eSimMon) access it via the query API.

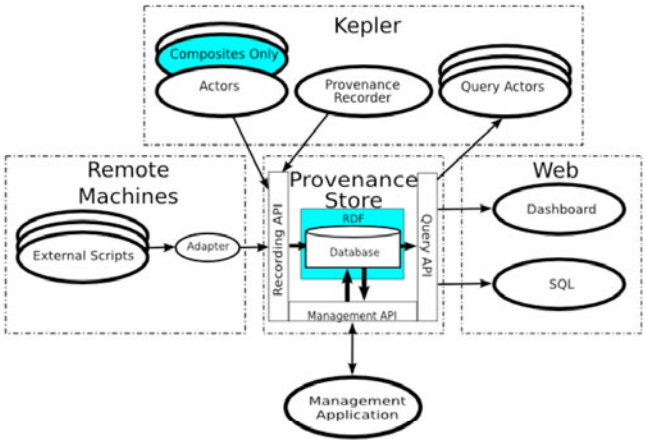

Figure 5. Illustration of the Provenance Collection System Architecture

4. eSimMon BASICS

The eSimMon is a web-based interface that presents scientists with several views: one view gives the scientist an overview of the activity on the computers of interest at DOE, another view allows the scientist to monitor data provenance and analytics related to either live runs (we call shots) or old shots, yet another view (under construction) focuses on display of other provenance information and status tools, and so on.

The purpose of the eSimMon is to provide simple "one-stop shopping" access to the status of the simulations, launch new simulations, and analyze/re-visit existing runs. It is a light-weight front-end for comprehensive end-to-end data and workflow management that provides users with access to large simulation datasets, and simulation and workflow provenance and performance records. Users can examine evolution and outputs of their runs or drill into run-time processes, data, workflow and system information. The dashboard offers features that range from reporting how many jobs are running on a particular DOE supercomputer, to viewing performance text files, to interacting with images and flash movies that describe the output of the simulations, and so on. In the near future, its capabilities will be extended to include submitting jobs and interacting with the workflow.

The dashboard is a Web 2.0 rich internet application based on LAMP (Linux, Apache, MySQL, PHP), and AJAX (JavaScript, XML) server-side environments, and Flash content created using Adobe's Flex SDK in the browser. AJAX and Flash create dynamic pages with local interaction and asynchronous server communication which provides data extraction, manipulation and complex visualization on the back-end.

A typical process that produces visualization media shown on the dashboard may run as follows (Figure 6). A simulation creates a large amount of data at each time step. This data is appended to HDF5 and NetCDF files. A parser creates a listing of the variables available in the NetCDF and HDF5 files and stores the list in the database. A process runs through every variable listed in the database and uses xmgrace to create xy plots of the data from the NetCDF data, and contour plots using the HDF5 data. The plots are then encoded as frames in FLV movies for viewing on the dashboard (as in Figure 4), or are loaded into the dashboard as simple images.

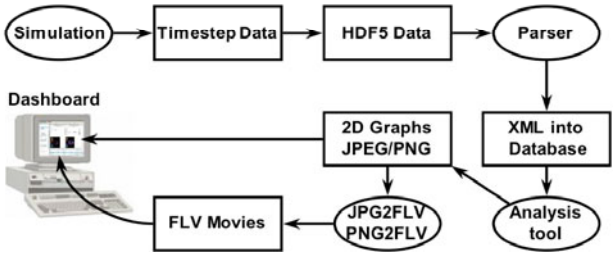

Figure 6. Dashboard-Workflow Architecture

The current version of the dashboard performs two major functions: machine monitoring and simulation monitoring. The collaborative nature of its use starts when users look at their jobs which are either old, or running jobs. The user then looks at the data from the shot, and possibly informs other collaborators to look at this shot.

Users can view and query computing resource queues (Figure 7), to see job submission status. Users may also decide who they want to share their work with. eSimMon provides collaborator management through the database. This allows the users to view not only running and past simulations but also the jobs belonging to collaborators who have authorized them to do so.

Selecting a job takes the user from the machine monitoring page to the simulation monitoring page (Figure 4) of the eSimMon where data produced by the simulation job can be viewed. Collaboration between users expands here with the notion of adding notes, annotating images and movies produced from simulations (e.g., Figure 8), and session management. These features are currently being developed and tested on the eSimMon. The collaborative work can be categorized into passive and active teamwork.

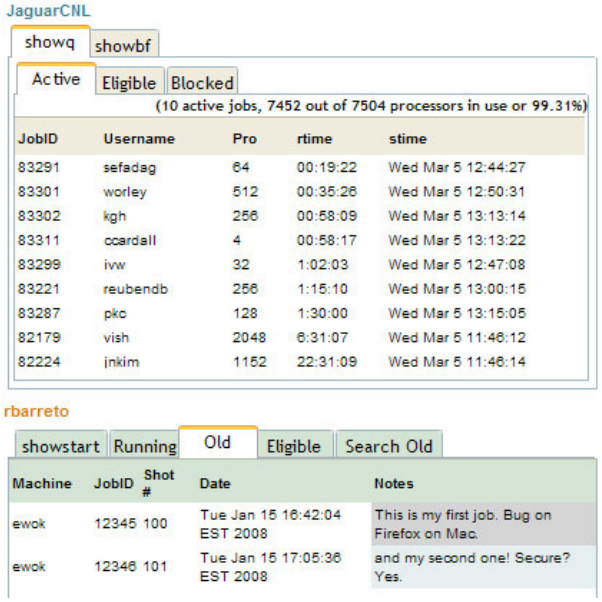

Figure 7. Machine Monitoring Page

Passive collaboration describes work being done by each user at a particular moment in time independently of what his or her collaborators are working on at the same time. The ability to collect notes on a particular run, annotations on images (Figure 8) and movies generated by the simulation and to save them in the database combines with support for collaborative users to allow sharing thoughts and ideas about the science. An additional essential part of passive collaboration is shot comparison: it would allow scientists to juxtapose two shots. For example, two users working on the same code could conveniently compare shots. Another illustration of the usefulness of this feature would be the case in which a user gets bad performance on a specific run and notifies a performance scientist about the issue. The latter can track the provenance and performance data for two shots and pinpoint the origin of the problem. The shot comparison idea will also be extended to comparing shots and experimental data. The eSimMon will have access to outside tools as well as its own database, such as MDS+ software tools for external data acquisition for fusion data [14]. Once we allow for access to external data, customized eSimMons will be developed for individual projects. One of the lessons that we have learned is not to overwhelm the users with too many options. Therefore, customization for each project will occur once we start adding more analysis features, as well as access to external data sources.

The concept of active collaboration, implemented as *sessions,* pushes updates to the user every time a collaborator takes an action such as playing a movie, adding an annotation to an image or a comment to the notes. This way, users who are logged on to the dashboard at the same time can see each other's changes to the same simulation data. Each user's contribution becomes easily distinguishable by label tagging or associating a particular user with a default and unique color so that, for example, information entered by this particular user would be displayed only in that particular color. The user would have access to this color/tag map of collaborators, and in future versions would be able to customize it. Naturally, even though all collaborators would be able to edit the notes for a shared simulation, only the owner will have the absolute administrative rights to edit and delete entries from others. Session management is discussed further in section 5.2.

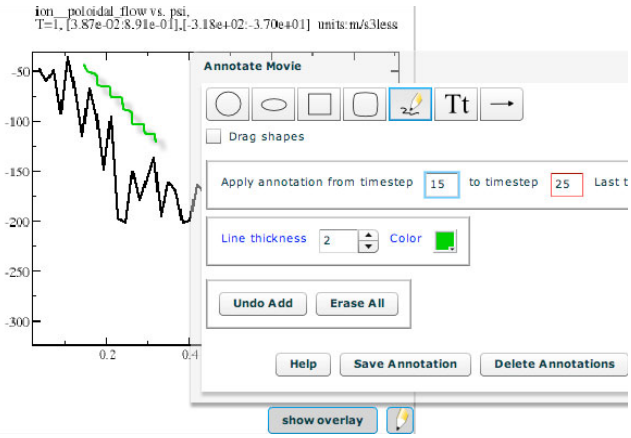

Figure 8. Annotations

5. FUTURE IMPROVEMENTS

5.1 Interactive Visualization

The purpose of adding interactive visualization to the eSimMon is not to replace existing visualization tools, but rather to add to the dashboard user's analysis toolkit. A browser-based tool combined with simulation monitoring allows the user to quickly assess the success of both a running and a completed simulation without waiting for visualization jobs to execute on a server. They also could be used to supplement existing visualization workflows.

In addition to efficiently creating dynamic and multimedia content, the Flex SDK also contains charting components that can be used to draw plots in the browser using vector graphics (xy, scatter, line plots, etc.) from data pulled from a web server. Our initial efforts have produced a tool that leverages charting components to plot particles and 2D slices and that uses query-driven visualization to identify and isolate regions containing interesting data points.

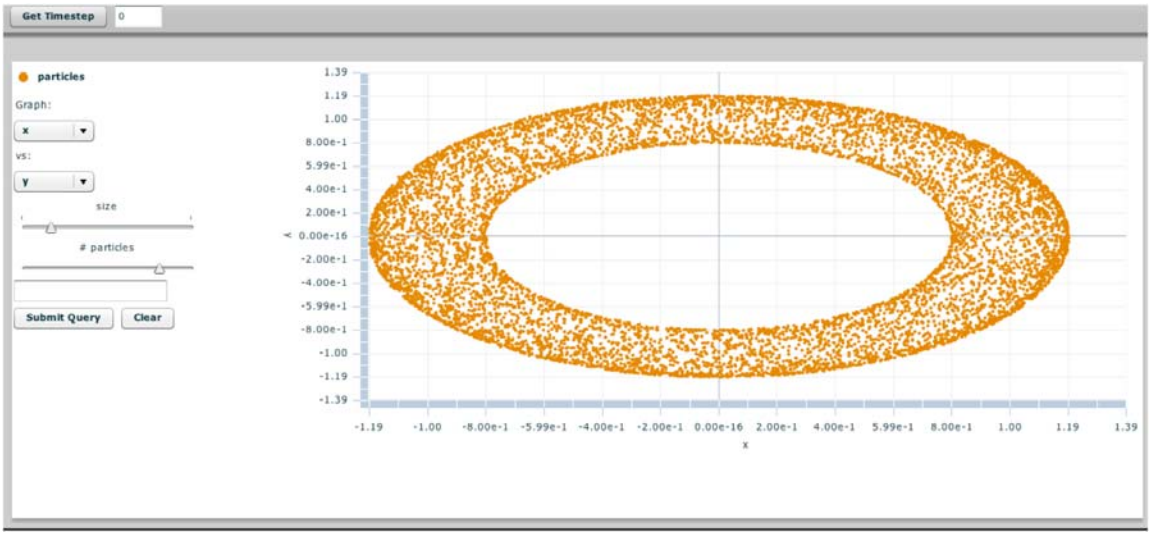

Figure 9. Query Tool

This plotting tool prototype (Figure 9) requests data sets by variable name from the server and plots the results, allowing the user to filter the number of data points displayed. The user may perform basic analysis by entering range queries on the data which are sent to server. The server calls a data processor (Figure 10) implemented in C++, which parses the query, reads the dataset and filters the data. The results are returned to the browser.

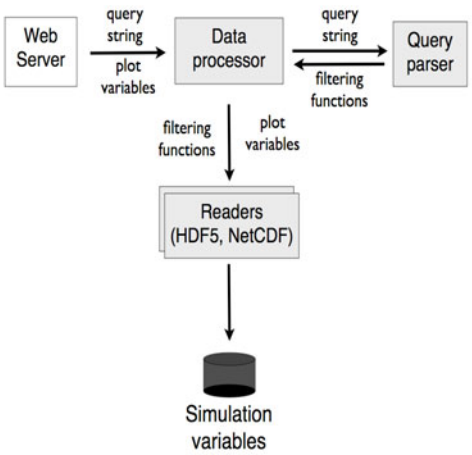

Figure 10. Data Processing Backend

This basic visualization and analysis tool is being expanded to offer some common plotting tools (plot comparison, linear regression, curve fitting, etc.) and a more intuitive interface for entering queries and interacting with the plot. The basic data processor will not scale to handle large datasets, so the query mechanisms on the back-end must be improved by efficiently indexing the data using tools such as the compressed bitmap index technology [15] behind the FastBit query engine [16][17] for fast lookups. The next major development thrust will be in integrating visualization tools that lend themselves well to client-server architecture, such as VisIt [18], by adding an interface the dashboard front-end that would allow the user to send data and processing commands and stream images back to the browser.

5.2. Session Management and Pushing Data from Database to Dashboard

Currently our collaborative dashboard system only allows applications to push data to the dashboard, and does not manage any sessions. When multiple users are looking at the same shot, each user can add annotations to that shot, but the system does not update a user's display while another is working on it. In order to fully create a collaborative environment, we are adding a session manager to our dashboard system.

The session manager keeps track of the entire user on the system. When multiple users are looking at a particular shot, the session manager updates the database every time one of the users makes a change. Typically this occurs when a user uploads new data to the system, or when the user annotates particular text. Our system allows a PHP

script to push data to all of the other users of the system when a change is made. This allows scientist to annotate individual plots or movies and sharing these annotations with other users in real-time in a collaborative manner.

Another important aspect of the session manager is that it allows one user to drive the session while the other users follow along. For example, if one user plays movie(s), the system at the other users will automatically play the movie(s), without any user input. Our system allows users to either be in a collaborative or 'stealth' mode when viewing data. In the 'stealth' mode, they do NOT get updates when other users are just looking at individual shots. This allows users to be productive in collaborative and non-collaborative setting. Note that when an action is performed, for instance the playing of a movie, the frames will not be in synch, but when the stop action is performed, the system tells all of the other users to stop playing the movie, and advance to the frame that the master collaborator is looking at. In this way, the scientist can then begin to annotate and discuss the frame. This approach reduces the overheads of managing the collaborations while still producing the desired collaboration experience.

We will integrate a simple chat system into our dashboard and enable users to journal the transcripts (if they want to store the annotations when the collaboration is ended). A key design goal of the collaboration environment is to minimize the network activity during collaboration, so that the system does not become overloaded and unusable. This is achieved by minimizing the traffic between the server and the user sites.

One other important area of collaboration is when a scientist wants to compare different datasets from one or more shots, or from multiple experiments. This means that we must have a suite of real-time analysis tools that run from a server and enable data comparison. For instance, scientists may want to look at the scalar potential from two fields, and discuss this with their colleagues. The operations are simple, but careful data management must be employed. For instance, the two runs may require that their data be interpolated (if they come from different geometries) allowing only the overlapped regions of space where the data can then be compared. The user may want to subtract the two quantities to see the difference, and display these results. These operations may not exist in the system so we need to create our system such that collaborators can 'plug-in' their own analysis routines. Typically such plug-ins will be programs that take in an input dataset of a known format, and output new transformed data. For instance, a plug-in will take two files (A,B), read in the mesh data and the data on top of the mesh, interpolate the data on mesh B to mesh A, subtract the quantities, and then output the mesh A with this new result. The resultant analysis piece may then be coupled with a visualization piece that the user chooses, such as a VisIt network, to create an image of this data. Since the data is typically time-varying the analysis workflow will then perform this operation for all of the relevant time slices. In the collaborative sense, we must ensure that users can upload their analysis (plug-in) routines, and have workflows created to execute these when the user wants this.

5.3. Advanced Visualization on the Dashboards for Collaboration.

Currently, there are two modes of visualization being utilized. The first involves post-processing interactive visualization where data is downloaded to the scientist's local workstation and interactively analyzed using a visualization environment. This is exploratory in nature and allows the scientist to interrogate the data through visualization processing. However, with peta-scale computing and the expanding size of the datasets being computed, it is becoming increasingly important to reduce the amount of data that needs to be interactively investigated and to try to automate the visualization by preprocessing the data as much as possible. Once a particular set of analysis/visualization routines has been investigated and tested, these routines can then be made part of the workflow and the visualization result viewed on the dashboard. This is the second mode of visualization. We envision empowering the dashboard with a limited set of interactive and 3D capabilities to enable simple querying of the data. For example, when viewing slices from an XGC run, one can view them as a movie through the dashboard (as described previously). Because of the complex geometry of the plasma-edge simulation (data is only on the edge of the ring), it helps to view the data in 3D to maintain temporal coherence [19]. An example is shown in Figure 11 with two different rotated views. Other types of 3D views include volume rendering, isosurfaces, streamlines etc. Some (like volume rendering) can be precomputing and just displayed on the dashboard, others that involve 3D geometry need a more interactive framework for appropriate viewing.

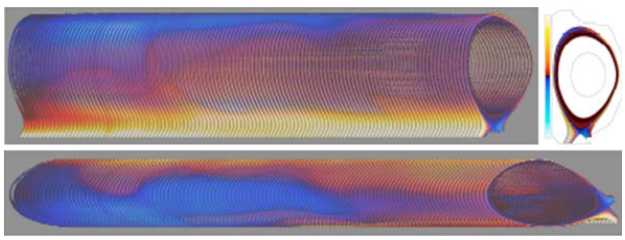

Figure 11. "Curtain Rod" view.

To facilitate collaboration, we hope to implement an annotation capability which will allow scientists to

comment directly onto the dashboard and make those comments available for others to see. This includes annotating images with typed text and freehand drawings, and even annotating 3D visualizations (which has the added difficulty of viewing the annotations correctly). Annotations may also be related web sites for comparisons. Another annotation could be a chat between scientists who are discussing a particular visualization (chat using texting or even voice if available). These conversations can be tagged to a particular visualization. As the provenance system logs everything, the chats between scientists can be retrieved so that others can view the discussion (if given permission) which can aid in understanding the simulation results and is especially helpful for new students or researchers joining the project.

On the horizon, we envision a full 3D virtual space to facilitate collaboration much like the 3D environment of Second Life (secondlife.com). Scientists could enter the virtual "room" which will display results from the dashboard and other visualization and/or analysis. As with virtual spaces, scientists can chat with other scientists who are there looking at their data and/or hold meetings all looking at the same images in both two and three dimensions. 3D annotations could be seen and collaboratively added in real time. Scientists can also invite others to see and discuss their results enabling a more dynamic approach to collaboration and possibly a more effective medium for communicating the results to a general audience.

6. CONCLUSION

Our work has initially been focused on using a workflow environment during live simulations to capture and record information which goes to our dashboard system. Our dashboard uses web-2 technologies to provide state of the art ways to analyze results from simulations. The backend of the dashboard is a database which records the provenance information, and can allow researchers to see the lineage of the data, as well as the provenance information from the codes used during the simulation. Our system is currently used by both the GTC team members, as well as the CPES team members.

REFERENCES

[1] ITER Home Page: *http://www.iter.org*

[2] http://www.scidac.gov/FES/FES_CPES.html

[3] *http://sdm.lbl.gov*

[4] J. Cummings, A. Pankin, N. Podhorszki, G. Park, S. Ku, R. Barreto, S. Klasky, C. S. Chang, H. Strauss, L. Sugiyama, P. Snyder, D. Pearlstein, B. Ludäscher, G. Bateman, A. Kritz, and the CPES Team, "Plasma edge kinetic-MHD modeling in tokamaks using Kepler workflow for code coupling, data management and visualization", *Communications in Computational Physics* special issue for the *20th International Conference on the Numerical Simulation of Plasma* (Austin, TX, Oct 2007).

[5] S. Ku, C. Chang, M. Adams, J. Cummings, F. Hinton, D. Keyes, S. Klasky, W. Lee, Z. Lin, S. Parker, and the CPES team, "Gyrokinetic particle simulation of neoclassical transport in the pedestal/scrape-off region of a tokamak plasma." *Institute of physics Publishing Journal of Physics: Conference Series*, 46, pages 87–91, 2006.

[6] W. Park, E. V. Belova, G. Y. Fu, X. Tang, H. R. Strauss, L. E. Sugiyama, "Plasma Simulation Studies Using Multilevel Physics Models", *Phys. Plasmas* **6**, 1796 (1999).

[7] C. R. Sovinec, A. H. Glasser, T. A. Gianakon, D. C. Barnes, R. A. Nebel, S. E. Kruger, D. D. Schnack, S. J. Plimpton, A. Tarditi, M. S. Chu, and the NIMROD Team, "Nonlinear Magneto-hydrodynamics with High-order Finite Elements", *J. Comput. Phys.* **195**, 355 (2004).

[8] B. Ludäscher, I. Altintas, C. Berkley, D. Higgins, E. Jaeger, M. Jones, E. A. Lee, J. Tao, and Y. Zhao, "Scientific Workflow Management and the Kepler System", *Concurrency and Computation: Practice & Experience* **18**(10), 1039 (August 2006).

[9] Kepler project, http://www.kepler-project.org.

[10] C.S. Chang and S. Ku, "Property of an X-point generated velocity space hole in a diverted tokamak plasma edge.", *Phys. Plasmas* **11** 5626 (2004).

[11] P.B. Snyder, H.R. Wilson, J. R. Ferron, L. L. Lao, A. W. Leonard, T. H. Osborne, A. D. Turnbull, D. Mossessian, M. Murakami, and X. Q. Xu, "Edge localized modes and the pedestal: A model based on coupled peeling–ballooning modes", Phys. Plasmas **9** 2037 (2002).

[12] N. Podhorszki, B. Ludäscher and S. Klasky, "Workflow Automation for Processing Plasma Fusion Simulation Data", *2nd Workshop on Workflows in Support of Large-Scale Science*, Monterey, CA, June 2007, pp 35-44.

[13] ADIOS: *http://www.cc.gatech.edu/~lofstead/adios*

[14] *http://www.mdsplus.org/index.php/Introduction*

[15] K. Wu, W. Koegler, J. Chen, and A. Shoshani. "Using bitmap index for interactive exploration of large datasets." In SSDBM 2003, pp. 65-74, 2003.

[16] K. Stockinger, J. Shalf, K. Wu, W. Bethel. "Query-Driven Visualization of Large Data Sets." In Proceedings of IEEE Visualization 2005, pp. 167-174. Minneapolis, MN., October 23-28, 2005.

[17] FastBit: *http://sdm.lbl.gov/fastbit*

[18] VisIt: *https://wci.llnl.gov/codes/visit*

[19] K. Bennett, D. Silver, C. Correa, S. Klasky and S-H. Ku, "Shape-Aware Focus and Context Views for Plasma Turbulence Simulation", Poster for IEEE Visualization 2007, October 2007.