EFFIS: an End-to-end Framework for Fusion Integrated Simulation

Julian Cummings
Center for Advanced Computing Research
California Institute of Technology
Pasadena, CA 91125, USA
cummings@cacr.caltech.edu

Jay Lofstead and Karsten Schwan
College of Computing
Georgia Institute of Technology
Atlanta, GA 30332, USA

Alexander Sim and Arie Shoshani
Lawrence Berkeley National Laboratory
Berkeley, CA 94720, USA

Ciprian Docan and Manish Parashar
Dept. of Electrical and Computer Engineering
Rutgers University
Piscataway, NJ 08854, USA

Scott Klasky, Norbert Podhorszki, and Roselyne Barreto
Oak Ridge National Laboratory
Oak Ridge, TN 37830, USA

Abstract—**The purpose of the Fusion Simulation Project is to develop a predictive capability for integrated modeling of magnetically confined burning plasmas. In support of this mission, the Center for Plasma Edge Simulation has developed an End-to-end Framework for Fusion Integrated Simulation (EFFIS) that combines critical computer science technologies in an effective manner to support leadership class computing and the coupling of complex plasma physics models. We describe here the main components of EFFIS and how they are being utilized to address our goal of integrated predictive plasma edge simulation.**

Keywords-fusion simulation; code integration; computational framework; leadership class computing

I. Introduction

The Center for Plasma Edge Simulation (CPES) is one of three multi-institutional research programs initiated by the US Department of Energy (DOE) in 2005 to address pressing needs for more advanced predictive modeling capabilities in the area of magnetically confined fusion plasmas. CPES and the other centers were established as key components of the Fusion Simulation Project (FSP) to investigate a range of physics issues deemed crucial to the understanding and efficient design and operation of next-generation tokamak fusion reactor devices such as ITER (the International Thermonuclear Experimental Reactor). Because the core confinement properties of tokamak plasmas are known to have strong correlation with plasma edge conditions, CPES was specifically tasked with developing a new integrated and predictive plasma edge simulation package applicable to existing magnetic fusion devices and future burning plasma experiments like ITER. This ambitious goal requires cooperation across several research institutions and a well coordinated effort involving experts in plasma physics, computer science, and applied mathematics.

The physics research focus of CPES has been to study the interplay of neoclassical transport, microturbulence, and magnetohydrodynamic (MHD) instabilities in the plasma edge. Neoclassical physics and microturbulence are most completely modeled using a kinetic approach such as a particle-in-cell (PIC) simulation, while MHD modes are more efficiently studied with a fluid code. Hence, an integrated code framework is needed to couple newly developed edge kinetic simulation capabilities with existing state-of-the-art MHD models using the most advanced computer science technologies. Such a framework can enable physicists to study the dynamic interaction of kinetic effects that cause a buildup of the edge pedestal in plasma density and temperature profiles and large bootstrap currents with so-called Edge Localized Modes (ELMs) that may limit pedestal growth and tokamak reactor performance [1]. An integrated model of this pedestal-ELM cycle that can make accurate predictions of such features as the maximum pedestal height and width and the strength of heat flux to device wall components due to ELMs is critical to the overall goals of the FSP.

Most of the computer science efforts of CPES have been to develop tools to facilitate this integrated plasma edge model within the larger scheme of an End-to-end Framework for Fusion Integrated Simulation, or EFFIS. The coupling of kinetic and MHD codes to address pedestal-ELM physics poses many challenges in terms of data exchange and code activity coordination and provides an excellent driver for the development of useful computer science technologies. However, the process of research within CPES has demonstrated that our computer science needs stretch well beyond basic code coupling and into the areas of massive data I/O, post-processing and archival storage, close collaboration of simulation scientists via Web technologies, and the integration of existing and widely accepted analysis tools to provide a complete end-to-end system for effective large-scale computing and simulation. EFFIS is intended to address these needs by providing a set of computer science tools that present the physics researcher with a simple means of planning, launching, monitoring, analyzing and sharing fusion plasma simulations on leadership class petascale computing platforms.

II. Leadership Class Computing and EFFIS

A. *Description of Leadership Class Computing Facilities*

Leadership class computing has brought unprecedented amounts of supercomputing power to open science computing. There are three major US Department of Energy (DOE) computing facilities: Argonne National Laboratory (ANL), Oak Ridge National Laboratory (ORNL), and the National Energy Research Scientific Computing Center (NERSC). The two major computing platforms at these facilities are the IBM Blue Gene P and Cray XT-5 systems. Each computer has over 100K cores with fast file systems, but obtaining high levels of performance for application codes has proved to be challenging.

One of the new programs in DOE intended to utilize high number of processors for simulations running on these machines is the INCITE program [2]. Current users have access to over 75M CPU hours in one year. If we compare this to the fastest computer that was available just four years ago in DOE open science computing, the Cray XT-3 at ORNL with 3748 cores, we see that users are getting the equivalent of 2.3 CPU years of processing power of four years ago. This now gives users the ability to compute for much longer, on much higher processor counts, and generate much more data in a simulation. Data I/O and workflow monitoring have become critical for large simulations and are now a crucial part of the CPES and other projects. Code coupling is also becoming more common as users begin to address physics issues that span multiple spatiotemporal scales and require simulation models that are tuned to different physical regimes or conditions.

B. *Brief Description of EFFIS*

The End-to-end Framework for Fusion Integrated Simulation (EFFIS) has successfully been used to automate integrated simulation of XGC0 [3] kinetic edge pedestal buildup, ELITE [4] stability-boundary check, M3D [5] edge localized mode crash, and back to XGC0 pedestal buildup, for multiple ELM cycles. EFFIS has also been recently used to monitor long running XGC1 simulations of microturbulence in the plasma edge, and to monitor core turbulence simulation codes GEM, GTC, and GTS. EFFIS uses highly advanced inter-operating computer science tools that include fast adaptable I/O, workflow management, fast presentation of image data on a web-based dashboard, the collection and management of provenance metadata, and wide-area data transfer.

One of the cornerstones of EFFIS is the ADaptable I/O System (ADIOS), which provides access to a variety of data transport mechanisms and data formats. The second main piece of this framework is the Kepler workflow system, which is used to orchestrate many parallel tasks on multiple computer platforms. The final component of this framework is the eSimMon dashboard, where users can monitor and analyze long running simulations on the web. A diagram of EFFIS is shown in Figure 1.

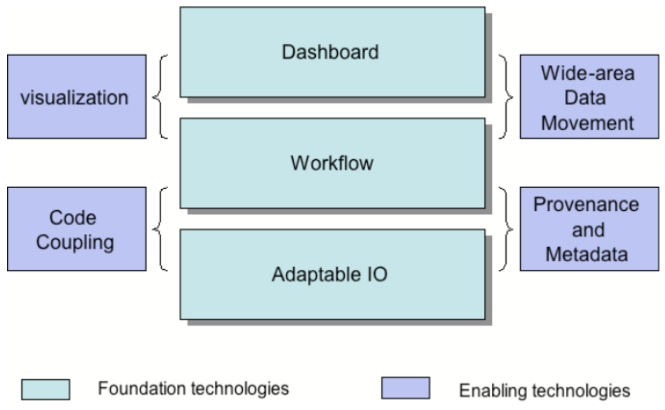

Figure 1. Schematic diagram of EFFIS components.

One of the keys to the technologies is the provenance and metadata capturing system, which is enabled in all aspects of EFFIS and allows provenance information to flow from the compute nodes over to the provenance capturing system. This allows users to operate on the data without knowledge of files, but simply using the names of the variables in the simulation. Furthermore, users on the dashboard can analyze the data and move the data to remote resources using SRM-lite for WAN data movement.

Visualization is another key to our system, as this combines high-speed data I/O with analysis. Our team has been working to provide adaptors for both Matlab and Visit to read in ADIOS-BP binary data files, yielding high speed parallel methods to input and output data with these analysis and visualization packages.

III. Code Coupling for the Pedestal-ELM Cycle

As discussed in Section I, the precise details of the cyclical process of edge pedestal buildup in the plasma density and temperature profiles that can drive ELM instabilities, followed by a nonlinear ELM evolution and "crash" that releases free energy from plasma profile gradients and creates a modified MHD equilibrium, is of great interest in magnetic confinement fusion research. A procedure for coupling kinetic and MHD codes together to study this pedestal-ELM cycle have been reported elsewhere [6] and will only be briefly recapped here. In this coupling scenario, the gyrokinetic PIC edge simulation code XGC0 [3] reads an *eqdsk* file contained fitted magnetic equilibrium data for a fusion device and experiment of interest. The code is initialized with model profiles for the plasma edge density and temperature, and it simulates the edge pedestal buildup due to kinetic effects. XGC0 periodically dumps plasma density, temperature, and bootstrap current data into an *m3d.in* file, which is processed by the M3D-OMP code using a Grad-Shafranov solver to generate an updated *eqdsk* file. XGC0 can then reread the *eqdsk* file in order to continue evolving the plasma particles through the updated equilibrium.

At the same time, the linear MHD stability of the updated equilibrium is checked by passing the *eqdsk* file, along with a *peqdsk* file containing plasma edge density profile data from XGC0, to the ELITE code [4]. This ideal MHD code can quickly assess the linear growth rate of several sample ELMs in the intermediate mode number range of n=5-30 and identify any modes that exceed a critical instability threshold. When an unstable mode is found, we can stop the XGC0 kinetic model, launch a nonlinear simulation of the ELM using the parallel M3D-MPP code [5], and then eventually recover a modified MHD equilibrium from this simulation that can be used to begin a new XGC0 run that will start the cycle over again.

The pedestal-ELM cycle simulation described here requires quite a lot of intricate coordination of activities between the four main simulation codes involved. In addition, there are a few utility codes and post-processing tools that play important roles in allowing the main codes to interact and enabling the user to effectively monitor and analyze the coupled simulation. The orchestration of such activities within EFFIS is performed by the Kepler workflow system, as described in Section V.

This kinetic-MHD code coupling uses a memory-to-disk and disk-to-memory approach, and the data I/O can be prescribed and managed using ADIOS. The amounts of data exchanged between codes are relatively small, however. The axisymmetric magnetic equilibrium data is typically given on a 257x257 mesh, and the plasma profile data are 1D functions of the poloidal magnetic flux with 50-100 points each. Thus, a valid alternate approach is to use memory-to-memory coupling, thus avoiding the overheads of reading, writing, and copying lots of small data files. The use of this technique to couple the XGC0 and M3D-OMP codes is discussed in Section IV.C below.

Another more detailed version of the pedestal-ELM problem that is currently being studied involves running the XGC0 and M3D-MPP codes simultaneously during the ELM crash phase and performing periodic exchanges of plasma profiles and the 3D perturbed magnetic fields. The main purpose of such a coupling is to examine the dynamic effects of ELM evolution on the heat flux into the divertor region in the kinetic simulation. Naturally, such a "tight" code coupling would require larger, more frequent data exchanges.

IV. ADIOS

A. *Overview of I/O Challenges and ADIOS Architecture*

Researchers have been continuing to face significant problems while optimizing I/O on leadership class computers, workstations, and even single-processor workstations. They are forced to make difficult coding decisions and to use either complex APIs, such as parallel HDF5 and NetCDF, or simple non-metadata rich I/O in binary or ASCII format. These decisions often force them into using non-optimal I/O practices when porting their code to new architectures, and may even render the I/O useless.

The ADaptable I/O System (ADIOS) [7] is essentially a componentization of the I/O layer. It provides an easy-to-use programming interface, which can be as simple as Fortran file I/O statements. ADIOS abstracts I/O metadata information and data structures from the source code into an external XML file, which can reduce code pollution and create the connection between high-level APIs and underlying I/O implementation details, such as buffering and scheduling. By separating the detailed I/O implementation from the APIs, ADIOS also allows users to simply change the declaration of the transport methods in the XML file without any source code modification. Figure 2 illustrates the architecture of the ADIOS framework.

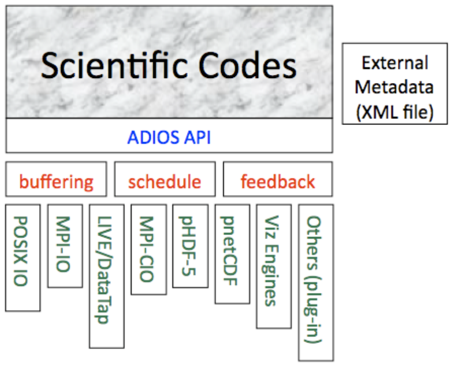

Figure 2. ADIOS framework architecture.

The ADIOS architecture exploits modern web technologies by using an external XML metadata file to describe all the I/O-related variables used in the code. These collections of data are described in terms of groups, which typically correspond to the individual subroutines. The file catalogues the following metadata for each element of these collections: the element name, data type, element path (similar to HDF5 path), and static or dynamic array size, as well as any annotations. If the array represents a mesh, information on the global bounds of this array and ghost regions used in real-time visualization is encoded at the XML group level. For each data collection/group, it describes the selected transport mechanism and parameters, as well as timing information for the data transmissions. Using this information, the ADIOS I/O implementation can then control when, how, and how much data is written or transferred at a time, thereby enabling efficient overlapping with computation phases of the scientific application and proper pacing to optimize the writing or transmission throughput.

Some of the key features of ADIOS are that it allows for extremely fast I/O [8], has a high level of resiliency [9], and can be used for creating multiple operations in I/O staging [10]. FSP projects will eventually need to couple multiple codes together. ADIOS can also be used to select code coupling methods; for example, to switch between file I/O and memory-to-memory code coupling. Users can select at runtime which method they want. Finally, using the provenance capturing method [11] ADIOS can capture the provenance information in both cases, without changing the codes or the workflow.

B. *ADIOS Performance*

ADIOS development has initially concentrated on write performance. In Figure 3, we show that we can achieve 80 GB/s when writing data on the Cray XT-5 at ORNL. This level of performance allows researchers to write metadata-rich data, in the ADIOS-BP format, which can easily be converted to HDF5 or NetCDF.

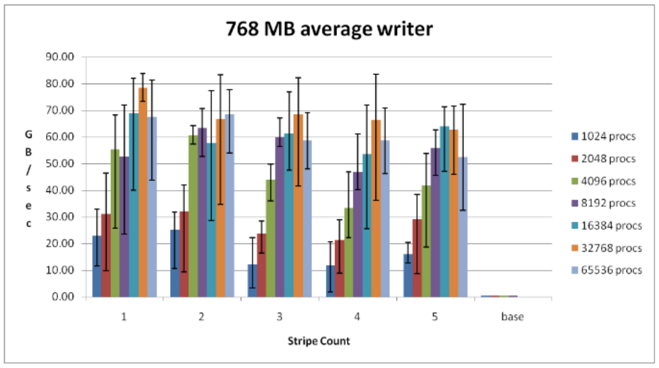

Figure 3. ADIOS write performance on Cray XT-5.

Our recent studies have now concentrated on read performance of ADIOS, and compare the performance of reading ADIOS-BP files to reading files from parallel NetCDF. We choose to compare with parallel NetCDF since this file format keeps a logically contiguous view of the data. Our initial results show that for GTC particle data written from 32K cores, over 32GB of data can be read into 256 processors in less than one second.

C. *DataSpaces framework*

The fusion simulation applications targeted by EFFIS consist of coupled, complex parallel computer codes that run independently, progress at different speeds, and have to cooperate at run time by exchanging parameters and data values. DataSpaces is a dynamic and asynchronous interaction framework that provides the abstraction of a virtual distributed shared space. This space can be associatively accessed by application nodes using a simple API (for example, *put()* and *get()* calls) and with semantically meaningful addressing (*e.g.*, the multidimensional coordinate space discretization used by the application). Access to DataSpaces is transparent; that is, it is independent of data/node location and distribution.

The architecture of the DataSpaces framework consists of three key layers. The *communication and data transport layer* provides the core communication functionality for data transfers as well as control of message exchange between the compute nodes of the space. This layer is based on DART [12], which allows fast decoupled and asynchronous remote data transfers with low latency and small overheads. DART is implemented using the RDMA communication paradigm via the Portals RDMA library, and it enables direct memory-to-memory communication. It provides flexible asynchronous APIs that allows an application to overlap computation with data communication and thus reduces the overheads due to I/O operations. DART is available as an optional transport method within ADIOS [7].

The *directory layer* of DataSpaces enables nodes to query metadata associated with the data in the shared space using semantically specialized query abstractions. It is implemented as a dynamic hash table across the nodes of the space, and it provides load balancing support when data is inserted in the space, as well as look-up support when data is retrieved from the space. It also supports subscribe/notification interactions and includes garbage collection mechanisms.

The *storage layer* maintains the actual data (*i.e.*, the objects put into the space). It provides a data coherency protocol, which defines when an object can be inserted, updated, destroyed or removed from the space. This protocol ensures synchronization when the data objects are accessed by multiple application nodes.

The API provided by DataSpaces is simple and flexible and enables multiple usage patterns for applications coupling and interactions. The example shown in Figure 4 demonstrates the use of DataSpaces to couple two fusion codes, XGC0 and M3D-OMP, which run on different numbers of nodes on the *jaguar* system at NCCS and have different data decompositions. The execution of each code is dependent on data values and parameters produced by the other code. Specifically, XGC0 computes the plasma radial profiles (density, pressure, and self-generated current), which it has to send to the M3D-OMP code. M3D-OMP needs the plasma profile data to compute the new MHD equilibrium, and then has to send back to the XGC0 code this updated equilibrium.

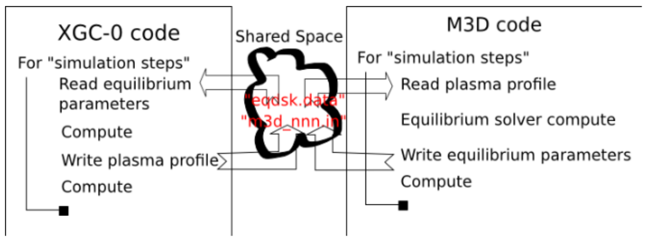

Figure 4. XGC0 - M3D coupling using the DataSpaces framework.

The implementation consists of a *client component* that is integrated with the two application codes and allows for dynamic data exchange at run time, and a *space component* that runs on a cloud of "staging nodes", which are dedicated nodes for the space and independent of the application nodes. The DataSpaces client in the XGC0 application inserts plasma profile data objects into the shared space, and the DataSpaces client in the M3D-OMP code extracts these data objects and passes them to the application. In the next step, the DataSpaces client in the M3D-OMP code inserts MHD equilibrium data objects back into the shared space, and the DataSpaces client in the XGC0 code extracts these data objects and passes them to the application.

Initial experiments with this coupling scenario ran XGC0 on 16 compute processors, M3D-OMP on 1 processor, and the DataSpaces space component on 4 processors. The amount of data exchanged between the two codes was relatively small, yet communication through the space was much faster than the approach based on file-exchange, which has been traditionally used for such coupling. We expect the benefits of using

DataSpaces to increase as the scale of the codes and the amount of data exchanged increase.

V. Workflows for Monitoring and Data Movement

In EFFIS, multiple codes are coupled running on different ORNL and other HPC resources, while they are constantly monitored. We need a framework that allows performing actions in parallel; *e.g.*, executing a parameter sweep of ELITE for stability checking while generating diagnostic plots from the XGC0 output. Since we want to monitor the XGC0 code status via diagnostic plots accessible from a dashboard, we need a finer grain resolution of task definition than a single job. Recent data from XGC0 should be transferred from the supercomputer to a satellite cluster, where the data is converted first and then plots are created from it, and this pipeline of tasks should be repeated as long as XGC0 is running.

We have chosen Kepler [13], a scientific workflow environment, as our technology for constructing code coupling applications. We have built Kepler workflows for orchestrating the execution of the codes, for transferring data among them, and for processing of their outputs. The coupling workflow is built as a *process network*, where each task is continuously running in a separate thread and is communicating with other tasks via predefined connections. Both task parallel and pipeline parallel processing are provided by this computational model, while certain packages allowed for remote SSH command execution and job-oriented operations. The Kepler framework and the coupling workflow are described in detail in [6].

The NCCS machines are protected by One-Time Password (OTP) authentication, so an established connection should be kept alive for the whole coupling session to avoid repeated authentications requiring a person to type in a new passcode each time. Since single jobs can run up to 24 hours on these systems and connections can live longer than this period, this does not prohibit us from monitoring simulations or performing the Full-ELM coupling scenario, which can last from a couple of hours up to a day. The SSH package we developed for Kepler to support access to OTP authenticated sites and the mechanisms for how we can run workflows as jobs that access such sites are described in [14]. Users are now, however, performing large-scale simulations divided up into multiple-day jobs (a couple of days for XGC1 but several months for astrophysics and combustion codes currently). For such long simulations, we cannot expect that the workflow stays connected to a supercomputer for the entire period; therefore, we need another way to get into these systems. Another new requirement is to launch workflows from the dashboard; that is, the user being connected through a web browser while the server side processes submitting the workflow that connects to the supercomputer in the name of the user.

NCCS has established an internal, grid certificate based authentication system by deploying GSI-SSH servers on the supercomputer *jaguar* as well as on the data processing cluster *ewok*, as shown in Figure 5. A modified MyProxy server, which authenticates the user with OTP instead of a user-defined passphrase, is also installed at NCCS. This allows one to establish an SSH connection to these machines using a grid certificate, but strictly only from some other NCCS resource and only with proxy certificates downloaded from the NCCS MyProxy server. This way, NCCS ensures that the user accessing the resource has been authenticated with OTP and the certificate is kept inside the safe environment of NCCS.

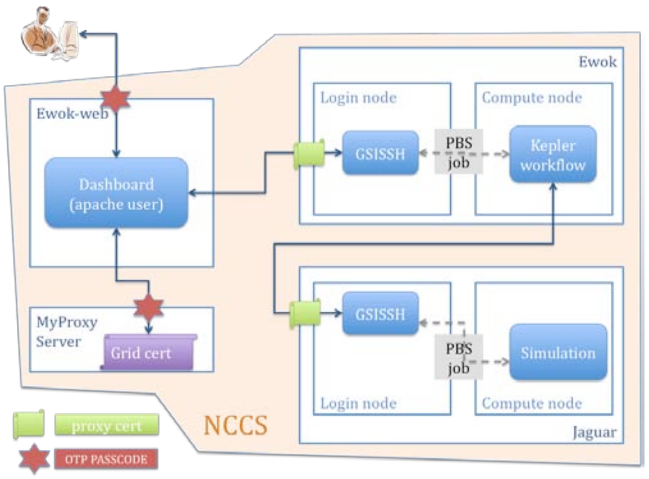

Figure 5. Grid certificate based authentication at ORNL.

The new (additional) authentication method enables both workflows running for an extended period of time to access data of a simulation and users to submit workflows and other jobs on NCCS resources from the dashboard. The dashboard can retrieve a proxy certificate from the MyProxy server when the user provides an OTP passcode, and then use the certificate to login to the processing cluster *ewok* in the name of the user and submit jobs; *e.g.*, a new coupling workflow or a Matlab analysis of some data from past coupling simulations. Similarly, a Kepler workflow can connect to the supercomputer *jaguar* repeatedly within the expiration period of the proxy certificate to submit a simulation or to access data from a running simulation.

VI. Dashboard

A. *Technologies and Key Features*

The eSimMon dashboard is a front-end tool for simulation monitoring that helps physicists manage, analyze, visualize and share data produced by their simulations. Adobe Flash is the technology of choice for the client side, while the server-side programming is a combination of PHP and MySQL queries.

The main objective when choosing this technology was to shift the focus of researchers away from the underlying IT details, such as file locations and formats, and onto the science itself. In other words, the ease of use is an invariable requisite in the eSimMon design. Other basic requirements include interactivity and responsiveness. There are several technology choices available for Rich Internet Applications [15] that create dynamic pages with local interaction and asynchronous communication with the server. A fundamental characteristic of most physical variables of simulation codes is their evolution through time. Videos allow scientists to monitor variables at different points in time. While playing movies is a requirement, it is also necessary to allow closer scrutiny of the data at each time step; hence, the need for vector graphics with zooming and panning capabilities. Flash has native support for videos and vector graphics. In addition Adobe has invested considerable effort in making their ubiquitous Flash Player fast, robust, scalable and reliable. Among the Web 2.0 technologies available, Flash provides all the desired features for the dashboard, including interactivity and cross-browser consistency but more importantly scalability and support for videos and vector graphics [16].

On the back end, the eSimMon dashboard uses PHP and a MySQL database to make the links between user requests on the interface and raw data files. The Kepler workflow records provenance information in the database, which the dashboard later queries to make the accurate connections. The provenance recorded includes the history of all data transformations and lineage of data products (data provenance), all operations executed by the workflow system (process provenance), and environment information combined with the source code of executed simulations and all actions of the users on the data (system provenance). The recording of provenance information and the algorithms to track the provenance for the dashboard use is described in detail in [17]. It is the key in enabling the dashboard to hide the details from the users and raise the focus from files to scientific variables.

The latest emphasis in the dashboard development is on data analysis; notably, analysis using vector graphics. Users have the option of looking at x-y plots in a resizable vector graphics window. They have zooming and panning capabilities as well as formatting preferences for plotting variables (see Figure 6). Scientists also have the ability to overlay two variables on one vector graphics cell. The ultimate purpose of the vector graphics is to allow scientists to draw and redraw plots until they are satisfied with their results, then click on a Publish button to request a publication quality image or movie to be generated on the back end.

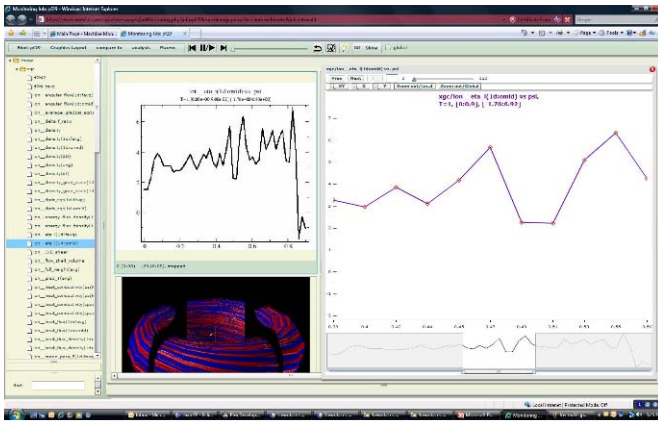

Figure 6. Dashboard New Feature: Vector Graphics.

Another example of analysis on the dashboard is the eSimMon calculator. This calculator tool allows users to perform basic math operations on the simulation variables (such as generating the difference between two graphs) and view the results on the fly. The eSimMon calculator was originally implemented for XGC1 variables. It works with x-y plots generated from NetCDF data files and can be extended to other types of x-y plots. To take the analysis one step further, users will soon be able to upload their own analysis routines and run them from the dashboard.

The same general concept is valid throughout the dashboard and carries on into the analysis. Users do not need to know where the appropriate raw data files are; neither do they need to know which software is used on the back end or how to use it. By simply clicking on movies and calculator operators, they compose an expression which the back end later interprets and executes.

B. Wide Area Data Movement

Using the dashboard to monitor simulations involves visualizing different image products that were extracted from code diagnostic output or checkpoint files. The user viewing an image may get an unexpected result, and therefore wish to download the original file or files from which the image was produced to his/her own site for further investigation. In order to support such functionality on the dashboard, three tasks need to be performed: (i) identify the original files from which the image was produced by the workflow system by their Logical File Names (LFNs); (ii) identify the current physical location where the files reside, referred to as Physical File Names (PFNs); and (iii) invoke a file movement tool to move the files from the physical locations to the destination site. All of these tasks and their details need to be invisible to the user; all the user needs to provide is the information about his/her destination site, and the rest should be taken care of automatically. The first two tasks can be accomplished by extracting the desired LFNs and PFNs from information provided by provenance recording. Since such information is stored in a database, these tasks require simple queries to the database. The third task is more complicated and is described below.

There are several requirements for accomplishing the data movement task. First, the data mover tool needs to accept large *multi-file requests* (10s-100s of files or more) and schedule the file transfers. This implies that the data movement happens asynchronously – that is, the user does not have to be logged into the dashboard for this action to take place. The user should be able to log onto the dashboard at a later time and find out the status of the file movement request. Second, the file movement must be *robust*. This implies that all file transfers must be monitored, and if failures occur, re-transmission must take place automatically to recover from such failures. Third, the data mover tool needs to take advantage of the available bandwidth by transferring multiple files *concurrently*. Fourth, the data mover tool should be able to support multiple transfer protocols in order to match the destination transfer servers. Fifth, it should be possible to invoke this tool on behalf of the user directly on the dashboard, acting as a “client program” behind a firewall and not as a general data mover service.

A tool meeting these requirements named the DataMover has been fully implemented and tested with large multi-file transfers, and it supports multiple transfer protocols (*e.g.*, SCP, SFTP, FTP, GridFTP, HTTPS). The implementation is based on a client version of the Storage Resource Manager [18] server implementation. Operation of the DataMover is shown schematically in Figure 7. It uses an XML file for the description of what needs to be transferred, uses SSH to establish connection to remote sites, and uses multi-threading for concurrent transfers. In one representative test of the connection between LBNL and ORNL (which potentially supports a 1 Gbps transfer rate), a rate of 93MB/s (0.744 Gbps) was observed using concurrent transfers.

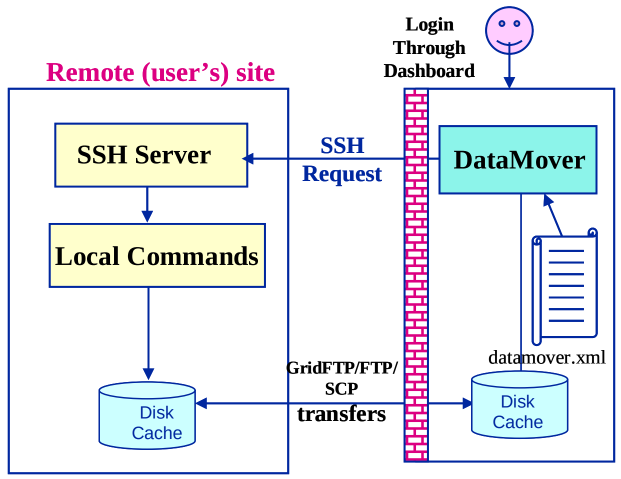

Figure 7. The DataMover is invoked through the dashboard and manages the transfer of files to the user's remote site.

VII. Conclusions

To address the many challenges of producing an integrated edge plasma simulation model, the Center for Plasma Edge Simulation has developed EFFIS, an End-to-end Framework for Fusion Integrated Simulation. EFFIS consists of a set of key computer science technologies, including the Adaptive I/O System (ADIOS), Kepler scientific workflows, and the eSimMon dashboard, which have been developed in tandem and customized to closely fit the needs of fusion simulation scientists. The application of these EFFIS tools has enabled CPES researchers to explore complex code coupling scenarios and to more easily launch, monitor, manage and analyze their plasma simulations on leadership class computing platforms

Acknowledgments

This work is part of the ongoing research activities within the Center for Plasma Edge Simulation, a SciDAC Fusion Simulation Prototype center that is supported by the Office of Fusion Energy Sciences and the Office of Advanced Scientific Computing Research within the US Department of Energy. We are grateful to the National Center for Computational Science at Oak Ridge National Laboratory and the National Energy Research Scientific Computing Center at Lawrence Berkeley National Laboratory for access to and support of their computing resources. We would especially like to thank NCCS system administrators Sergey Shpanskiy and Josh Lothian for their ongoing support in installing and maintaining the services we need for our framework.

References

[1] P. B. Snyder et al., "Edge localized modes and the pedestal: A model based on coupled peeling-ballooning modes", Phys. Plasmas, vol. 9, p. 2037, May 2002.

[2] US DOE Office of Science, INCITE Leadership Computing, *http://www.er.doe.gov/ascr/incite*, May 2009.

[3] C. S. Chang, S. Ku and H. Weitzner, "Numerical study of neoclassical plasma pedestal in a tokamak geometry", Phys. Plasmas, vol. 11, p. 2649, May 2004.

[4] H. R. Wilson, P. B. Snyder, G. T. A. Huysmans and R. L. Miller, "Numerical studies of edge localized instabilities in tokamaks", Phys. Plasmas, vol. 9, p. 1277, April 2002.

[5] W. Park, E. V. Belova, G. Y. Fu, X. Z. Tang, H. R. Strauss and L. E. Sugiyama, "Plasma simulation studies using multilevel physics models", Phys. Plasmas, vol. 6, p. 1796, May 1999.

[6] J. Cummings et al., "Plasma Edge Kinetic-MHD Modeling in Tokamaks Using Kepler Workflow for Code Coupling, Data Management and Visualization", Commun. Comput. Phys., vol. 4(3), pp. 675-702, September 2008.

[7] S. Klasky et al., "Adaptive IO System", Proceedings of the 50th Cray User Group meeting (CUG 2008), Helsinki, Finland, May 2008.

[8] J. Lofstead, S. Klasky, M. Booth, H. Abbasi, F. Zheng, M. Wolf and K. Schwan, "Petascale IO using the Adaptable IO System", Proceedings of the 51st Cray User Group meeting (CUG 2009), Atlanta, GA, May 2009.

[9] J. Lofstead, F. Zheng, S. Klasky and K. Schwan, "Adaptable, Metadata Rich IO Methods for Portable High Performance IO", Proceedings of the 23rd IEEE International Parallel & Distributed Processing Symposium (IPDPS 2009), Rome, Italy, May 2009.

[10] H. Abbasi, J. Lofstead, F. Zheng, S. Klasky, K. Schwan and M. Wolf, "Extending I/O through High Performance Data Services", to appear at Cluster Computing 2009, New Orleans, LA, August 2009.

[11] J. Lofstead, S. Klasky, K. Schwan, N. Podhorszki and C. Jin, "Flexible IO and Integration for Scientific Codes through the Adaptable IO System (ADIOS)", Proceedings of the 6th ACM/IEEE International Workshop on Challenges of Large Applications in Distributed Environments (CLADE 2008), Boston, MA, June 2008.

[12] C. Docan, M. Parashar and S. Klasky, "DART: A Substrate for High Speed Asynchronous Data IO", Proceedings of the 17th IEEE Symposium on High Performance Distributed Computing (HPDC 2008), Boston, MA, June 2008.

[13] B. Ludaescher, 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, vol. 18(10), p. 1039, August 2006.

[14] N. Podhorszki and S. Klasky,"Workflows in a secure environment", Proceedings of the 7th International Conference on Distributed and Parallel Systems (DAPSYS 2008), Debrecen, Hungary, September 2008.

[15] J. Allaire, "Macromedia Flash MX -- A next-generation rich client", *http://download.macromedia.com/pub/flash/whitepapers/richclient.pdf*, March 2002.

[16] R. Barreto, S. Klasky, N. Podhorszki, P. Mouallem and M. Vouk, "Collaboration Portal for Petascale Simulations", Proceedings of the 2009 International Symposium on Collaborative Technologies and Systems (CTS 2009), Baltimore, MD, May 2009.

[17] P. Mouallem, R. Barreto, S. Klasky, N. Podhorszki and M. Vouk, "Tracking Files in the Kepler Provenance Framework", Proceedings of the 21st International Conference on Scientific and Statistical Database Management (SSDBM 2009), New Orleans, LA, June 2009 [Lecture Notes in Computer Science, vol. 5566, Springer, 2009, pp. 273-282].

[18] A. Shoshani, A. Sim and J. Gu, "Storage Resource Managers: Essential Components for the Grid", in Grid Resource Management: State of the Art and Future Trends, J. Nabrzyski, J. M. Schopf and J. Weglarz, Eds., Kluwer Academic Publishers, 2003.