Institute of Physics Publishing Journal of Physics: Conference Series **16** (2005) 510–520
doi:10.1088/1742-6596/16/1/070 SciDAC 2005

Data management on the fusion computational pipeline

S. Klasky[1], M. Beck[3], V. Bhat[1, 2], E. Feibush[1], B. Ludäscher[5], M. Parashar[2], A. Shoshani[4], D. Silver[2], M. Vouk[6]

[1]*Plasma Physics Laboratory, Princeton University, NJ {sklasky,efeibush,vbhat}@pppl.gov*
[2]*Electrical and Comp Engr. Rutgers University, NJ parashar@caip.rutgers.edu*
[3] *Computer Science Dept, University of Tennessee, TN {atchley, mbeck}@cs.utk.edu*
[4]*Lawrence Berkeley Laboratory, Berkeley, CA shoshani@lbl.gov*
[5]*Computer Science, U.C. Davis, CA ludaesch@ucdavis.edu*
[6]*Computer Science, N.C. State, NC vouk@ncsu.edu*

Abstract. Fusion energy science, like other science areas in DOE, is becoming increasingly data intensive and network distributed. We discuss data management techniques that are essential for scientists making discoveries from their simulations and experiments, with special focus on the techniques and support that Fusion Simulation Project (FSP) scientists may need. However, the discussion applies to a broader audience since most of the fusion SciDAC's, and FSP proposals include a strong data management component. Simulations on ultra scale computing platforms imply an ability to efficiently integrate and network heterogeneous components (computational, storage, networks, codes, etc.), and to move large amounts of data over large distances. We discuss the workflow categories needed to support such research as well as the automation and other aspects that can allow an FSP scientist to focus on the science and spend less time tending information technology.

1. Introduction

From March 2004 – June 2004, the DOE held workshops, headed by Richard Mount, to discuss the ASCR/MICS strategic plan for data management [18]. By the end of the workshop it was clear that the data management requirements for several application domains were similar in at least three application areas: simulations (such as fusion and combustion and astrophysics), observation/experimental-driven applications (high-energy physics, fusion), and information-intensive applications (such as chemistry, biology, and nanoscience). This paper focuses on several data management issues that are necessary for our funded Fusion Simulation Project (FSP).

It is important to note that the Office of Energy Science (OFES) has a strong data management program for current experiments. However, when this data management system was applied to a "real 3D " simulation, it proved to be inadequate, not just in its speed, but also in its inflexibility to handle the needs of simulation scientists. The time is ripe for OFES to join in collaborative efforts with other DOE data management researchers and design a system, which will be scalable to a FSP and ultimately to the needs of ITER.

Simulations are typically executed in batch mode, they are long running, and the computational resources they use are located at just a few supercomputing centers. To accelerate the discovery process for simulations, a new generation of comprehensive data management solutions will be required, which span all areas of data management and visualization.

The Center for Plasma Edge Simulation is going to build a new integrated predictive plasma edge simulation framework. This framework will be applicable to existing magnetic fusion facilities and to next-generation burning plasma experiments such as ITER. The multi-scale nature of this problem occurs partly because the microturbulence and neoclassical physics time scale must be studied kinetically, using the XGC-ET code, while the faster and larger scale MHD modes are more efficiently studied using a fluid code, M3D [33]. These codes will be loosely coupled[1], and therefore we must

[1] Coupling can be defined in many ways. It can be synchronous and asynchronous. It also spans several orders of magnitude. For the purposes of this paper we can think of several categories a) microsecond coupling (e.g., tight computational coupling on shared memory and high-performance clusters), b) millisecond couplings (e.g., among internet distributed clients and computational nodes), c) seconds to minutes (synchronous and asynchronous analyses involving high end applications and human interactions), and d) hours to days (e.g., large volume of backup data or asynchronous interactions)

focus on large data sets generated from these codes, and the transfer of this data to the collaborators in this project. This project is ambitious in both the physics and the enabling sciences. Over twenty scientists will be working on different aspects, and good data management techniques will be essential.

The key feature of the data management techniques that we will use in the FSP is that they will be driven by the demands of the applications. It is crucial to devise a data management system that is easy for the physicists to use and easy to incorporate into their codes and workflows. In order for this project to be successful, scientific discovery must be accelerated by using leadership class computational solutions and state-of-the-art enabling technologies, e.g., those that reduce overhead of the information technology and provide automation of workflows. There are two major steps in our discovery system:

1. During the simulation stage where information must be immediately (often synchronously) processed and displayed to enable the user to control the simulation.
2. During the *Validation and Verification* stage where users will compare the data from this simulation to other simulations and/or experimental data.

Below we describe the scientific investigation process commonly used by fusion simulation scientists to do their science. We then describe the core data management technologies that will be used in our Fusion Simulation Process. Finally, we will describe the challenges faced in this project.

2. The Scientific Investigation Process

A fundamental goal of a simulation scientist is to analyze and understand information generated from a combination of simulation codes and empirical data, and have these processes lead to an increased understanding of the problem being simulated. One can describe the scientific investigation process in terms of seven conceptual stages. These same stages can be identified across a wide range of disciplines and provide a useful framework for identifying data management challenges [18].

Formulation of a hypothesis for explanation of this phenomenon (*Idea Stage*) leads to formulation of requirements for testing it in the *Implementation Stage*. Regression tests are developed to ensure that the modifications do not violate the prior developments. Changes need to be tracked and captured in metadata for accountability, backtracking and reproducibility. Implementation often intermixes with the *Validation/Verification (V&V) Stage*. V&V requires scientists to analyze and interpret results, e.g., through data transformations, data mining and visualization. This introduces the *Interpretation Stage*. During the *Pre-production Stage* scientists run parameter surveys and/or sensitivity analyses to define the regime of interest and/or define correct scaling. This stage is intermixed with the Interpretation Stage. These two stages combined often provide some insight into whether the hypothesis under investigation makes sense. Accumulation of bulk raw data happens in the *Production Stage*, when scientists run production experiments and simulations and perform massive observations. The data acquired during this phase is generally large and growing. Production and interpretation are intermixed as part of the original hypothesis testing. The *Assimilation Stage* is the final step of the scientific process. Results from all of previous steps are assimilated and reformulation of the original hypothesis may be needed. The final output is the dissemination of the knowledge through peer-reviewed papers, presentations and, increasingly, publication of some portion of the data itself.

The process can be captured in one or more *workflows*. We believe that a significant fraction of the scientists' time is spent on interpretation of their data through data analysis and visualization. As the data grows, some steps may become prohibitively slow, especially if there is a technology overhead. The challenge is to develop appropriate analysis and visualization tools, and automate data acquisition and manipulation workflows, that increase total scientific productivity.

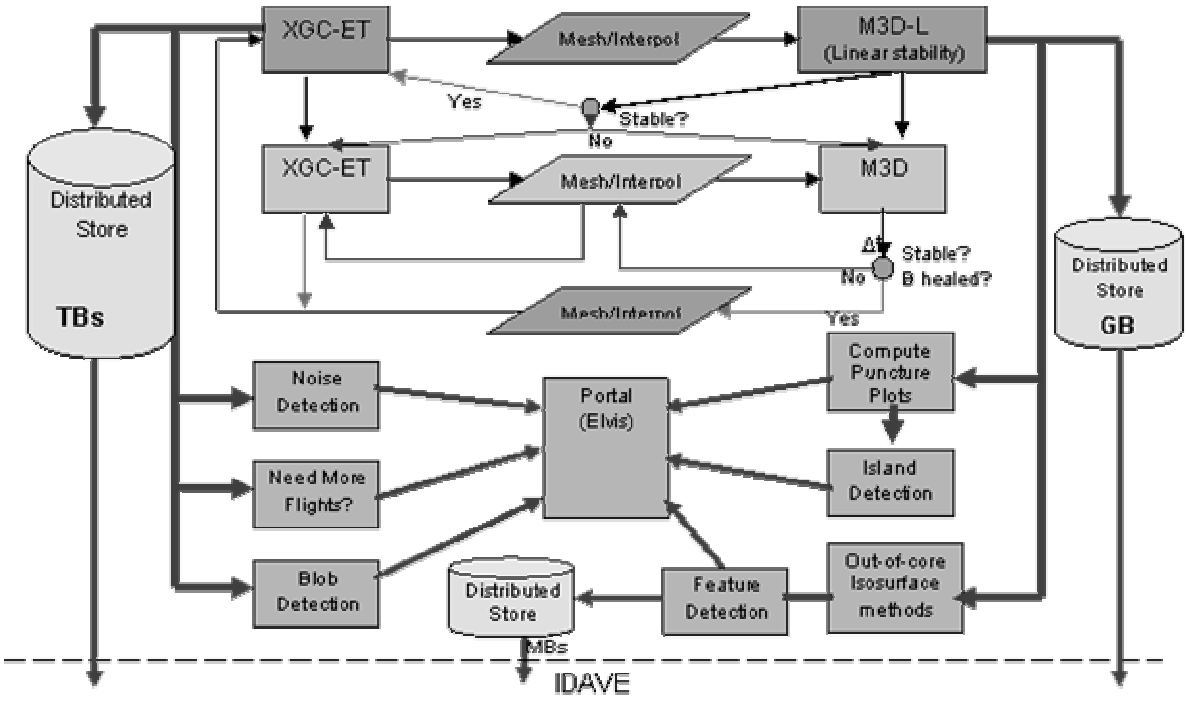


Figure 1. Illustration of FSP workflow

3. Adaptive and Automated Workflows

The FSP is a sufficiently complex problem, that classic composition and manual execution of its workflows represents a serious overhead. Advanced workflow automation is needed. One of the proposed FSP workflows is illustrated in Figure 1. The top part of the diagram shows interactions among the simulation components running the XGC-ET and the M3D codes. These codes generate large volumes of data that are archived by the workflow agents. The bottom part shows analysis components that are invoked automatically by the workflow engine at specific time steps, and that feed into the Integrated Data Analysis and Visualization Environment (IDAVE) portal. Summaries of the monitored information and the dynamic analyses are also sent to a portal. This information is used by the scientists to track workflows progress over the Internet. We now discuss the requirements for the FSP workflow support framework.

3.1. Workflow Automation.

Coupling scientific simulations in the manner envisioned in FSP requires coordination of several simulations, running on different distributed computing resources, and includes reliable movement of data between tightly-coupled components, but also over wide area networks. Also needed is the orchestration of intermediate analysis tasks and the automation of dynamic steering of the simulation. These requirements warrant a *scientific workflow system* that can setup, run, control, monitor, and track this process.

Scientific workflows have unique requirements distinct from business workflows. Perhaps the most relevant for a FSP is the requirement to run tasks based on some step granularity, and to stream the data generated by these steps between components. This is essential for running time-stepped simulations. Other important requirements include triggering an action every so many steps (such as "checkpoints" that need to be transferred to another computer, archived, or analyzed), and triggering an alarm or automatically adjusting the parameters of a simulation based on dynamic analysis of data generated by time steps. It is also essential that the workflow system is flexible enough to allow the plug-in of existing components, such as simulation programs written by scientists, or specialized analysis components, in a standardized fashion. Finally, because real workflows can be quite complex it is essential that a scientific workflow system supports workflows that can be composed from other

(sub-) workflows. Since many scientific workflows run for a long time, another important requirement is to monitor and track the progress of the workflow, and to produce alarms if the progress is stalled or flawed.

There are a number of general requirements for scientific workflows, [e.g., [26][27][24][23][22][19][20]. Some principal scientific workflow requirements include:

- **User Interface:** Intuitive design, execution, and monitoring environment for end user
- **Re-use:** Component reusability and exchangeability; in particular, ability to dynamically add new process nodes and data sets to a workflow (e.g., using "plug-ins" for Web services and data), or change existing ones
- **Data transforms:** Extensive and flexible data format transformation support to mediate between consecutive processing steps and between the data sources and data sinks (a.k.a. workflow "shims" or adapters)
- **Interaction and Batch Modes:** Support for user interaction (including process steering) during process execution. Ability to monitor an automated processes in real-time, to "background" it and retrieve results later, stop/resume options with long-term state memory, etc.
- **Streaming:** Support for pipelined concurrent execution of fine & course-grained (loosely coupled) workflows
- **Location transparency**: Support for both local and distributed computation and data access
- **Interoperability:** Ability to incorporate ("wrap") external applications, e.g. for data access, analysis, visualization, and to interoperate with other component-oriented frameworks and systems.
- **Workflow complexity:** Ability to handle nested workflows, different computation models, data/control flow
- **Optimization:** Ability to collect cost and performance data, and based on these, predict cost and performance of workflows execution plans. Plan efficiency depends on host resources, bandwidth, data transport mechanisms (e.g., SABUL, FastTCP, SRB, etc.) chosen for a particular workflow connection, etc.
- **Dependability and Scalability:** Workflow engine and resources need to be reliable and scalable; so mechanisms for fault-tolerance, recoverability, and parallel execution of workflows are needed
- **Verification and Validation:** Ability to verify and validate constructed workflows, imported workflows, and results obtained through an automated workflow

There are several scientific workflow support systems today, many of them evolving [22][28][32][31]. The Scientific Data Management (SDM) Center has identified a framework called Ptolemy [9] that, with appropriate additions SDM and collaborating projects are developing, provides a robust and well suited environment for scientific workflows. In the resulting Kepler system [27], workflow components (legacy or new) are plugged in as "actors". Simple actor pipelines and complex workflows (with nested subworkflows and loops) can be graphically defined. Workflow execution is orchestrated by one or more "directors", defining appropriate execution models (process network, synchronous dataflow, discrete event, etc.) Specific actors have been developed for repetitive execution of steps, command-line initiation of tasks, file staging and file caching, large scale data movement, and notification. Notifications are triggered by conditions in the tasks, and perform an action, such posting a message to a website, sending an email, etc. The Kepler system has been successfully applied by the SDM Center staff to several applications in the fields of astrophysics, fusion, and computational biology and chemistry [BGS+05], and to ecoinformatics and geoinformatics applications by other projects contributing to Kepler [29][30].

For FSP workflows, we will extend Kepler, e.g. with actors for moving data specifically from large machines such as Seaborg (the IBM SP at NERSC) running XGC code to the cluster running the M3D code. While large data movement elements already exist as part of the Kepler/SPA library [26], FSP synchronization needs may require some customization. Similarly, FSP workflows require data archival to mass storage (e.g. HPSS). We may take advantage of the Storage Resource Management (SRM) technology available at the SDM center or reuse existing Kepler/SRB actors. Actors can also

be used to invoke software that analyzes intermediate results, e.g. to classify puncture plots or to invoke a monitoring and/or visualization component and display the results on a target system.

3.2. Interactive and Autonomic Control of Workflows and Simulations

The scale, complexity and dynamic nature of the FSP coupled with similar scale and complexity of emerging parallel/distributed execution environments requires that these applications be accessed, monitored and controlled during execution. This is necessary to ensure correct and efficient execution of the simulations, as the choice of algorithms and parameters often depends on the current state/requirements of the application and the state of the system, which are not known *a priori*. For example, simulation component behaviors, their compositions and the simulation workflows can no longer be statically defined. Further, their performance characteristics can no longer be derived from a small synthetic run, as they depend on the state of the simulations and the underlying system. Algorithms that worked well at the beginning of the simulation may become suboptimal as the solution deviates from the space the algorithm was optimized for or as the execution context changes. This requirement presents a new set of deployment and runtime management challenges.

We are investigating programming and runtime management solutions to support the development and deployment of applications (i.e., simulations elements, their interactions and the workflows) that can be externally monitored and interactively or autonomically controlled. Further, we are also investigating programming and runtime systems that can support efficient and scalable implementations of these autonomic simulations. This includes designing control networks to allow computational elements to be accessed and managed externally, both interactively and using automated policies, to support runtime monitoring, dynamic data injection and simulation control. This research is built on our current and prior research efforts and software projects including AutoMate/Accord[11][12], and Discover/DIOS [13].

3.3. Collaborative Runtime Monitoring with Elvis

Scientists desire to monitor a long running simulation in case they want to stop it before completion. Stopping an errant run conserves compute and human time. It is useful to monitor at any time and from several locations. Providing the information over the Internet facilitates frequent monitoring. A scientist can check a run from an Internet browser at work, home, or while traveling. Monitoring is typically performed on a laptop or desktop computer on an office or home network. A simulation can produce more data than can be monitored. In the FSP, the XGC code will compute profile data that is input to the M3D code. The data is produced incrementally as the XGC runs and represents the area and variables of interest. A subset of the total output is stored in a portal for monitoring.

Several scientists at different locations will be interested in an Edge Dynamics run. Monitoring over the Internet makes the data instantly available to multiple locations. This enables a basic level of data sharing among users. The Elvis system [4], based on the Scivis system [7], provides more advanced collaboration by implementing a collaborative whiteboard so users can interactively annotate, highlight, and graphically explore the shared data. This improves collaboration and communication between remote users. Development has started to make Elvis available for Kepler-controlled workflows.

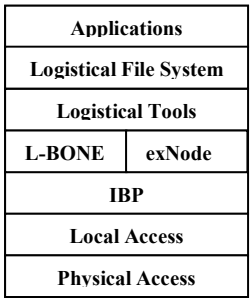
4. Efficient Data Access /Movement

The key to the success of any workflow scheme is the ability to move data, so we will focus on data access/data transfer issues in some detail. The efficiency of data access is greatly affected by two factors: 1) the network path between the point of access (a storage resource or buffer) and the system to or from which the data is read or written, and 2) the ability to move data when it is available, rather than only at certain predefined points in a computation (most commonly at the beginning and end). We will describe two mechanisms which address these issues: 1) Logistical Networking, which provides a uniform and ubiquitous model of storage managed on servers known as “depots” that are located throughout the network, and 2) Data Streaming techniques that overlap data access and transfer with computation and that move data directly from the application to storage locations close to its ultimate destination [8].

4.1. Logistical Networking

To achieve the kind of global deployment scalability some high-end applications require for data management, *Logistical Networking* (*LN*) uses a highly generic, best effort storage service, called the Internet Backplane Protocol (IBP). The design of IBP is shaped by analogy with the design of IP in order to produce a common storage service with similar characteristics. Though it has been implemented as an overlay on TCP/IP, it represents the foundational layer of the "network storage stack" [1]. Just as IP datagram service is a more abstract service based on link-layer packet delivery, so is IBP a more abstract service based on blocks of data (on disk, memory, tape or other media) that are managed as "byte arrays." By masking the details of the local disk storage — fixed block size, different failure modes, local addressing schemes — this byte array abstraction allows a uniform IBP model to be applied to storage resources generally. The use of IP networking to access IBP storage resources creates a globally accessible storage service.

As the case of IP shows, however, in order to scale globally the service guarantees that IBP offers must be weakened, i.e. it must present a "best effort" storage service. First and foremost, this means that, by default, IBP storage allocations are time limited. When the lease on an IBP allocation expires, the storage resource can be reused and all data structures associated with it can be deleted. Additionally an IBP allocation can be refused by a storage resource in response to over-allocation, much as routers can drop packets; such "admission decisions" can be based on both size and duration. Forcing time limits puts transience into storage allocation, giving it some of the fluidity of datagram delivery. More importantly, it makes network storage far more sharable, and therefore easier to scale up. The semantics of IBP storage allocation also assume that an IBP storage resource can be transiently unavailable. Since the user of remote storage resources depends on so many uncontrolled, remote variables, it may be necessary to assume that storage can be permanently lost. In all cases such weak semantics mean that the level of service must be characterized statistically.



4.2. PPPL Data Streaming

Interactive data analysis/remote visualization has been studied over a long period of time. Wide area networks, however, exhibit high latencies and widely varying throughput, which hampers remote analysis and visualization as well as overall scientific throughput. These latencies and varying throughput make interactive visualization impractical because reading and displaying the data can take tens of seconds to several minutes for every frame. Also, for high performance simulation components, analysis and visualization routines normally require a lower order of processors (M) compared to the actual simulation runs (N where M<< N) further arguing for moving data to different resources for post-processing. To avoid loss of raw or processed data, the data should be transferred fault tolerantly. Thus, the goal of the data streaming is to transfer data from a live simulation running in batch on a remote supercomputer (at NERSC/ORNL) over the Wide Area Network (WAN) to a local analysis/visualization cluster and with replication for fault-tolerance [3]. To transfer blocks of data which are buffered by storing and managing it in buffers allocated at the application layer we use the Logistical Backbone as a scalable infrastructure for our application specific Data Grid [2][3].

4.2.1. Design of the Threaded Buffer for Streaming Data on multiple processors.

Since large scale simulations execute on a large number of processors (N) which is much greater than the data-receiving processors (M) it is advantageous to form groups of processors which group data from the simulation into a single entity, and send this data to receiving processors over the WAN to PPPL. To handle this mismatch we designate certain data-generating processors as I/O processors and failsafe processors which have the capability to allocate buffers and transfer data to the receiving processors. These processors are normal processors which take part in the simulation process but they are allocated an extra task of transferring the generated data. A processor can also be both an I/O processor and failsafe processor. Every processor in the simulation belongs to a particular group known as TWRITE group which consists of an I/O processor which allocates an I/O buffer, a failsafe

processor which allocates a failsafe buffer and normal simulation processors. Normal simulation processors have knowledge of the I/O processor in their group. An I/O processor collects data from each of its group members (including the failsafe processor and itself) and copied it into its buffer.

4.2.2. Adaptive Buffer and Data Management

Simulations are based on a series of time steps in which the data is generated after finite computation. Data which is generated by the simulation is copied to a buffer allocated in the simulation. The buffer has a hard limit/physical limit attached to it. Buffers allocated in the simulation wrap around once they reach the hard limit. The data generation rate of a simulation is the amount of data generated per step, divided by the time required to perform the step. The network connectivity between the simulation processors and the analysis processors places an upper limit on the transfer throughput. The algorithm for data management in the buffer tries to dynamically adjust to the data generation rate and the available network rate. It does this by sending all the data that has accumulated since the start of the last data transfer. If the data generation rate exceeds the transfer rate, more data will be in the buffer. In this case, the queue manager will increase the amount of multi-threading in the transfer routines to improve throughput. If the transfer rate exceeds the data generation rate, then less data will appear in the buffer for the next transfer. All subsequent transfers start as soon as the prior transfer ends. After some number of time-steps, if the network is stable and the data generation rate is less than the network transfer capacity, then the buffer manager tends to reach equilibrium and match the transfer rate to the data generation rate.

4.2.3. Failsafe Adaptive Buffer Management

Failsafe buffer management for buffer overflows when buffers are located on different processors is shown in Figure 3. This figure illustrates a simulation which is running on 48 processors; the processors are divided into 3 groups which send data out, known as TWRITE groups. Each of these groups has an I/O processor and Failsafe processor. The I/O and Failsafe processors are responsible for transferring the data in addition to participating in the simulation. The I/O processor has a buffer which it uses to transfer data to remote depots at PPPL. The Failsafe processor for every group has a buffer which it uses to transfer data to the local depots close to the simulation or is on a local area network where the data is being generated. The philosophy for the failsafe buffer is that it data will be transferred quickly to local depots compared to remote depots. The failsafe buffers are normally smaller in size compared to the I/O buffers.

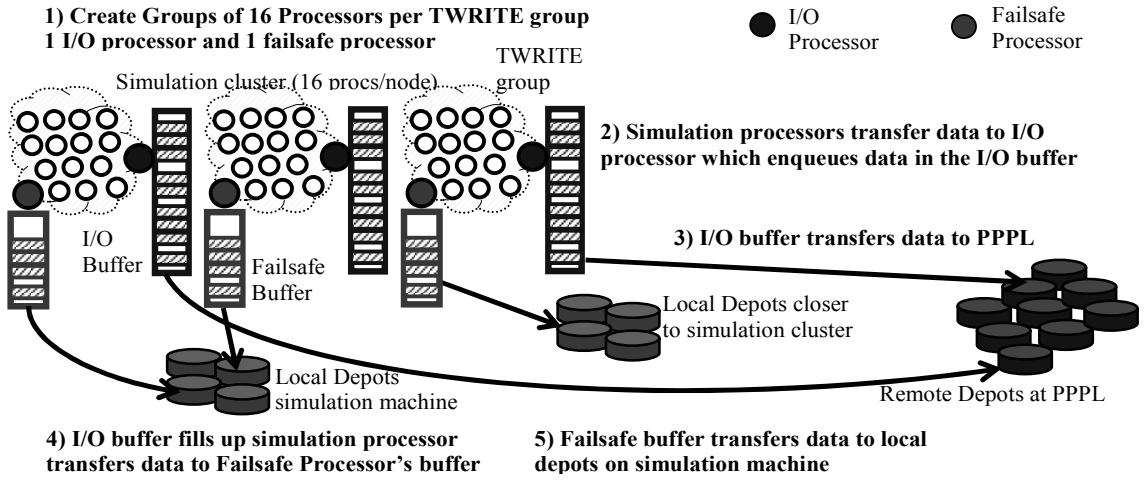


Figure 3: Data Management using I/O buffer and Failsafe buffer on different processors.

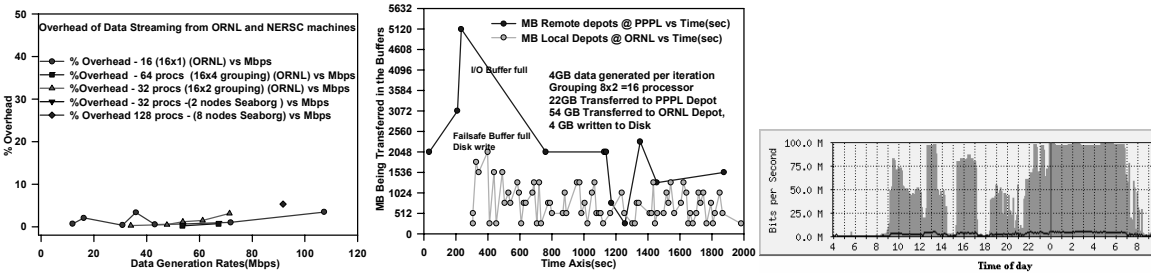


Figure 4a (left): Overhead of creating groups, 4b (middle): Failsafe Mechanisms, 4c (right): Router statistics at PPPL while streaming data from an 8 hour simulation at ORNL.

4.2.4. Results

Figure 4a shows the overhead involved in creating groups of processors when running the simulation with various processor sizes at ORNL and at NERSC for various Data generation rates. The data streaming approach with grouping imposes minimal overhead on the simulation and is below the 10% limit. Figure 4b shows the failsafe mechanisms when streaming data at 350Mbps. The maximum network transfer rate from PPPL to ORNL is about 100Mbps, the excess data will have to be either written to depots at ORNL located on a local area network, or the local disk. This is observed from the graph during the transfer of 4GB of data and as talked about in the previous section. We observe from the Figure 4c that we are able to utilize 99% of the bandwidth during the eight hour simulation.

5. Integrated Data Analysis and Visualization Environment (IDAVE)

There are three main areas where visualization arises in our FSP. The first involves real-time visualization of profile data produced from the real-time monitoring of the simulations. This data will be compact and will be visualized through the runtime monitoring portal. The second involves post processing visualization routines. Both the Kinetic-Edge and MHD codes produce periodic data dumps that need to be analyzed and visualized. This type of visualization can be done either at the scientist's local workstation or at a more advanced, visualization server, which may have distributed processing power, display wall, etc. We will be investigating routines that allow juxtaposition of the different datasets produced by the simulations and which will enable comparisons between simulation and experimental data. An important component of the visualization will be data mining algorithms such as feature detection and activity recognition. The third area involves incorporating automated visualization routines within the scientific workflow. This allows the data to be preprocessed and enables a more efficient first look at the data as it is being produced. In all of the above categories, the proposal will encompass both "nuts and bolts" visualization routines and research into novel visualization techniques. Our goal is to provide usable and reliable visualization solutions to the physicists, in addition to exploring new methodologies.

Our goal for data analysis and visualization is to enhance the existing Integrated Data Analysis and Visualization Environment (IDAVE) in the fusion community to support robust and accessible visualization, to incorporate and tightly integrate visualization into the scientific workflow, and to support advanced visualization/ data mining capabilities on the simulation and experimental data produced.

One area of interest is to analyze the scientific workflow and incorporate visualization algorithms that can be split between the workflow automation and the IDAVE. For example, a large dataset can be preprocessed (as it is output from the simulation and before it is saved to disk) for out-of-core isosurfacing [14]. The data can then be accessed from the IDAVE for fast isosurface processing. Many other visualization techniques can also be predefined and automatically implemented as part of the scientific workflow. These reduced representations can be displayed as part of the profiling and as a quick look at the data before a full interactive investigation of the data within the IDAVE. We will investigate incorporating visualization and analysis within the workflow and identify those algorithms which are suited for this process.

6. Ubiquitous and transparent data sharing

The scaling of simulations to ever finer granularity and timesteps brings new challenges in managing and accessing the data generated by the simulations. First, the volume of the full gyrokinetic edge code dataset is expected to reach many terabytes and even petabytes. This requires the data to be stored on multiple storage systems, as well as on mass storage systems such as HPSS. There are significant data storage and movement problems associated with storing replicas at locations where they are likely to be accessed and using highly distributed resources managed by community members in order to make the data available on demand [16].

Second, accessing a small subset of the data from large datasets, such as requesting a hyper-slab (about 1 GB) out of a complete electromagnetic field dataset from the XGC-ET code, should be made simple so that scientists can get such data on-demand even from their laptops. A high degree of flexibility is required in the mechanisms used for localizing data for visualization and post-processing, including application-specific control over dynamic caching. Third, post-processing entire datasets repeatedly for analysis and visualization is prohibitive. For example, velocity moments of the plasma distribution function such as density and temperature are needed in the post-processing phase and should be derived dynamically as the simulation data is generated. Because post-processing is a collaborative effort carried out within the distributed research community, the results of analyses carried out at a specific site must be redistributed globally for the benefit of the entire community [17]. Furthermore, the post-processing and initial visualization can be incorporated into the scientific workflow. Lastly, there is a need to keep track of the lineage and semantic information about the datasets generated, which is referred to as metadata. The metadata requires a simple but powerful data model in order for specific queries to find the desired parts of the datasets. The metadata can also include feature-based information for more advanced data mining.

Our goal is to support transparent data access by combining semantic models with the Logistic Networking framework and incorporating visualization and post-processing within the scientific workflow. To support transparent data access, it is necessary to present the user with a "logical name space" for datasets and files that belong to the datasets. To support ubiquitous data access, it is necessary to permit multiple replicas of logical files to exist in the system, so that the most used files (so called "hot" files) are replicated in storage systems that are more readily available to the users. Such an infrastructure, once in place, will permit not only on-demand access [16], but can also use intelligent data placement technology that dynamically manages the replication of files based on usage patterns. We do not propose to develop the above technology from scratch. Fortunately, there is a body of knowledge on managing large volumes of replicated data in other scientific domains (e.g., High Energy Physics and Earth Sciences), where middleware components have been developed to support such data management tasks. The main components that we find useful are:

(1) Metadata catalogs –that allow the description of datasets according to their properties, and upon a query based on these properties return the set of logical file names.

(2) Replica catalogs – that maintain the mapping between logical file names and physical file names. The replica catalog keeps the one-to-many mapping and provides indexes for fast search.

(3) Storage Resource Managers (SRMs) [5] that provide a uniform interface to different storage systems, including disk systems or mass storage systems such as HPSS. SRMs have been used for large scale robust file replication in production [6].

An important advance that will be undertaken in the execution of this project is the integration of the replica management and data transfer technologies currently in use at the SDM Center with Logistical Networking technologies designed to take advantage of highly distributed resources that are not located at traditional computation or data centers. This synthesis would represent a new approach to flexible data storage and management, creating an interoperable framework for managing the location and transfer of data stored in the network. By generalizing the concept of replica management as implemented in current tools, we will enable many new methods of working with large datasets.

It must be possible to obtain a subset of a large dataset either through an advance request to the replica manager or through dynamic mechanisms such as caching that use application hints and lightweight resources available only at runtime. SRMs can be used to bring the files to a site that will perform the data extraction before providing it to the scientist. Shared storage resources can be

managed by a combination of SRMs and Logistical Networking depots serving different classes of application needs. Using these tools, dynamic data movement for the purpose of performance optimization will be transparent to the end user.

7. Conclusions

A successful Fusion Simulation Project running on leadership class computers ultimately requires a very strong data management component. As codes become mature, and are optimized on leadership class computers, the bottleneck in the scientific investigation process will no longer be the runtime of the simulation on the computer, but rather the other steps in the process, in particular the workflow automation/data streaming technologies.

Data management techniques span six important areas which were highlighted in the 2004 DOE Data management workshop:

- Workflow, data flow, data transformation
- Metadata, data description, logical organization
- Efficient access and queries, data integration
- Distributed data management, data movement, networks
- Storage and caching
- Data analysis, visualization, and integrated environment.

In this paper, we highlighted several parts from this list. It is important to keep in mind that as the number of people working on the code grows; strong data management techniques become a necessity, and not a luxury. It is vital for the various offices in the DOE community to support a strong data management effort, and to be able to link this effort with efforts in related technologies, such as visualization.

References

[1] J.S. Plank et al: The Internet Backplane Protocol: Storage in the Network, *NetStore99: The Network Storage Symposium*, Seattle, WA, USA, 1999

[2] V. Bhat et al: High Performance Threaded Data Streaming for Large Scale Simulations. GRID 2004: 243-250

[3] V. Bhat et al: Fault Tolerant Data Streaming for Ultra Scale Production Simulations. PPPL Technical Report

[4] http://w3.pppl.gov/elvis

[5] Arie Shoshani, Alexander Sim, and Junmin Gu, Storage Resource Managers: Essential Components for the Grid, chapter in book: Grid Resource Management: State of the Art and Future Trends, Edited by Jarek Nabrzyski, Jennifer M. Schopf, Jan weglarz, Kluwer Academic Publishers, 2003.

[6] Alex Sim, Junmin Gu, Arie Shoshani, Vijaya Natarajan, DataMover: Robust Terabyte-Scale Multi-file Replication over Wide-Area Networks, Proceedings of the 16th International Conference on Scientific and Statistical Database Management (SSDBM 2004), Greece.

[7] Ki, Klasky: “Scivis”, Concurrency-Practice and Experience 10 (11-13): 1107-1115 Sept. 1998.

[8] Klasky, et al.:“Grid-based Parallel Data Streaming Implemented for the Gyrokinetic Toroidal Code, SC2003 technical paper, 2003.

[9] http://ptolemy.berkeley.edu

[10] H. Liu, L. Jiang, M. Parashar and D. Silver, "Rule-based Visualization in the Discover Computational Steering Collaboratory", *Journal of Future Generation Computer System, Special Issue on Engineering Autonomic Systems*, 21(1), 2005, 53-59.

[11] H. Liu and M. Parashar**,** "Accord: A Programming Framework for Autonomic Applications"**,** *IEEE Transactions on Systems, Man and Cybernetics, Special Issue on Engineering Autonomic Systems, IEEE Press*, 2005.

[12] M. Parashar, H. Liu, Z. Li, V. Matossian, C. Schmidt, G. Zhang and S. Hariri, "AutoMate: Enabling Autonomic Grid Applications", Cluster Computing: The Journal of Networks, Software Tools, and Applications, Special Issue on Autonomic Computing, Kluwer

Academic Publishers, (2005).
[13] V. Mann and M. Parashar, *DISCOVER: A Computational Collaboratory for Interactive Grid Applications*, in T. Hey, ed., *Grid Computing: Making the Global Infrastructure a Reality*, John Wiley and Sons, 2003, pp. 727-744.
[14] Y.-J. Chiang and C. T. Silva. I/O optimal isosurface extraction. In Proc. IEEE Visualization, pages 293–300, (1997).
[15] J. B. Douglas Thain, Se-Chang Son, and Miron Livny, "The Kangaroo Approach to Data Movement on the Grid," in *Proceedings of the Tenth (IEEE) Symposium on High Performance Distributed Computing (HPDC10)*, 2001.
[16] J. B. B. Allcock, J. Bresnahan, A. L. Chervenak, I. and C. K. Foster, S. Meder, V. Nefedova, D. Quesnal, S. Tuecke, "Data Management and Transfer in High Performance Computational Grid Environments", *Parallel Computing Journal*, 28(5), 2002, 749 - 771.
[17] I. F. M. Ripeanu, "A Decentralized, Adaptive, Replica Location Service," in Proceedings of 11th IEEE International Symposium on High Performance Distributed Computing (HPDC-11), 2002.
[18] http://www-user.slac.stanford.edu/rmount/dm-workshop-04/Final-report.pdf
[19] I Altintas., S. Bhagwanani, D. Buttler, S. Chandra, Z. Cheng, M. Coleman, T. Critchlow, A. Gupta, W. Han, L. Liu, B. Ludaescher, C. Pu, R. Moore, A. Shoshani, M.A. Vouk, "A Modeling and Execution Environment for Distributed Scientific Workflows," Proc. 15th IEEE Intl. Conference on Scientific and Statisical Database Management, 2003.
[20] B. Ludäscher, I. Altintas, C. Berkley, D. Higgins, E. Jaeger, M. Jones, E. Lee, J. Tao, Y. Zhao, Scientific Workflow Management and the Kepler System, *Concurrency and Computation: Practice & Experience*, Special Issue on Scientific Workflows, to appear, 2005.
[21] K. K. Baldridge, J. P. Greenberg, W. Sudholt, S. Mock, I. Altintas, C. Amoreira, Y. Potier, A. Birnbaum, K. Bhatia, M. Taufer, The Computational Chemistry Prototyping Environment, Proc. of the IEEE, Special Issue on Grid Computing, 2005
[22] Sandeep Chandra, "Service-based Support for Scientific Workflows," M.S. Thesis, N.C. State University, 2002
[23] Vouk M.A., and M.P. Singh, "Quality of Service and Scientific Workflows," in *The Quality of Numerical Software: Assessment and Enhancements*, editor: R. Boisvert, Chapman & Hall, pp.77-89 , 1997
[24] Dennis R.L., D.W. Byun, J.H. Novak, K.J. Galluppi, C.C. Coats, M.A. Vouk, "The Next Generation of Integrated Air Quality Modeling: EPA's Models-3," Atmospheric Environment, Vol 30 (12), pp 1925-1938, 1996.
[25] Scientific Data Management Center project, http://sdm.lbl.gov/sdmcenter
[26] Scientific Process Automation Project (http://www-casc.llnl.gov/sdm/)
[27] Kepler Scientific Workflow System (http://kepler-project.org/)
[28] B. Ludaescher, C. Goble, editors, ACM SIGMOD-Record, Special Section on Scientific Workflows, Sept. 2005, to appear.
[29] GEON, Cyberinfrastructure for the Geosciences, http://www.geongrid.org
[30] Science Environment for Ecological Knowledge, http://seek.ecoinformatics.org
[31] Daniel Colonnese (M.S., 2004, "Grid Service Data Needed for Estimation of Reliability in Scientific Workflow Systems")
[32] Sangeeta Ramesh Bhagwanani, (M.S, 2005, "An Evaluation of End-User Interfaces of Scientific Workflow Management Systems")
[33] http://w3.pppl.gov/cemm/