The Scientific Data Management Center for Enabling Technologies

Abstract

With the increasing volume and complexity of data produced by ultra-scale simulations and high-throughput experiments, understanding the science is largely hampered by the lack of comprehensive, end-to-end data management solutions ranging from initial data acquisition to final analysis and visualization. The SciDAC-1 Scientific Data Management (SDM) Center succeeded in bringing an initial set of advanced data management technologies to DOE application scientists in astrophysics, climate, fusion, and biology. Equally important, it established collaborations with these scientists to better understand their science as well as their forthcoming data management and data analytics challenges.

Building on our early successes, we will improve the SDM framework to address the needs of ultra-scale science. Specifically, we will enhance and extend our existing tools to allow for more interactivity and fault tolerance when managing scientists' workflows, for better parallelism and feature extraction capabilities in their data analytics operations, and for greater efficiency and functionality in users' interactions with local parallel file systems and remote storage. These improvements will prepare the SDM framework for the scalability and complexity challenges presented by hardware and applications at ultra scale, and are complemented by our targeted data management efforts under partnerships with application and computer scientists.

1. Executive Summary

Motivation

Managing scientific data has been identified as one of the most important emerging needs by the scientific community because of the sheer volume and increasing complexity of data being collected. Effectively generating, managing, and analyzing this information requires a comprehensive, end-to-end approach to data management that encompasses all of the stages from the initial data acquisition to the final analysis of the data. Fortunately, the data management problems encountered by most scientific domains are common enough to be addressed through shared technology solutions. Based on the community input, we have identified three significant requirements. First, *more efficient access to storage systems is needed*. In particular, parallel file system improvements are needed to write and read large volumes of data without slowing a simulation, analysis, or visualization engine. These processes are complicated by the fact that scientific data are structured differently for specific application domains, and are stored in specialized file formats. Second, scientists require technologies to facilitate better understanding of their data, in particular *the ability to effectively perform complex data analysis and searches over large data sets*. Specialized feature discovery and statistical analysis techniques are needed before the data can be understood or visualized. To facilitate efficient access it is necessary to keep track of the location of the datasets, effectively manage storage resources, and efficiently select subsets of the data. Finally, generating the data, collecting and storing the results, data post-processing, and analysis of results is a tedious, fragmented process. Tools for *automation of this process in a robust, tractable, and recoverable fashion are required to enhance scientific exploration*.

Accomplishments

The Scientific Data Management (SDM) Center (http://sdmcenter.lbl.gov), funded under the first phase of SciDAC (referred to as SciDAC-1), has made a large impact on the DOE domain scientists and the community by providing advanced data management technologies addressing the aforementioned problems. As part of our evolutionary technology development and deployment process (from research through prototypes to deployment and infrastructure) we have organized our activities in three layers that abstract the end-to-end data flow described above. We labeled the layers as **Storage Efficient Access** (**SEA**), **Data**

Mining and Analytics (**DMA**), and **Scientific Process Automation** (**SPA**). The SEA layer is immediately on top of hardware, operating systems, file systems, and mass storage systems, and provides parallel data access technology and transparent access to archival storage. The DMA layer, which builds on the functionality of the SEA layer, consists of indexing, feature selection, and parallel statistical analysis technology. The SPA layer, which is on top of the DMA layer, provides the ability to compose scientific workflows from the components in the DMA layer as well as application specific modules. Figure 1 shows this organization and the components developed by the center and applied to various scientific applications.

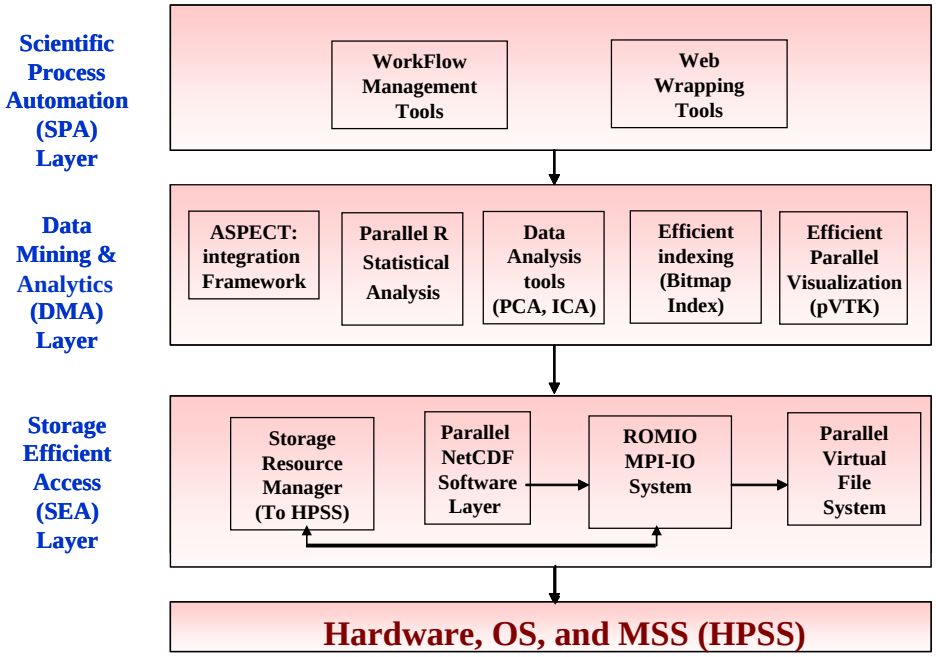


Figure 1: The Scientific Data Management Framework

Over the last several years, the technologies supported by the SDM center have been deployed for a variety of application domains. Some of the most notable achievements are:

- More than a tenfold speedup in writing and reading netCDF files has been achieved by developing MPI-IO based *Parallel netCDF* software being utilized by astrophysics, climate, and Parallel VTK [LLC+03].
- Scalability to thousands of processors has been achieved by sophisticated usage of parallel and collective I/O techniques in Blast, a widely used bioinformatics sequence analysis tool [LMC+05]. The resulting *mpiBlast-pio* open source library has overcome a hundred node scalability bottleneck of mpiBlast.
- An improved version of PVFS is now offered by cluster vendors, including Dell, Atipa, and Platform, and PVFS is the only freely available parallel file system supported on IBM's BlueGene/L [RL03].
- A method for the correct classification of orbits in puncture plots from the National Compact Stellarator eXperiment (NCSX) at PPPL was developed by converting the data into polar coordinates and fitting second-order polynomials to the data points [BK06].
- A new method for signal separation of observational data was achieved by the use of a combination of Principal Component Analysis (PCA) and Independent Component Analysis (ICA). This was used to identify accurately El Nino signals in observed data that included a volcano signal in a Climate application [FK03].
- A new *bitmap indexing* method has enabled efficient search over billions of collisions (events) in High Energy Physics, and is being applied to combustion, astrophysics, and visualization domains. It achieves more than a tenfold speedup in generating regions and tracking them over time [WGL+05, WOS06].

- The development of a *Parallel R*, an open source parallel version of the popular statistical package R. These are being applied to climate, GIS, and mass spec proteomics applications [YSB+05].
- An easy-to-use GUI-based software, called *ProRata*, has provided DOE GTL researchers with robust quantification of protein abundance in high-throughput shotgun proteomics data [PKT+05, PKM+06].
- A *scientific workflow* management and execution system (called *Kepler*) has been developed and deployed within multiple scientific domains, including genomics and astrophysics. The system supports design and the execution of flexible and reusable, component-oriented workflows [ABB+03, AJB+04].

Future focus

The technology developed by the SDM center is of a general nature that can be applied to multiple application areas based on their needs. Building on our early successes, we plan to improve the SDM framework to address the needs of ultra-scale science, providing end-to-end support for the data management needs of scientists. To achieve end-to-end capabilities we will incorporate into the scientific workflow framework any appropriate technology, whether developed by the center or developed elsewhere, such as visualization tools, or fast wide-area data transfer tools.

Specifically, we propose to focus our attention on the following aspects:

- In the SEA area we plan to: (i) extend the efficiency of parallel-IO and MPI-IO through advances in the areas of collaborative caching, efficient coordination, extended attributes, and collective I/O; (ii) continue our evaluation of available HEC I/O systems and collaborate with the PERC SciDAC to enable the use of their performance analysis, prediction, and tuning techniques in the I/O area; (iii) develop a common "bridge" API that many applications can use to abstract away the details of specific underlying storage (e.g. HDF5, PnetCDF) and allow for eventual migration to new formats; and (iv) provide transparent access to archived and remote data using Storage Resource Managers (collaborating with the SRM CET).
- In the DMA area we plan to: (i) enhance and parallelize the specialized indexing technology, called FastBit, to support important new data structures, such as Adaptive Mesh Refinement (AMR) structures; (ii) expand and apply the feature extraction software to new applications areas, notably combustion and fusion, and package the software to be used in workflows; (iii) develop “Parallel Scientific Data Analysis Library” technology to allow various data analysis modules such as R, MATLAB, or IDL to be more easily parallelized and integrated into a generic data analysis framework.
- In the SPA area we plan to: (i) provide systematic support for monitoring and debugging of large scale workflow processes; (ii) enhance the Kepler system to provide fault tolerance and graceful handling of failures: (iii) improve scientists’ ability to interact with workflows, (iv) provide integrated support for both tightly and loosely coupled workflows.

The SDM center’s management & coordination

Management

In order to support the additional goals of the Center over the years we are proposing to add two new partners that will enhance our capabilities in two areas. PNNL will enhance the Center’s core infrastructure through the addition of Active Storage technology and help in making a strong connection to two new application areas of interest to DOE: computational biology and groundwater reactive transport modeling and simulation. MIT was added in order to enhance the center’s capability in the scientific data analysis framework, through their plug-ins for parallel Matlab.

The center’s director is the PI, Arie Shoshani, and Doron Rotem acts as his deputy helping with the day-to-day operation of the center. The center is now organized along the three areas: SEA, DMA, and SPA. In order to ensure vertical integration of these areas, the center has “area leaders” who coordinate meetings, conference calls and integration in their respective areas. The area leaders are also responsible for the agenda and summary of conference calls as well as quarterly report for their area. The area leaders are Rob Ross (ANL) for SEA, Nagiza Samatova (ORNL) for DMA, and Terence Critchlow (LLNL) for SPA.

We have set up weekly conference calls for each of the three areas, with the fourth week assigned to monthly executive committee (area leaders and PIs) conference calls. The executive calls ensure vertical integration of activities between the three areas. We also hold semi-annual all-hands meetings to review progress, and work together on joint plans. Quarterly progress reports are generated.

Collaboration with other SciDAC activities

Our interactions with application scientists in the past proved most successful when we dedicate our staff members' time to work closely with the scientists, understand their specific needs, and introduced them to technology they have not used before. Naturally, this mode of operation is limited. For this reason, we have made contacts with new potential SciDAC activities as well. If funded, we intend to collaborate with other CETs, Institutes, and SAPs to enhance the center's capabilities.

Two of the SDM center members are also part of two other centers, CCA and PERC. The collaboration with CCA helps bring the CCA technology into workflow actors. The collaboration with PERC is mainly in the area of performance of parallel-IO systems. Another collaboration activity already in place is with the Fusion Center for Plasma Edge Simulation (CPES), which provided the SDM center with funding in order to adapt the Kepler workflow technology to their needs, and provide data archiving technology in the workflow using Storage Resource Management (SRM).

As for collaboration with proposed activities, we intend to continue using the Storage Resource Management (SRM) technology from the proposed SRM CET, well-packaged visualization technology from the Visualization and Analytics center (VACET), metadata technology from the Center for Enabling Data Intensive Computing Applications (CEDIC), and integrate fault management capabilities into our parallel I/O components, particularly PVFS2 for Center for Improvement of Fault Tolerance in Systems (CIFTS).

We also intend to collaborate with numerous SAPs and Institutes to apply the SDM center's technology in various application areas. This includes the LBNL-led SAP on Infrastructure Support for Adaptive Mesh Refinement (in the areas of parallel-IO and automated data archiving and transparent remote access); the BNL-led SAP on Objects On-Demand (using bitmap indexing in HENP applications), the ORNL-led PetaScale Initiative (PSI) project (workflow automation tools and parallel I/O performance analysis and tuning), and providing workflow design, development, and customization for the UCSD Shewanella project and the UC Davis Dark Matter project. We intend to collaborate with two institutes that can potentially reach new application domain scientists as well as students by participating in their workshops: the Data Gateways Institute and Princeton Institute for Computational Science and Engineering Research.

Outreach

In addition to these very intensive outreach activities working closely with various scientists, we conducted tutorials and "Birds of a Feather" at SuperComputing conferences and SIAM conferences, published more than 120 papers, including a best paper award (see Appendix 2), supervised 7 theses and gave more than 40 invited presentations (some are keynote addresses) at conferences and meeting (see Appendix 3).

Open source availability

For software developed under funding from this proposal, the SRM center intends to make the source code fully and freely available for use and modification throughout the scientific community, consistent with open source policy guidance issued by DOE's Office of Science. The SDM center will use a variety of distribution mechanisms, tailored to the particulars of the target audience, including web sites, Sourceforge.org web sites, on-site workshops, and collaborations with other research and industrial organizations.

2. Some Motivating Use Cases

Workflows provide a way for the center to achieve an end-to-end data management capability and to integrate the use of the various components provided by the center and elsewhere. For example, running a sequence of time steps of a simulation code on a parallel machine, storing the output data effectively on a parallel file system, archiving the intermediate check-point data, concurrently running an analysis of one or more steps dynamically, then examining the results either statistically or with a visualization tool, can be orchestrated with a workflow engine. We discuss below some use cases that can benefit from such orchestration, and the

challenges they pose to ensure the correct execution of such a demanding activity, especially when large data volumes are involved.

Code Monitoring and Adaptation. An important part of any long-running simulation is monitoring of its progress and outputs. This realtime feedback enables scientists to decide whether (a) to stop a simulation, (b) change a set of parameters (e.g. the number of paths in the Monte Carlo routine for higher accuracy), (c) adjust the simulation code (e.g. invoke a different solver or analysis routine), (d) utilize different system resources (e.g. duplicate to increase throughput and guarantee termination within the expected timeframe), or (e) continue. These adjustments can potentially lead to better science and avoid unnecessary use of computational cycles, data storage to file systems, and data transfers to archives. As easy as it sounds, manually orchestrating this complex scientific workflow is a daunting task, so automation is necessary. Workflow automation tools, while still maturing, already play an important role in effective and efficient analysis of the monitored data. However, the demands on the workflow automation system and the analysis tools used for monitoring increase as science grows more sophisticated (e.g. across multiple spatial and temporal scales with hundreds of variables and thousands of dimensions). First, proper interpretation of this high-dimensional data requires more than a simple monitoring of a few variables but a search for a subset of data of interest (Section 3.2.1) and even advanced multivariate features extraction and tracking (Section 3.2.2). Second, scientists want to be able to plug-in into a workflow system an analysis routine written in their language of preference (Matlab, IDL, R) and to run it efficiently (Section 3.3.3). Third, as more simulation data gets monitored, efficient storage of this data (Section 3.1), utilization of storage resources for "mainstream" data analysis (Section 3.1.5) and sharing the results of these analysis across distributed sites (Section 3.1.4) become more demanding. Finally, a workflow engine that invokes the appropriate analysis components automatically should be dynamically and easily updatable (Section 3.3.1) to allow explicit modification of the data analysis pipeline in both loosely and tightly coupled manner (Section 3.3.3), with modifications applied as new data enters the pipeline. It should support the efficient termination of a specific execution request, halting all related pre- and post-processing activities when the simulation is explicitly stopped or any intermediate component fails (Section 3.3.2). It should also allow to capture dynamic workflow evolution by recording metadata about the runs and analyses steps that were performed (i.e. data provenance) (Sections 3.3.2 and 3.1.1).

Interactive Debugging of HPC-codes. A task that is quickly becoming overwhelming is the debugging of complex simulations. While parallel debugging tools exist, they are not commonly used because of their complexity and cost. Instead, scientists tend to insert print statements into their code to output information as they try to understand a certain runtime phenomenon. This information routinely needs to be processed further to gain insight. Unfortunately, it is extremely rare that a single debug run generates sufficient insight to solve the problem. Instead, multiple runs, with slightly different parameter settings and slightly different output statements, are usually needed for debugging. Again, workflow automation tools can significantly help in this process. A workflow could be set up that executes the simulation and performs the required data analysis repeatedly with a variety of input parameters. An adaptive workflow system would easily support modification based on the available variables. Data and workflow provenance tracking can provide a mechanism for recording the runs and their results.

Other Special Requirements. As workflows are increasingly used in production environments, it becomes increasingly important that they are able to effectively manage large data sets and perform long-distance data transfers efficiently. This requires a multi-pronged approach including handling of streaming data between machines different from the machine executing the workflow engine, persistent archiving of data and handling of MxN data transfers (dynamically restructuring the data coming from a cluster with M processors to a cluster with N processors) between simulations and the associated data post-processing steps. While improving data transfer protocols is outside the scope of this center, our workflow automation system is able to easily incorporate tools such as those used for data transfer, and as the system becomes cognizant of the data and processes it will be able to automatically adjust to reduce potential bottlenecks.

3. Summary of accomplishments and plans

The following three sections describe each of the three layers of the framework. The sections are organized starting from the bottom layer up, as upper layers rely on the layer(s) below them. Thus, the order of the sections is SEA, DMA, and SPA. The detailed milestones for every section are provided in Appendix 1. Appendices 2 and 3 contain lists of publications, references, presentations, and theses. Appendix 4 contains further details of previous accomplishments organized according to the sections. Appendix 5 contains letters-of-support, and Appendix 6 contains a glossary of terms used in the proposal.

3.1 Storage Efficient Access (SEA)

The core I/O functionality on today's high-end computing systems consists of a collection of I/O software (the I/O software stack, Figure 2) that provides a convenient and efficient interface to the available I/O hardware. The projects in this layer focus on this core I/O functionality, and they have two complementary goals. First, we develop and support a collection of highly-scalable and freely available I/O software components that are used in production applications by scientists, and we actively engage the community to help application scientists better understand how to use these tools. Second, through our interactions with the community we identify specific deficiencies in functionality, performance, and usability that we then work to address. Successful improvements are subsequently integrated into production releases, ensuring that these benefits are made widely available.

Our three keystone components are the PVFS2 parallel file system [CLR+00], the ROMIO MPI-IO implementation [TGL99], and the Parallel netCDF (PnetCDF) high-level I/O library [LLC+03]. These tools together address the scalability requirements of current and upcoming high-end systems and are designed to leverage the technology improvements in areas such as high-performance networking and object storage. All three components are open source, made freely available, and operate on a wide variety of systems, making them broadly applicable for production use. They are also widely used as the basis of I/O research projects, so optimizations are often first available for these packages.

Wide-area data access is becoming an increasingly important part of many scientific workflows, with the LBNL Storage Resource Manager (SRM) tools being a popular mechanism for enabling these workflows. In order to most seamlessly interact with wide-area storage systems, these resources should be accessible through the existing I/O software stack. Integrating support for these resources into the I/O stack began in the previous SDM effort, and we plan to continue that effort. Effectively leveraging these systems is a challenge that impacts all layers of the software stack.

3.1.1 Advanced I/O Infrastructure

Background and Motivation

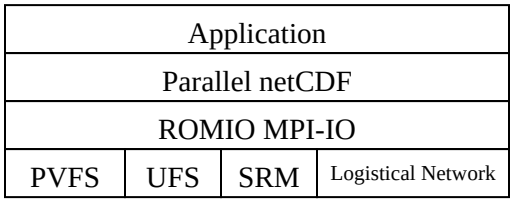


Figure 2: Example I/O software stack

Multiple parallel file system options are now available, and most HEC systems now include a rudimentary I/O software stack. However, the capabilities of most file system options are very limited because they are based on the POSIX I/O API, which was originally developed for sequential applications. This limitation causes a cascade effect, where inefficiencies at the file system layer propagate up the I/O software stack, resulting in very poor performance for parallel application access patterns.

As HEC systems scale and application complexity increases, enhancing the capabilities of I/O system software is critical to maintaining the usability and performance of the I/O system. A significant fraction of the SEA effort is dedicated to bringing improvements in this core infrastructure into production use.

Accomplishments

The development, deployment, and support of the PVFS2 parallel file system, the ROMIO MPI-IO implementation, and the Parallel netCDF high-level I/O library have had a tremendous impact on the community. These tools form a complete I/O software stack that may be deployed on a variety of systems

(and in particular Linux clusters) at no additional cost. In fact, these components are deployed at many large computing centers: ROMIO is the basis of the IBM BG/L, Cray X1E and XT3, and MPICH2 and OpenMPI MPI-IO implementations; PVFS2 has been deployed at sites including Argonne National Laboratory, Ohio Supercomputer Center, and Acxiom Corp. (a commercial data management company); and Parallel netCDF is being built and installed on many HEC systems.

Improvements to these packages such as the list I/O and datatype I/O optimizations continue to push the envelope in terms of the efficiency and capability of HEC I/O systems [CCL+03, CCC+03. Enhancements such as these are integrated into supported versions so that users gain access to these advances, reference implementations are made available for study, and competition in the field of I/O software is fostered.

Future Work

Through our base research program we have developed algorithms using portable MPI-2 one-sided operations to implement **MPI-IO shared-file-pointers, atomic mode, and range locks** for use in I/O optimizations [LRT+05,RLG+05,TRL05]. In this effort we will integrate these algorithms into production ROMIO releases and evaluate their performance for SciDAC applications using application I/O kernels (discussed below). These algorithms offer significantly improved performance over the capabilities in commercial file systems.

To address the latency issues inherent in remote data access we propose to implement and integrate a **portable, collaborative, access pattern aware client-side cache**. Prototypes of this system have been implemented previously by members of this team [CCL+05]. The approach is superior to existing file system caching in that it leverages a globally accessible pool that eliminates copies, moves the burden of coherence control away from the file system, and uses distributed metadata to avoid bottlenecks. This effort will develop and integrate a production-quality collaborative cache into the ROMIO MPI-IO implementation using portable communication primitives (the prototype uses the non-portable Portals library from Sandia). Additional work will augment the system with access pattern awareness and cache page migration.

We propose to investigate the use of **extended attributes for HEC computing** purposes. Extended attributes (EAs) are becoming a standard feature of file systems, allowing user and library data to be associated with a file without disturbing the contents. PVFS2 now includes EA support, allowing experimentation with EAs in the context of HEC applications and libraries. This work proposes three major functionality improvements. First, methods to store MPI-IO hints and access pattern data in EAs will be created, allowing other components to leverage this data. Second, MPI-IO interface extensions for storing and accessing extended attributes will be defined and prototyped in ROMIO. For example, this will allow for portable storage of provenance information created during scientific workflow execution. Finally, options for storing high-level I/O library metadata in EAs will be prototyped, allowing for more efficient high-level I/O access.

We plan to implement a **versioned parallel file system**. Version-based approaches can enable a greater degree of concurrency and remove shared state from the system, providing a more fault-tolerant design than possible with traditional lock-based approaches used to implement sequential consistency. We will implement a version server for PVFS2 and use versioning to order operations in a way that provides stronger consistency semantics than possible with the existing system. Versioning will also allow us to efficiently implement parity-based RAID techniques, and the log-based format will efficiently store noncontiguous data, allowing for faster noncontiguous writes. File snapshots are also trivial in this versioning system, as an artifact of the versioning process. Finally, MPI-IO atomic mode operations will be optimized further using this system.

3.1.2 Understanding Science I/O

Background and Motivation

The high peak rates of HEC I/O systems simply do not translate into adequate sustained performance for computational science applications. The root cause of this performance gap is the mismatch between the requirements of the system's applications and the capabilities of I/O hardware and software. Systematic evaluation of both I/O system capabilities and application requirements would provide much needed insight into the efficient use of existing systems and help guide the design of next-generation I/O systems.

Accomplishments

This group has already begun to characterize existing I/O subsystems for the purpose of better understanding their strengths and weaknesses [RLT+05] and created the Parallel I/O Benchmarking Consortium, a collection of parties interested in I/O benchmarking for HEC computing. These efforts form the basis for continued evaluation and understanding of I/O subsystems. On the application side, the FLASH I/O kernel has had a significant impact on the community as a tool for understanding application I/O performance [RNC+01]. This code that simulates the checkpoint process used in the FLASH astrophysics application. We have used the FLASH I/O kernel extensively as a research tool, and through our interactions with SciDAC application teams we are well-positioned to create similar benchmarks representative of SciDAC application domains.

Future Work

Because HEC I/O systems change over time, a **regular characterization of I/O on HEC systems** is warranted**.** This characterization will build on the recent work performed as part of the SciDAC SDM center and presented at SC05 [RLT+05]. We will evaluate existing I/O benchmarks and create new benchmarks to represent real SciDAC application I/O requirements. We will also investigate options for processing of benchmark data to aid in tracking historical trends and performing comparative studies. Using these tools, we will regularly evaluate existing I/O systems at a variety of sites, and we will offer a set of guidelines to both the SciDAC application teams and the system operators for improving performance based on our findings.

We propose to create three new **I/O kernels for key SciDAC application areas**. We will interact directly with SciDAC application groups to distill key I/O patterns from their applications in order to create codes that closely match the real behaviors of the applications themselves. We will then enhance the kernels to allow for multiple output options, including HDF5 and PnetCDF. These codes will form the basis of a more representative toolset for evaluation and tuning of HEC I/O architectures, and they will provide I/O researchers with convenient proxy workloads. As an emerging area with challenging I/O characteristics, bioinformatics is one area of focus for this work.

We propose to **collaborate with the PERC SciDAC** to enable the use of their performance engineering techniques in the I/O arena. Significant advances have been made in the area of performance prediction, analysis, and tuning in recent years, primarily in the areas of computation and communication. We will instrument our PVFS2, ROMIO, and PnetCDF packages to provide the input necessary for the performance engineering process, and we will provide synthetic I/O benchmarks along with the new I/O kernels as sample workloads for analysis. PERC-3, if funded, will integrate I/O into its existing methodologies and tools for performance engineering, using our instrumentation and I/O benchmarks. With this addition, PERC-3 will be able to provide users with a complete spectrum of performance engineering techniques for all of the major application performance factors: computation, communication, and I/O. In the event PERC-3 is not funded, we will investigate the use of other tools, such as the Jumpshot MPI visualization tool, for I/O visualization.

3.1.3 Application I/O Interfaces and Storage Formats

Background and Motivation

The previous section discussed how to better understand application I/O patterns. This understanding allows us to design and tune systems to react well to the behavior we see today, and this is important because applications cannot currently adjust their I/O behavior quickly or easily. In order to make applications more nimble with respect to their I/O behavior, more effort must be spent on the applications and the interfaces that they use to interact with the I/O system. Improvements in these areas will allow applications to rapidly adopt superior I/O technologies as these are made available. Our ability to effect these changes is greatly enhanced by our planned participation in SciDAC efforts in the areas of parallel visualization and astrophysics, providing eager collaborators and early adopters.

Accomplishments

After our experiences with HDF5 showed much room for performance improvement [RNC+01], we chose to develop the Parallel netCDF (PnetCDF) interface [LCC+03], based on netCDF3. We have seen significant performance benefits for real applications using this new interface. Through the development of this PnetCDF

we have given application groups a choice in terms of interfaces to use and have rekindled discussion of application I/O interfaces for SciDAC.

Future Work

First we will **interact with SciDAC application teams to understand their I/O requirements**: what these applications wish to store (e.g. multidimensional arrays, adaptive mesh structures), and what language would be most natural for describing the data they wish to store and access. The goal of this effort is to best understand the commonalities between SciDAC applications with respect to their I/O needs.

Next we will **develop a common "bridge" API** that many applications can use to abstract away the hooks to a specific underlying storage interface, such as HDF5. This bridge API is important for two reasons. First, using the bridge API allows applications to reduce their dependence on particular underlying I/O libraries. Second, the bridge API provides a more accurate description of the I/O desired by the application, enabling more efficient I/O. We will experiment with integrating this bridge API into I/O kernels to validate the approach before approaching SciDAC application teams directly

At the same time, we will **enhance the PnetCDF library** with respect to its ability to meet the needs of application groups. We see two potential improvements to the PnetCDF library as being important to SciDAC applications. The first augmentation is to add variable processing (reduction) operations inside the library itself, such as mean, minimum, maximum, and variance. We also see applications storing data with different spatial ranges in PnetCDF files. For these applications it would be useful to be able to specify spatial ranges directly, rather than manually mapping spatial ranges into variable indices. We will investigate augmentations to the PnetCDF API that will enable this correlation of variables during access.

Finally, we plan to construct a **new storage interface and format** using the knowledge that we have gained from interacting with the community, augmenting the PnetCDF library, constructing the bridge API, and testing the bridge API with I/O kernels and applications. Using the bridge API as a starting point, we will develop a highly-tuned system that better supports I/O for a collection of SciDAC application codes. This new package will provide an API that naturally matches to application needs and will store data in an efficient manner on state-of-the-art HEC systems. Our new application I/O kernels will provide a natural prototyping environment for this development effort.

3.1.4 Multi-Site Data Access to Distributed Storage

Background and Motivation

While high-performance scientific computing is performed mostly on centralized facilities, data grids are becoming the preferred infrastructure for servicing the massive datasets associated with SciDAC applications. Generally these datasets are generated in centralized sites but are archived over distributed mass-storage systems at geographically dispersed sites. We propose to develop mechanisms that SciDAC applications would use to transparently extract desired subsets of datasets from these distributed data sources. This will be achieved by providing interfaces that interact with Storage Resource Managers (SRMs), middleware services that manage disk space quotas, queue requests for multiple files, fetch the requested files from remote sites if necessary, and subsequently deliver the files to the applications. These interfaces can simplify the data-flow in scientific workflows (Section 3.3) by eliminating intermediate data transfers and pre-processing steps.

Accomplishments

Previous accomplishments include the development and implementation of new caching policies that better meet the use of an SRM by taking into account the cost of network delays and sizes of requested files [ORS03, ORRS05], and development of libraries that couple an SRM with the PVFS2 parallel file systems so that MPI applications achieve transparent access to data on a mass storage system, currently the IBM's High Performance Storage System (HPSS) [OSS06].

Future Work

First, we will complete the **MPI-IO interface to SRMs**, removing all additional proprietary function calls, and implementing prefetching and caching of datasets based on application hints. This important step will allow transparent access to remote data using this widely-deployed software infrastructure.

Second, we will provide a **distributed collection of SRMs** that serve effectively as proxy servers (and/or reverse proxy servers) to the client applications and data sources. To accomplish this, the basic SRM technology must be extended to allow the content of a distributed collection of SRMs to be accessed through a peer-to-peer replica service layer. Clients would then also be able to access data from a variety of SRM implementations servicing JASMine, CASTOR, and ENSTORE mass storage systems, in addition to HPSS.

Third, **mechanisms for reducing wide-area data transfer** will be integrated into the system, such as compression and real-time data subsetting and filtering, along with augmented interfaces for describing these types of operations on data stored in some predefined multidimensional file format (e.g. netCDF, HDF5) or specialized file format [Oto06,OR05]. This work is especially important in its impact on the application I/O interfaces and formats described in Section 3.1.3, as we view application interfaces as one place where we might be able to hide the details of local vs. remote data access inside the application interface.

Fourth, **enhancements to the SRM implementation** will allow direct connection between the SRM service and an underlying parallel file system, increased reliability, and monitoring of long-running data transfers.

3.1.5 Data Processing with Active Storage

Background and Motivation

Despite recent advancements in storage technologies for many data intensive applications, analysis of data remains a serious bottleneck. One option for data analysis is to leverage resources not on the client side, but on the storage side. The original research efforts on active storage were based on a premise that modern storage architectures might include usable processing resources at the storage controller or disk; unfortunately, commodity storage has not yet reached this point. However, parallel file systems offer a similar opportunity. Because the servers used in parallel file sytems often include commodity processors similar to the ones used in compute nodes, many Giga-op/s of aggregate processing power are often available in the parallel file system. Our goal is to leverage these resources for data processing. Scientific applications that rely on out-of-core computation are likely candidates for application of this technique, because their data is already being moved through the file system. The Principle Component Analysis (PCA) [Pear01] of the trajectories in molecular dynamics simulations is one example. Data analytics operations, described in Section 3.2, would also be amenable to this technique.

Accomplishments

Our approach to active storage takes advantage of the underutilized CPU resources on file system servers to process data in addition to storing it, additionally avoiding redundant data transfers over networks, interconnects, and storage busses. We have prototyped this concept by modifying the Lustre source base to enable the Active Storage (AS) concept and validated it in the context of a bioinformatics application [FFR+05].

Future Work

We propose to **define and utilize an API for Active Storage Components** within the PVFS2 file system. By modifying the existing Active Storage code to work with the PVFS2 flow components we can easily add a modular Active Storage component that will allow us to use the storage server processing power. Furthermore, we will stabilize and define an external API that will be usable by many implementations of active storage in Lustre or in PVFS and in future ports of the technology. A library of useful AS components will also be built to enhance the usability of AS. One target area for these components will be structured datasets such as NetCDF or HDF to optimize data analytics operations (see Section 3.2) so that summary data for datasets can be calculated at the servers, rather than moving entire datasets over the network.

3.2 Data Mining and Analytics (DMA)

The Data Mining and Analytics (DMA) layer provides the data-understanding technologies necessary for efficient and effective analytics of complex scientific data. This is accomplished through the development and integration of core components for efficiently selecting the subsets of multivariate data (Section 3.2.1), extracting features relevant to the phenomenon of interest (Section 3.2.2), and the optimized library for high-

performance statistical computing (Section 3.2.3). To provide a desired level of efficiency and automation, these components are being tightly coupled with high-level I/O libraries and data-migration tools (Section 3.1) and made available as components of scientific process automation (Section 3.3).

3.2.1 Efficient Searching and Filtering

Background and Motivation

In DOE data management workshop report [Mou04], most applications articulated the strong need to efficient find interesting data records and weed out uninteresting ones. This section addresses the technology for efficient searching and filtering of large-scale scientific multivariate datasets with hundreds of searchable attributes to deliver the most relevant data records to the appropriate analysis tools such as those in Tasks 3.2.2 and 3.2.3. More specifically, **our goal is to develop a scalable searching tool for scientific data and to integrate this tool with data analysis environments**.

A key feature that distinguishes scientific data from typical commercial data is that the bulk of the data is written once but read many times. Our basic approach is to take advantage of this feature to develop a more efficient indexing method for searching scientific datasets. In the past few years, we have been working on a technology that addresses the most common type of searching and filtering through range queries such as "temperature > 100 AND energy < 109." However, there are many data analysis and visualization applications that require more advanced searching capabilities such as "joins" in order to execute queries across datasets. For many data-intensive applications it is also critical to tightly integrate the searching step into the data analysis process, for example, tracking ignition kernels through time. In this section, we discuss our plans (1) *to enrich the types of searches*, (2) *to integrate them into common data analysis frameworks*, and (3) *to make our search tools more robust and broaden their deployment across SciDAC applications.*

Accomplishments

The most efficient indexing method for write-once datasets is known as the *bitmap index*. In the past few years, we have significantly improved the bitmap index by reducing its size and at the same time decreasing the query response time. The software package is called *FastBit*. It uses an efficient bitmap compression, named Word-Aligned Hybrid (WAH) code [WOS01]. The WAH compressed indices are not only modest in size but also have the same optimality property as the most efficient indexing methods such as B-trees [WOS04, WOS06] (Figure 3).

We have also been working with a number of scientific applications to reduce the time needed for their data analyses. For example, we have demonstrated that FastBit can significantly speedup the feature tracking procedure in the analyses of simulation data produced from SNL's combustion research facility [WKCS03] (Figure 4). In addition, we provided high-energy physicists with event-level filtering of their experimental data over the Grid [WZS+03]. This work received a best paper award from International Supercomputing Conference 2005 [WGL+05].

Future Work

We have made significant strides in applying our searching technology to improve scientific applications. We have identified some key issues and the following is our plan to address them.

Enriched types of searches. The proven strength of FastBit is in answering range queries. Though, this is sufficient for many searching and filtering applications on scientific datasets, there are plenty of other types of searching problems that challenge application scientists. For example, in analysis of combustion simulation data, an ignition kernel may grow, split and start multiple kernels. In the database community, the process of searching and tracking this evolution is generally expressed as "*join*" and it is known to be a challenging task for any database system to handle. In the DOE data management report [Mou04], the join operation was emphasized in a number of applications including biology. We have researched the possibility of using bitmap indices to answer *joins*. We plan to implement a set of robust *join* algorithms as part of this effort.

Another common type of queries is known as "*top-K query*", where the *K* top-most records satisfying some user specified conditions are requested, for example, e.g. finding the top 10 most expressed genes in a microarray experiment. This query type has many applications but is challenging to most indexing methods.

We need to study the available options in our research program and evaluate their performance characteristics. Efficient algorithms for answering *joins* and *top-K queries* are planned research topics. Practical algorithms are expected near the end of this project's life time.

Integration with analysis frameworks. Commercial database systems have a notorious reputation for requiring specialists to manage their day-to-day operations. We intend to produce a tool that is seamlessly integrated into existing user analysis frameworks to avoid this usability problem. To this end, we plan to integrate FastBit with a popular analysis framework in high-energy physics community, named ROOT. We also plan to integrate FastBit with popular scientific file formats such as netCDF and HDF. We will collaborate with the SEA team of the SDM center to work on netCDF integration, and with the Visualization group of LBNL and the proposed visualization center on HDF integration.

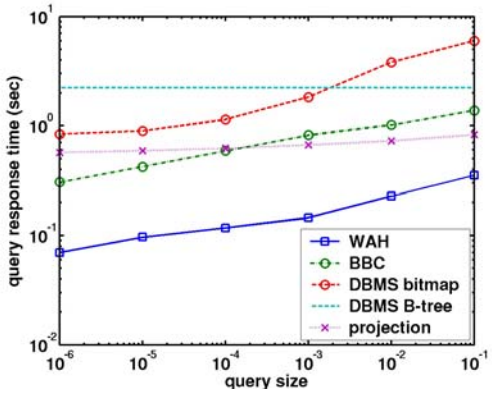


Figure 3: WAH compressed bitmap indices are significantly faster in answering multi-dimensional queries. Time is measured on a set of 5-dimensional queries over a sample of high-energy physics data.

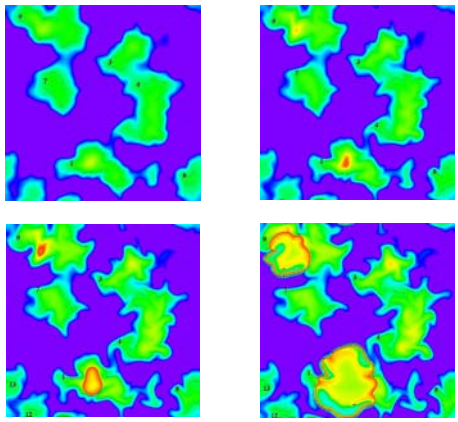


Figure 4: Evolution of ignition kernels in a simulation of hydrogen-air mixture. Kernerls are defined with multivariate conditions and tracked with compressed bitmaps.

Robust and broad deployment. We plan to improve the FastBit software by rewriting its core functionality for 64-bit file handling, parallelizing the software, and adding new features demanded by application scientists. A brief list of new features planned is listed later in the project milestones. We also plan to leverage the proposed to Analysis and Knowledge Discovery Toolkit center and Objects On Demand partnership to speedup the development efforts.

3.2.2 Feature Extraction and Tracking

Background and Motivation

As outlined in [Mou04], the large-scale simulations in support of **combustion** grand challenges are generating terabytes of data per simulation. Of particular interest in these simulations are transient events such as ignition, extinction, and re-ignition, not all of which are well-defined or well-understood. This drives the need for *efficient, on-line algorithms and software for feature detection and tracking*. Similar problems also exist in the area of **fusion**. For example, feature tracking can be used to track blobs of plasma in high-resolution, ultra-high speed images of edge turbulence in the National Spherical Torus Experiment (NSTX) at PPPL. Feature extraction is relevant to the analysis of Poincaré plots (Figure 5) in the National Compact Stellarator Experiment (NCSX) at PPPL. Each orbit in a plot is generated when a particle moving around the device intersects a plane perpendicular to the magnetic axis. The orbits have different shapes depending on the initial starting point of the particle. Physicists are interested in identifying orbits of a particular type and extracting certain key features to characterize the plot.

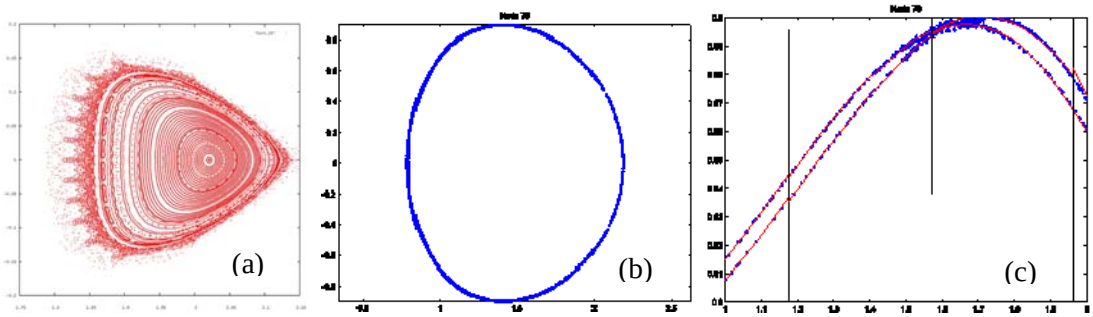


Figure 5: (a) A plot simulating an experiment from the W7A stellarator. (b) A separatrix orbit, which is difficult to identify, until we zoom in as shown in (c). The piece-wise polynomial approach fits two polynomials (in red) to the points (in blue) of the orbit, easily identifying it as a separatrix.

The features extracted from experimental and simulation data can then be used to design new experiments or to verify and validate the simulations by comparing them to each other, to theory, or to experiment.

Accomplishments

Our early work in SciDAC-1 included the separation of signals in climate data from simulations and the identification of key features in sensor data from the D-III Tokamak [FK03, CNK04]. This work on dimension reduction techniques was licensed to General Atomics in August 2005 for continued use in their analyses. Our ongoing work focuses on the analysis of Poincaré plots and the research and development of tracking algorithms. For each problem, we develop and apply more than one solution approach as some approaches may be more accurate or robust based on the quality or the characteristics of the data. Multiple approaches also enable us to address variations in the data and provide scientists with a more reliable result from the analysis.

Our work on Poincaré plots has focused on the classification of orbits into one of three categories: quasi-periodic, separatrix, and island chain. We implemented four approaches: three based on a graphical technique called KAM [Yip91] and one based on a piecewise polynomial approach. The features from the graph or the polynomials are used in rules to assign a class label to each orbit. Our early results show that each method has its pros and cons. The piece-wise polynomial approach is very effective in classifying separatrix orbits, even when the width of the separatrix is too small to be easily detected visually (Figure 5), while KAM requires extensive fine-tuning of the parameters [BK06].

Future Work

Our proposed work will focus on feature detection and tracking, including research and development of algorithms and software, and the deployment and enhancement of the analysis software across various applications. This will leverage the feature extraction and tracking software developed as part of the Sapphire scientific data mining effort at LLNL (http://www.llnl.gov/casc/sapphire). We will first complete the analysis of Poincaré plots. This will include testing and enhancement of the algorithms for classification of orbits in simulation data, extension of these techniques to experimental data, and incorporation of feedback to the domain scientists to improve the selection of points generating the orbits. Next, we will extract features such as the number of islands in an island chain, the width of the separatrix lobes, and the width of islands in an island chain. A particularly challenging problem will be the accurate identification of missing separatrices and the analysis of experimental data where the number of points is small and not explicitly assigned to orbits.

We will then apply and enhance our techniques to data from combustion simulations and experiments. We will address problems with feature extraction from curvilinear grids and features distributed across processors. We will also investigate metrics for the comparison of features and enhance our tracking algorithms to handle features moving across processors and improve the efficiency for real-time tracking.

The analysis algorithms and software will be deployed as either standalone codes, incorporated directly into the application, or used by the application via the workflow developed for it. Specifically, our codes for Poincaré plots will be deployed as part of a workflow [see section 3.3]. The feature extraction and tracking codes for combustion will initially be standalone and after verification, be incorporated into the simulation

software. We will work closely with the application scientists, both in the development of the algorithms and software, and in the deployment and enhancement of the software. All development and testing will be done using data from the applications. Specifically, we will work with Drs. Neil Pomphrey and Don Monticello from PPPL and Dr. Jackie Chen from Sandia National Laboratory.

3.2.3 High Performance Statistical Computing

Background and Motivation

Our goal is to scale-up the existing data analysis tools, both in capability and capacity, to bridge the gap between scientific data production and data processing rates – a well-recognized need by the DOE scientific applications [Mou04]. The challenge is that *many of the current data analysis routines are written in non-parallel languages such as IDL and are not scalable to massive data sets.* To address this challenge, our current approach is based on using a statistical package, called R [VSR05]. Similar to IDL, R is a non-parallel language and is not designed to handle massive data-sets. Our choice of R, however, is driven by its open-source availability, extensibility, and very broad usability by a large scientific community. Our strategy is to provide a middleware between a *specific high-level* data analytics language such as R, IDL, or MATLAB and a *generic, parallel, optimized, portable* computational analytics engine.

Accomplishments

Over the last two years of SciDAC-1, we designed and prototyped this strategy using R as an underlying data analytics language and applied the resulting *Parallel R* [S05] system to a few applications in climate, biology, and GIS. We utilized two major levels of parallelism within Parallel R: *data parallelism* and *task parallelism*. With data parallelism, we provided "hooks" to embed a parallel data analysis routine into an R environment with low performance overhead. As a proof-of-principle demonstration of how to use such hooks, we released an *RScaLAPACK* library [SYBK05, YSB+05] that replaced data analysis routines in R that are based on LAPACK [ABD+99] linear algebra solvers with the corresponding parallel, optimized and portable ScaLAPACK [BCC+97] routines. The new interface has mimicked the original R interface through a single function call (Figure 6). The introduced minor changes to the interface are not mandatory and are provided for the purpose of having user-level control over serial vs. parallel routines.

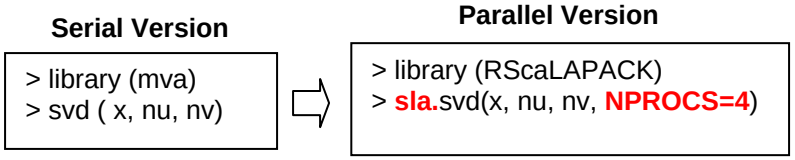


Figure 6: Parallel interface for a serial *svd*() function in R.

With task parallelism, our goal was to provide the task parallel capability, where a given job is divided into various tasks that are executed in parallel to achieve a reduction in the time involved in an analysis process. Specifically, we prototyped a library, called *taskPR* [SBY05] that provides a means to detect the out-of-order independent R tasks and intelligently delegates each task to remote worker processes, to execute them in parallel. Again, the introduced changes to the interface were minimal simply to provide user-level control over task parallel vs. serial execution.

By incorporating these two aforementioned strategies in R, we have demonstrated their usefulness while dealing with large volumes of scientific data. We worked with ORNL Mass Spec Proteomics Center funded under the DOE Genomics:GTL program to develop and parallelize their protein quantification pipeline.

The quantitative proteomics has posed unprecedented informatics challenges in two fronts: massive data management system and development of novel algorithms for very noisy data. We addressed those challenges within our software package, called ProRata [PHM+05, PKT+05, PKT+06, PKM+06] released as an open source to a broad biological community. In addition, we used *taskPR* to significantly reduce protein ratio calculations by employing loop optimization techniques and out-of-order evaluation of serial tasks. We also worked with the SciDAC climate group at ORNL, who use R statistical package for data analysis. Currently, they have a capability to use ParallelR tools for selection and extraction of a data-set from terabytes of simulation data.

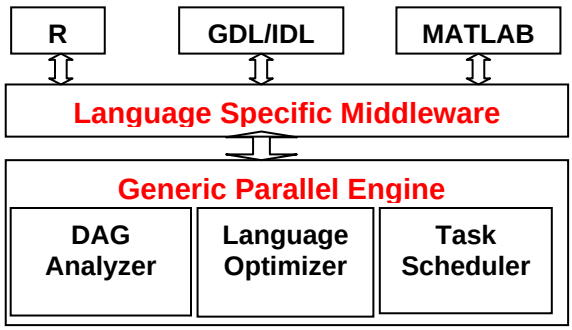


Figure 7: Parallel Scientific Data Analysis Library.

RScaLAPACK and taskPR libraries were successfully tested in both distributed and shared memory systems. These packages are currently distributed across 24 countries through R's CRAN network [SBY05, SYBK05]. RPMs for these packages have been distributed by RedHat, SUSE, and Quantian Linux distributions.

Future Work

To address scalable data analytics needs articulated in the DOE workshop report [Mou04], we aim (1) *to provide an easily extensible mechanism to add third party parallel analysis capabilities to the R library*; (2) *to support additional data analysis languages like MATLAB and IDL*; (3) *to make parallel analysis capabilities available as part of more complex scientific workflows*; and (4) *to broadly deploy this parallel analysis infrastructure across various DOE scientific applications*.

Our overarching approach is to develop a *Parallel Scientific Data Analysis* (*PSDA)* library that will incorporate these techniques in a common parallel statistical computing environment. This library would allow application scientists to use their traditional data analysis codes but execute them on a parallel platform with minor modifications to the codes. The PSDA library comprises *Language Specific Middleware* (*LSM*) and *Generic Parallel Engine* (*GPE*) (Figure 7). LSM will provide language specific parser routines along with a unified interface to the parallel statistical analysis functionalities provided by GPE. The GPE that forms the core module of the PSDA library would be no more limited to R but would be pluggable to similar mathematical analysis environments like MATLAB, IDL/GDL etc. A *Directed Acyclic Graph (DAG) Analyzer* will be developed to identify out-of-order tasks using data dependency techniques. Loop and other control structure optimizations are done through a *Language Optimizer*. An efficient critical path based static scheduling algorithm [KA99] will be implemented to manage computing resources. More specifically, the following outlines our plan to address our objectives.

To address objective #1: We propose to develop a *Generic Parallel Agent* framework that would enable users to easily plug-in their parallel algorithms into their analysis environment. XML based extension templates will be provided to the users to easily build language-compatible interface routines. The framework will allow seamless addition of third party MPI-based parallel analysis and parallel I/O (section 3.1) routines.

To address objective #2: We plan to develop a language specific middleware that allows scientific data-analysis platforms to effectively utilize our parallel routines in their programs, thus employing external parallel computation capability in their day-to-day workflow. By enabling several parallel computation strategies in R, we realized that similar strategies could be used in other data-analysis software environments like MATLAB, IDL etc. Reusing thousands of lines of scientists' codes with minimal changes in parallel environments could be very effective. This will also bring the efficiency to existing serial codes due to underlying transparent parallelization. Re-usage of existing codes also ensures minimal errors and debugging.

To address objective #3: We will make the analysis modules developed under PSDA library accessible as CCA-compliant components, so that these high-performance modules could be used as part of a more generic scientific workflow. The analysis modules will also be made accessible through Web Services to avoid data migration by hosting a service at the location of data. We will work with SPA team to provide this capability.

To address objective #4: We plan to deploy PSDA infrastructure at various SciDAC groups. In the past we have been working with Drs. J. Drake's climate and R. Hettich's proteomics groups. We will explore the usability of PSDA library across other application domains. For example, the primary data analytics platform in climate by NCAR is written in R or MATLAB. Likewise, the SciDAC fusion teams own thousands of lines of IDL codes. An HEP ROOT team is moving towards R as their data analytics platform.

3.3 Scientific Process Automation (SPA)

Scientific Workflows

Workflow technologies have a long history in the databases and information systems communities [GHS95]. Similarly, the scientific community has developed a number of problem-solving environments, most of them as integrated solutions [HRG+00], and finally, component-based solution support systems are also proliferating [CL02, CCA06]. Scientific workflows merge advances in all these areas to automate support for sophisticated scientific problem-solving [LAB+06, LG05, DOE04, ABB+03, BVP00, VS97, SV96].

We use the term ***scientific workflow*** as a blanket term describing a series of structured activities and computations (called workflow components or *actors*) that arise in scientific problem-solving as part of the discovery process (e.g., Figure 8). This description includes the actions performed (by actors), the decisions made (control-flow), and the underlying coordination, such as data transfers (dataflow) and scheduling, required to execute the workflow. In its simplest case, a workflow is a linear sequence of tasks, each one implemented by an actor. An example of a simple workflow is: transfer a configuration file to a cluster, run an application passing this file as an input parameter, transfer the results of the application to a desktop machine, select a known variable, and generate a movie showing how this variable evolves over time. Scientific workflows can exhibit and exploit data-, task-, and pipeline-parallelism. In science and engineering, process tasks and computations often are large-scale, complex, and structured with intricate dependencies [DOE04, Den96, Elm95, Elm66]. Scientific workflows have several common requirements:

1. Invocation of multiple application and analysis tools. While these tools are typically invoked in a routine manner, there may also be changes in the workflow as scientists interactively explore new options. Developing an executable workflow requires resolving mismatches between what a tool expects and what the previous step in the process generated.
2. Significant heterogeneous, computational resources. Many large-scale scientific workflows will execute for weeks, if not months, and require user intervention at multiple times. If the workflow or one of the associated computations runs into trouble, fault-tolerant behavior, e.g., via fail-over techniques must be attempted because returning to the initial starting point is usually not acceptable.
3. Validation of both intermediate and final results. This validation ensures that the computation as a whole remains on track and that resources are not wasted exploring invalid data. Due to the extensive durations of workflow execution, near-real-time status tracking capabilities are required.
4. Tracking of intermediate and final results. Due to the size and complexity of scientific workflows, it is critical that provenance information (e.g., the lineage of data products) is recorded in a way that is consistent, persistent, and easily retrievable.

Accomplishments

As part of original SciDAC SDM Center, the Scientific Process Automation (SPA) thrust has made several significant contributions to improve scientific workflow technology, leading to the adoption of scientific workflows within the DOE community:

- **Co-founded the *Kepler* project**. Based on extensive evaluation of multiple, competing technologies [Cha02, Col04, Bha05, Moul05, MVB+04] we co-founded the Kepler project [Kep06, LAB+06, ABB+03]; a multi-site open source effort to extend the Ptolemy system [Pto06] and create an integrated scientific workflow infrastructure. We have also started to incorporate the SCIRun [SCI06] system into our environment, to better handle the needs of tightly-coupled and visualization-intensive workflows.
- **Deployed workflows in multiple scientific domains.** We have worked closely with application scientists to design, implement, and deploy workflows that address their real-world needs (e.g., Figure 8). In particular, we have active users on the SciDAC Terascale Supernova Initiative (TSI) team and within the LLNL Biotechnology Directorate. The success of these deployments has resulted in the SPA team being approached to participate in several SAPs.

- **Developed specialized actors**. We designed and implemented many actors commonly needed by scientists, e.g., a web service actor to invoke web services and an SSH actor for remote task execution.
- **Extended existing capabilities**. We have added asynchronous service-based process status tracking, as well as a simple provenance tracking capability to the Kepler infrastructure. While significant work is required to fully implement these capabilities for general workflows, the current implementations meet our customers' near-term requirements.
- **Established initial deployment infrastructure**. We have developed a web-based deployment infrastructure for distributing our system, including updates and patches, to users (available on-line at https://www-casc.llnl.gov/sdm/). This infrastructure includes documentation automatically extracted from the actor class definition.

Proposed Research and Development

Despite significant successes, much work remains before scientific workflow technology reaches a state where it is a commodity tool, easily used by many application scientists. As detailed below, the work we are proposing will make another set of significant advances in this direction by fundamentally extending the systems current capabilities in three major areas:

- **User Interactions and Workflow Design**. Current user interfaces will be extended to support "custom dashboards" and web-based portals. New design mechanisms (templates, semantic types, nested collections, user histories) will greatly simplify workflow design and increase its understandability and dependability. We will also evolve workflow user interface capabilities to better support the dynamic, spontaneous nature of computational scientific research.
- **Data Collection and Workflow Management.** The system will be more resilient and fault tolerant, part of which is based on a comprehensive data and workflow provenance framework and new specialized and generic actors, e.g., for reliable data transport, alternative resource acquisition, etc.
- **Workflow Infrastructure.** We will develop system extensions for distributed workflow execution, which can include both tightly-coupled and loosely coupled sub-workflows. The current deployment infrastructure will be improved and extended to include the DOE security infrastructure.

3.3.1 Extending User Interaction and Design Capabilities

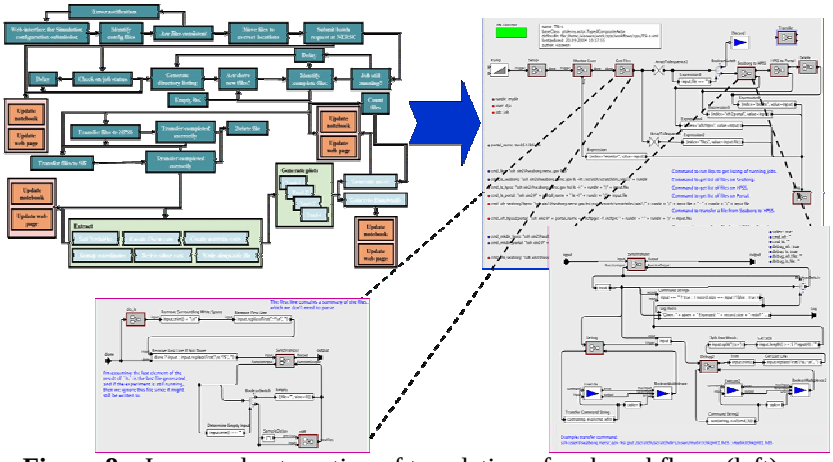


Figure 8 – Increased automation of translation of real workflows (left) into SPA/Kepler is one of the goals of this project. (TSI- D. Swesty w/f).

As highlighted in the motivating use case (Section 2), there are several user interaction and design capabilities that need to be provided by any widely-adopted, scientific workflow infrastructure. First, the ability to initiate a workflow, disconnect from it, and later reconnect to obtain status and progress updates is crucial. Moreover, the UI must be straightforward enough for the application scientist to not only execute and monitor a workflow, but also to modify and (re-)design it (e.g., Figure 8). Thus, we propose to:

Develop domain-specific "dashboards": Workflow related information tracking and provenance has both generic character (e.g., a provenance framework), and domain or application specific character. A dashboard is a mechanism for providing domain-specific "one-stop-shopping" control and monitoring capabilities and presenting a user-friendly interface to the workflow infrastructure. These interfaces will support scientists bringing in new data and applications on-the-fly based on the immediate needs of their experiments,

dynamically defining application parameters based on results emerging from previous steps, and pausing at critical junctures during experiments to examine, analyze, or verify intermediate results. This capability already exists within SCIRun, where so-called PowerApps provide a scientist-oriented interface while retaining the full power of the underlying workflow system. We will evolve and extend workflow user interface concepts in Kepler to support the dynamic nature of computational research processes through new capabilities such as finer execution control of processes and workflows, monitoring capabilities for viewing dataflows, and dynamic forms for collecting parameters and other user input in the midst of workflow execution.

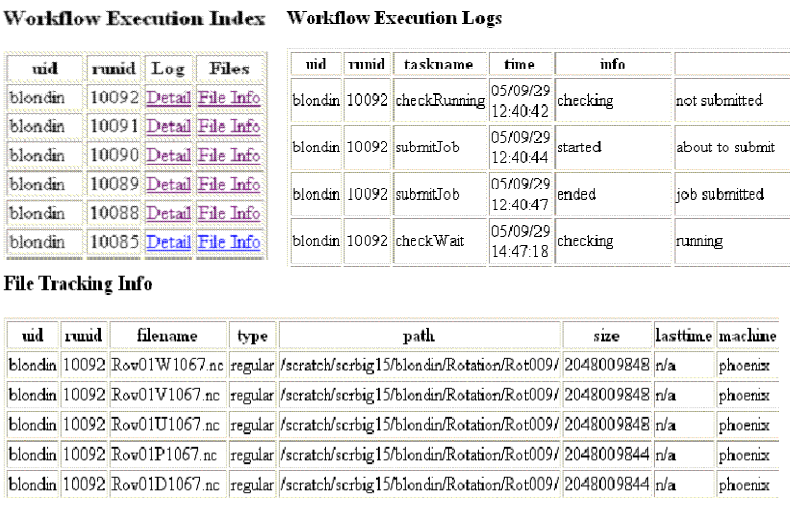
Provide capabilities via portals: Portals are a popular technology and a powerful way to disseminate workflow capabilities through "web-based dashboards" (e.g., Figure 9). We will create a small number of custom portals (for target scientists/communities) which will allow users to interact with scientific workflows in a personalized way. These features are essentially web-accessible server versions of custom dashboards:

- *Personalization.* Workflow execution parameters will be customizable per (portal) user.
- *Workflow control.* Workflows can be started, paused, resumed, and stopped via the portal.
- *Workflow monitoring.* Provenance and monitoring information will be available through the portal.
- *Result dissemination.* The portal will support sharing of results and data within defined user groups.

Develop new workflow design capabilities: Workflow development is currently hard and requires, e.g., detailed information about the operational infrastructure, the set of available actors, specific actions the workflow needs to take, the appropriate error handling, and many additional factors. Fortunately, many of the workflows within a scientific domain have similar requirements, and thus workflow templates and patterns can be used as a starting point for creating a workflow. We will develop workflow "wizards" that simplify the construction of workflows, based on user input, existing workflow templates and patterns, and "semantic type" declarations (annotations of actor port types using controlled vocabularies) [BL06]. The use of templates, patterns, and semantic types, can greatly simplify workflow design and reuse [BL05]. Workflow design and maintenance is further enhanced by developing support for collection-oriented workflows [MB05]. This paradigm is expected to dramatically reduce the (control-flow and branching) complexity of workflow designs by streaming nested data collection, appropriately tagged for downstream processing [BLM06].

3.3.2 Extending Data Collection and Workflow Management Capabilities

So far Kepler development has focused on providing functionality over reliability. However, workflows such as our motivating use case have to be executed on a reliable platform. We thus propose to:



Workflow Execution Index

uid	runid	Log	Files
blondin	10092	Detail	File Info
blondin	10091	Detail	File Info
blondin	10090	Detail	File Info
blondin	10089	Detail	File Info
blondin	10088	Detail	File Info
blondin	10085	Detail	File Info

Workflow Execution Logs

uid	runid	taskname	time	info	
blondin	10092	checkRunning	05/09/29 12:40:42	checking	not submitted
blondin	10092	submitJob	05/09/29 12:40:44	started	about to submit
blondin	10092	submitJob	05/09/29 12:40:47	ended	job submitted
blondin	10092	checkWait	05/09/29 14:47:18	checking	running

File Tracking Info

uid	runid	filename	type	path	size	lasttime	machine
blondin	10092	Row01W1067.nc	regular	/scratch/scrbig15/blondin/Rotation/Rot009/	2048009848	n/a	phoenix
blondin	10092	Row01V1067.nc	regular	/scratch/scrbig15/blondin/Rotation/Rot009/	2048009848	n/a	phoenix
blondin	10092	Row01U1067.nc	regular	/scratch/scrbig15/blondin/Rotation/Rot009/	2048009848	n/a	phoenix
blondin	10092	Row01P1067.nc	regular	/scratch/scrbig15/blondin/Rotation/Rot009/	2048009844	n/a	phoenix
blondin	10092	Row01D1067.nc	regular	/scratch/scrbig15/blondin/Rotation/Rot009/	2048009844	n/a	phoenix

Figure 9. Pilot for real-time monitoring of J. Blondin TSI

Deploy data and workflow provenance capabilities. In scientific applications, effectively managing data and workflow provenance is extremely important [MGBM05, GMF+05, CFS+05, SPG05]. Data provenance [MGBM05] can be thought of as the complete processing history of a data product, e.g., data dependencies, actor invocation parameters (including application codes launched by actors), properties such as the time, location, and user-id of invocation, relevant environment and configuration parameters, etc. This information needs to be persistent and permanently associated with a data product so that its provenance is readily available. A scientist's execution history may be provided in a graphical form from

which specific execution threads may be selected and saved as templates for future extension and reuse. This information should also be searchable, so that data with certain provenance can be easily identified – for example, if a bug is identified and corrected, the provenance can help identifying which runs should be repeated. We will extend the current Kepler data provenance functionality to provide a user-customizable infrastructure for recording and querying provenance information. We will also explore and develop workflow history mechanisms for use in conjunction with the aforementioned workflow templating capabilities.

Workflow provenance tracks the history of the workflow itself, both during workflow design and at runtime: e.g., a workflow may be derived from a template or a previously deployed workflow; or during exploratory "workflow tuning" a parameterized workflow instance will be run with different parameters. Keeping track of this information in a persistent, searchable format, is important for a variety of tasks such as propagation of bug fixes; undoing of changes; and regeneration and validation of previous results. We will develop a workflow provenance system that allows scientists to track and query data and workflow provenance, and share this information with authorized users (e.g., a pilot is shown in Figure 9).

Develop fault-tolerance mechanisms. By their nature, many scientific workflows require a diverse runtime infrastructure. Whenever key resources such as a supercomputer, storage space, or reliable network are not available, the workflow cannot terminate successfully. By default, a run-time exception in a single actor or a single line of script-code (for script-based workflows) may prematurely terminate the entire workflow execution. We propose to enhance fault-tolerance in our system in several ways: a) through (partial or complete) system state recovery and restart techniques; b) through forward recovery techniques, e.g., through a more resilient "tag-and-forward" workflow design, in which faulty intermediate results are tagged and passed on to downstream "exception catchers" (instead of bringing the whole workflow to a halt). Provenance information also plays a critical role here, e.g., for check-pointing. Forward error recovery methods, such as error encoding, automatically detect an erroneous internal workflow state and try to either mask it or transparently recover from it [LC83], [CHI+98], [Vou05]. Our experiences have shown that fault-tolerant software techniques can significantly improve workflow execution success rates. We will improve the Kepler overall workflow reliability by incorporating fault tolerance mechanisms via failure-masking and fail-over through redundancy and virtualization of resources, with emphasis on individual actors/services. Later, we will evaluate check-pointing and recovery, collective service behaviors, and pro-active fault-tolerance.

Develop New Actors. The basic building block within the Kepler architecture is the actor. We will design and develop actors that implement domain-specific functionality, and generic actors, which represent a general capability. Generic actors will provide a single, unified, view of an entire collection of specialized actors, thus freeing the scientist from having to select the "best" implementation for their specific requirements [AKA05, ABS05]. For example, we will implement a generic File Transfer Actor to provide a simplified view of specialized FTP, GridFTP, SRB, and scp actors. This generic actor will encapsulate all of these, providing a single mechanism for performing file transfers between machines.

3.3.3 Enhancing the Workflow Infrastructure

Current workflow systems, including Kepler and SCIRun, are not capable of handling all the distributed, heterogeneous, multi-scale demands of scientific workflows in a way that is easily utilized by application scientists. In order to effectively meet these demands requires a highly "usable" and flexible workflow infrastructure. Furthermore, the deployment mechanism for the entire workflow infrastructure, including the engine, actors, and documentation, needs to be as straightforward as possible, with the complexity of the installation process hidden from the users. Significant extensions to the current SPA infrastructure will be required to meet these goals. Under this proposal, we will extend the existing infrastructure by:

Developing a framework to bridge tightly- and loosely-coupled workflows. Tightly coupled workflow engines, such as SCIRun, are exceptionally good at coordinating activities running on a single machine or small number of similar machines. In these cases, information can be passed between actors as memory pointers, without requiring secondary storage or reliable, high-speed WAN connectivity. This type of high-throughput connectivity is critical for a variety of tasks. For example a simple data analysis workflow may

run an isocontouring algorithm on a terabyte data set, followed by an interactive visualization of the resulting mesh. A tight connection between the two activities is critical to ensure acceptable performance because of the interactive demands. On the other hand, a loosely coupled system, such as Kepler does not have the requirement of memory-to-memory data transfers, and thus may incorporate a broader range of activities and be more appropriate for a distributed, heterogeneous, computational environment. For example, COTS tools, batch processes, and tasks generating too much data to effectively transfer memory-to-memory are all good candidates for loosely-coupled workflows. Most realistic scientific workflows have both tight-coupled and loosely-coupled regions contained within the overall workflow specification. In order for the scientific community to gain the full value of workflows, there needs to be a mechanism for easily combining these regions into a single workflow specification. As part of this proposal, we will extend the existing, preliminary mechanism for coupling Kepler and SCIRun workflows: we will make the data communication between a Kepler actor and SCIRun more expressive, and support data-types appropriately; make the data communication between a Kepler actor and SCIRun more efficient, for example, via low-level socket connections; and develop a user interface that supports development of a coupled or even unified SCIRun / Kepler workflows, while hiding the complexity of the underlying workflow from the scientist.

Supporting distributed workflow control. Due to the distributed nature of their jobs, scientists need to interact with their workflow from multiple locations. While a portal interface will support remote submission, execution and monitoring of a workflow, additional controls are sometimes needed. In particular, a scientist may need to dynamically pause, investigate, modify, and restart a workflow. When actors are executing independently in a distributed environment, this becomes a significant challenge. Given that there may be costs associated with computational resources, determining the correct behavior for these actions becomes more complex. In a portal environment, the workflow engine may also be executing on a machine different than the web server that is driving it. In order to control the workflow engine from a remote machine, which may have intermittent connectivity, we will extend the Kepler workflow engine to support a detached interface by refactoring the current workflow engine into a client-server based architecture, and through virtualization of resources. This new architecture will support a fully-functional detachable (asynchronous or "background") remote workflow engine and interface. Additional extensions will be made to support finer granularity control of actors.

Integrating with DOE security infrastructures. The DOE complex is a collection of institutions, each with their own computer security infrastructure. In order for scientists to effectively utilize a workflow engine in this environment, the ability to remotely execute tasks and transfer data is required. We will work with NERSC and ORNL, among others, to develop and deploy a simple mechanisms for user authentication that meet with their respective computing policies while minimizing the user interaction required during execution of a typical, long-running, scientific workflow. This mechanism will then be generalized to work with other institutions, such as our university partners.

Improving our deployment infrastructure. A reliable, robust deployment infrastructure is critical to ensuring the adoption of our workflow technology framework and components. The deployed system must be available on many platforms, including an easy and effective distribution mechanism and contain a wide variety of relevant actors, including domain-specific actors. Under the SDM Center, we established a web presence, where both a source code distribution and a more user-friendly WebStart distribution of the code can be obtained. This infrastructure will be continued and built upon as part of this proposal. In particular, a new infrastructure that supports downloads of individual actors, domain-specific groups of actors, and workflows will be added to the current distribution mechanisms. This new infrastructure will encourage sharing of actors and workflows among scientists, as well as providing a one-stop location for scientists looking for specific capabilities. Documentation on both the workflow infrastructure and the actors will also be improved. NC State will supply hosting services for infrastructural servers, including workflow virtualization services, version repository services, etc. Finally, we will actively develop and disseminate a tutorial targeted at application scientists that effectively conveys the information required to begin to utilize the SPA workflow infrastructure**.**